



ПРИ ПОДДЕРЖКЕ
**ФОНДА
ПРЕЗИДЕНТСКИХ
ГРАНТОВ**

**Специальное учебно-методическое пособие
по информатике и информационно-коммуникационным
технологиям для 11-12 классов школ
для слепых и слабовидящих детей**

**Пособие разработано:
Общероссийской общественной организацией инвалидов -
Российский союз инвалидов**

Руководитель проекта: Президент Российского союза инвалидов, кандидат
экономических наук Кузнецов Вениамин Алексеевич

**Москва
2022**

При реализации проекта используются средства государственной поддержки, выделенные в качестве гранта Фонда Президентских Грантов

Распространяется бесплатно!

ISBN 978-5-6045213-9-7

Глава 1

Информация и информационные процессы

§1.1. Информация в современном мире

Как вы уже знаете, термин «информация» происходит от латинского слова «informatio», что означает разъяснение, осведомление, изложение. Информация относится к фундаментальным, неопределяемым понятиям науки информатика. Во многом смысл понятия «информация» зависит от сферы его использования:

- В быту под информацией понимают любые сведения, которые кого-либо интересуют. Например, сообщение о каких-либо событиях, о чей-либо деятельности и т.п.;
- В технике под информацией понимают сообщения, передаваемые в форме знаков или сигналов от источника по каналу связи к приемнику;
- Применительно к компьютерной обработке данных, под информацией понимают некоторую последовательность символических обозначений (букв, цифр, звуков, графиков, рисунков и др.), несущую смысловую нагрузку и представленную в понятном компьютеру виде. Физически информация в компьютере записывается и передается в виде электрических сигналов;
- Информатика рассматривает информацию как совокупность связанных между собой сведений, уменьшающих меру неопределенности знаний об окружающем мире.

При изучении информатики важно помнить, что ни одна из приведенных выше трактовок не может считаться строгим определением.

Можно выделить следующие свойства информации:

- Релевантность – способность информации соответствовать нуждам (запросам) потребителя;
- Полнота – свойство информации всеобъемлюще характеризовать отображаемый объект для данного потребителя;
- Своевременность – способность информации соответствовать необходимости в нужный момент времени;

- Достоверность – свойство информации не иметь скрытых ошибок. Достоверная информация со временем может стать недостоверной, если устареет и перестанет отражать истинное положение дел;
- Доступность – возможность получения информации данным потребителем;
- Защищенность – свойство, характеризующее невозможность несанкционированного использования или изменения информации.

Действия, выполняемые с информацией, называются информационными процессами. Информационный процесс – это процесс приема, передачи (обмена), преобразования и использования информации.

В информационных процессах выделяются следующие виды: получение, хранение, передача, обработка, использование информации.

Деятельность человека, которая связана с процессами получения, преобразования, накопления, передачи и использования информации называют информационной деятельностью.

В информатике наряду с понятием «информация» существует понятие «данные». Данные – это результаты наблюдений над объектами и явлениями, которые по каким-либо причинам не используются, а только хранятся. Как только данные начинают использовать в практических целях, они превращаются в информацию. Исходя из этого, можно сказать, что информация – это используемые данные.

Данные всегда находятся на каком-либо источнике (хранилище). В последнее время количество данных достигло невероятного роста. Это было вызвано быстрым развитием сети Интернет.

Измерить данные нельзя. Как только мы станем подсчитывать данные, начнется процесс обработки, и данные превратятся в информацию. В отличие от данных Информацию измерить можно. О том, как это сделать, будет рассказано в параграфе 1.5.

Поясним разницу между «данными» и «информацией» на следующем примере. Пусть, например, есть таблица, содержащая цены на билеты во все кинотеатры вашего города. Эта таблица будет представлять собой данные. Если начать её обрабатывать отыскивая цену

на билет в ближайший кинотеатр, то найденная цена будет представлять собой уже информацию.

Как вы уже знаете, наиболее универсальным устройством обработки информации является компьютер. Первые электронно-вычислительные машины (ЭВМ), которые могли автоматически по заданной программе обрабатывать большие объемы информации, были созданы в 1946 году в США (ЭНИАК) и в 1950 году в СССР (МЭСМ). В 40 – 60-х годах производство ЭВМ измерялась единицами, десятками и, в лучшем случае, сотнями штук. ЭВМ были очень дорогими и очень большими, они занимали целые залы и поэтому оставались недоступными для массового потребителя.

Массовое производство сравнительно недорогих персональных компьютеров началось с середины 70-х годов XX века с компьютера Apple II. Именно с этого компьютера начала свое существование компания Apple Computer. Количество производимых персональных компьютеров начало составлять десятки тысяч в год, что по тем временам было колоссальным достижением.

В начале 80-х годов приступила к массовому производству персональных компьютеров уже давно существовавшая корпорация IBM (International Business Machines). Эти компьютеры так и назывались IBM Personal Computer – IBM PC. Достаточно скоро IBM – совместимые компьютеры стали выпускать многие фирмы.

Персональный компьютер стал доступен массовому потребителю, и теперь в развитых странах мира компьютер имеется на большинстве рабочих мест и в большинстве семей.

Обычно процесс развития вычислительной техники разделяют на четыре этапа выделяя, соответственно, четыре поколения компьютеров.

ЭВМ первого поколения (с середины 40-х годов XX века) были основаны на использовании электронных ламп. Ламповые ЭВМ отличаются большими габаритами, большим потреблением энергии, малой скоростью работы, низкой надежностью. Программировались эти машины в специальных кодах.

ЭВМ второго поколения (с конца 50-х годов XX века) работали на полупроводниковых деталях (транзистор, диод). По сравнению

с ЭВМ предыдущего поколения были улучшены все технические характеристики. Для программирования использовались уже алгоритмические языки.

ЭВМ третьего поколения (с середины 60-х годов XX века) были основаны на использовании интегральных схем и печатных плат. У этих машин значительно уменьшились габариты, повысилась надежность, увеличилась производительность.

Четвертое поколение (с конца 70-х годов XX века по настоящее время) – это компьютеры, основанные на микропроцессорах. У них улучшены все технические характеристики. Стал возможен массовый выпуск персональных компьютеров. Появились мощные многопроцессорные вычислительные системы с высокой производительностью, произошло Внедрение во все сферы человеческой деятельности компьютерных сетей. Повсеместное использование компьютерных информационных технологий стало нормой.

Таким образом, появление нового типа ЭВМ было обусловлено развитием микроэлектроники. С позиций информатики информационную революцию можно связать с появлением ЭВМ 4-го поколения – персонального компьютера, позволяющего решать проблему хранения, обработки и передачи информации на качественно новом уровне.

В настоящее время на передний план вышла новая отрасль человеческой деятельности – информационные технологии (ИТ). Согласно определению, принятому ЮНЕСКО, ИТ – это комплекс взаимосвязанных, научных, технологических, инженерных дисциплин, изучающих методы эффективной организации труда людей, занятых обработкой и хранением информации; вычислительную технику и методы организации и взаимодействия с людьми и производственным оборудованием, их практические приложения, а также связанные со всем этим социальные, экономические и культурные проблемы.

В 70-е годы XX века компьютер стал доступен незрячим пользователям. Появились и стали совершенствоваться брайлевские дисплеи, синтезаторы речи, программы невидимого доступа к информации, «читающие» машины и другие тифлоинформационные

(тифлотехнические) устройства. Будем понимать под тифлоинформационными техническими и программными средствами, позволяющими людям с частично или полностью отсутствующим зрением получать, обрабатывать и передавать информацию.

В настоящее время персональный компьютер оснащенный необходимым программным обеспечением и брайлевским дисплеем позволяет пользователю с нарушением зрения обрабатывать, создавать, хранить и передавать информацию почти без ограничений, связанных с отсутствием зрения. Наша задача состоит в том, что бы овладеть методами невизуального доступа к информации в широком смысле и стать полноправными членами современного общества.

С середины XX века, с момента появления электронных устройств обработки и хранения информации (ЭВМ, а затем персонального компьютера), начался постепенный переход от индустриального общества к информационному. В информационном обществе главным ресурсом является информация, именно на основе владения информацией о самых различных процессах и явлениях можно эффективно и оптимально строить любую деятельность.

Информационная революция, произошедшая в 70-х годах прошлого века, привела к тому, что человеческая цивилизация в конце XX столетия оказалась в состоянии перехода от индустриальной фазы своего развития к информационной.

В информационном обществе использование компьютеров во всех сферах человеческой деятельности обеспечит доступ к надежным источникам информации, избавит людей от рутинной работы, ускорит принятие оптимальных решений, автоматизирует обработку информации в производственной и социальной сферах. В результате движущей силой развития общества должно стать производство информационного, а не материального продукта. В информационном обществе изменится не только производство, но и весь уклад жизни, система ценностей, возрастет значимость культурного досуга. По сравнению с индустриальным обществом, где все направлено на производство и потребление товаров, в информационном обществе интеллект и знания — это средство и продукт производства, что, в свою очередь, приведет к увеличению доли умственного труда.

От человека потребуется способность к творчеству, возрастет спрос на знания.

Информационное общество – это общество, в котором большая часть населения занята получением, переработкой, передачей и хранением информации.

Школьный курс информатики и информационно-коммуникационных технологий играет особую роль в эпоху перехода от индустриального общества к информационному, так как готовит выпускников школы к жизни и деятельности в информационном обществе.

В современном обществе человек должен обладать информационной культурой, то есть иметь не только знания и умения в области информационных технологий, но и владеть определенными юридическими и этическими нормами в этой сфере.

Необходимость упорядочить информацию, например, о людях, с которыми вы контактируете, требует использования записной книжки. Однако часто удобнее использовать для хранения такой информации компьютерную базу данных «Записная книжка». При поиске информации в современной библиотеке или в Интернете необходимо иметь навыки поиска информации в базах данных. В информационном обществе очень полезным является умение создавать базы данных, а также вести в них поиск.

Квалифицированный пользователь компьютера может на основе применения средств объектно-ориентированного программирования создавать необходимые ему специализированные приложения. Например, можно создать приложение, которое автоматизирует заполнение многочисленных квитанций оплаты за квартиру, электроэнергию, газ и др.

Современному человеку необходимо овладеть коммуникативной культурой, то есть умениями создавать и посылать электронные письма, находить нужную информацию в сети Интернет или в файловых архивах, участвовать в чатах и так далее. Необходимым условием успешной профессиональной деятельности становится создание и публикация в Интернете Web-сайтов с информацией о деятельности организации или предприятия.

Информационная культура состоит не только в овладении определенным комплексом знаний и умений в области информационных и коммуникационных технологий, но предполагает знание и соблюдение юридических и этических норм и правил. Законы запрещают использование пиратского компьютерного обеспечения и пропаганду насилия, наркотиков и порнографии в Интернете. Общение с помощью электронной почты или в чатах, участие в телеконференциях предполагают соблюдение определенных правил и норм.

Контрольные вопросы

1. Как вы понимаете термин «информация»?
2. Какие свойства информации вы знаете?
3. Что такое информационный процесс?
4. Какие виды информационных процессов вы знаете?
5. Приведите примеры информационных процессов.
6. В каких областях деятельности человека преобладают информационные процессы?
7. Что называют информационной деятельностью человека?
8. В чем разница между понятиями «информация» и «данные»?
9. Перечислите поколения ЭВМ и дайте их краткую характеристику.
10. В чем значение программ невизуального доступа к информации на экране компьютера?
11. Как вы представляете информационное общество?
12. Как вы думаете, по каким основным параметрам можно судить о степени развитости информационного общества и почему?
13. Как изменяется содержание жизни и деятельности людей в процессе перехода от индустриального к информационному обществу?
14. Является ли наше общество информационным? Обоснуйте ответ.
15. Каковы основные компоненты информационной культуры, которые необходимы человеку для жизни в информационном обществе?

§1.2. Формы представления информации

Как уже говорилось выше, информация является неопределяемым понятием, значение которого зависит от контекста, в котором оно используется. Информация может быть выражена в самых разнообразных формах. По типу восприятия среди них обычно выделяют следующие, наиболее употребительные формы представления информации :

- Текстовая – закодированная специальными знаками (буквами) речь человека;
- Графическая – картины, фотографии, чертежи, голограммы, различные виды реального мира (например, северное сияние), географические карты и т.п.;
- Видеоинформация – последовательность кадров (графическая форма), следующих друг за другом с некоторой частотой (фильм);
- Звуковая – речь, музыка, серена, дверной звонок, инфра-низкие звуки от землетрясений, ультразвук в медицине и технике и т.п.;
- Числовая – количественная мера объектов и их свойств, закодированная специальными знаками аналогично текстовой;
- Тактильная – воспринимаемая человеком при касании, либо с помощью специализированных датчиков;
- Органолептическая – передаваемая через органы чувств человека (запах, вкус и др.).

Пользователю персонального компьютера в подавляющем большинстве случаев приходится работать с текстовой, графической и звуковой формой представления информации. Именно эти три формы мы будем изучать более подробно в следующей главе этой книги.

Форма представления информации очень важна при ее передаче и восприятии, поскольку в зависимости от цели, которую Вы перед собой поставили, одна и та же информация может быть представлена в различных формах.

В информатике с понятием информации тесно связаны такие понятия, как сигнал и сообщение.

Сигнал – это любой процесс, несущий информацию (электрический сигнал в проводах, радиосигнал в эфире, свет, воспринимаемый телескопом и т.д.).

Сообщение – это информация, представленная в определенной форме и предназначенная для передачи.

Для передачи сообщений используют языки. Основу любого языка составляет алфавит – набор определенных знаков (символов), с помощью которых представляется информация. Языки делятся на естественные (разговорные) и формальные. Алфавит естественных языков зависит от национальных традиций. Формальные языки встречаются в специальных областях человеческой деятельности (математике, физике, химии, информационных технологиях и т. д.).

Так, например, русский язык использует хорошо известный вам алфавит из 33-ех букв. С помощью него вы можете написать или прочитать произвольное сообщение. Данная книга закодирована с помощью этого алфавита. Многие естественные языки используют латинский алфавит из 26-ти букв (английский, казахский и др.).

Вся цифровая техника для работы с информацией использует формальный язык, основанный на алфавите из двух символов 0 и 1. Способ представления информации с помощью языка, алфавит которого состоит всего из двух символов, предложил еще в XVII веке знаменитый немецкий ученый Готфрид Вильгельм Лейбниц.

Сегодня такой способ представления информации широко используется во всех компьютерах и других цифровых устройствах. Эти два символа 0 и 1 принято называть двоичными цифрами или битами (от англ. bit – Binary Digit – двоичный знак). Техническая реализация такого алфавита оказалась наиболее простой. В электронных устройствах 0 обычно представляется низким напряжением, а 1 – высоким. В запоминающих устройствах используются бистабильные ячейки, одно состояние которых принимается за ноль, а второе – за единицу. В волоконно-оптических линиях связи 0 – отсутствие светового сигнала, а 1 – наличие светового сигнала.

Представление информации с помощью формального языка называют кодированием. Кодированию посвящена следующая глава этой книги.

Как вы знаете, люди с нарушением зрения при работе с информацией используют тифлоинформационные средства, т.е. средства, позволяющие инвалидам по зрению получать, создавать, обрабатывать

и передавать информацию. Самым распространенным видом тифло-информационных средств в настоящее время является персональный компьютер, оснащенный программой невидимого доступа к информации и брайлевским (тактильным) дисплеем.

В этой книге рассматриваются невидимые приемы работы с информацией на персональном компьютере с соответствующим программным и аппаратным обеспечением. Подобные приемы делают возможным создание, получение, обработку и передачу информации в различных формах пользователем с глубоким нарушением зрения.

Контрольные вопросы

1. Какие формы представления информации вы знаете?
2. Приведите примеры информации:
 - А) В текстовой форме;
 - Б) В графической форме;
 - В) В звуковой форме.
3. Что такое органолептическая информация?
4. Приведите пример одной и той же информации, представленной в различных формах.
5. Что такое сигнал в информатике?
6. Что такое сообщение в информатике?
7. В чём разница между естественным и формальным языками?
8. Какой алфавит используется в компьютерной технике?
9. В чём преимущества формального языка, основанного на алфавите из двух символов?
10. Расскажите о технической реализации двоичного алфавита.
11. Что такое кодирование?
12. В каких формах используют информацию незрячие люди? Приведите примеры.

§1.3. Различные системы счисления

Система счисления — это совокупность приемов и правил, по которым числа записываются и читаются.

Существуют позиционные и непозиционные системы счисления. В непозиционных системах счисления вес цифры (т.е. тот вклад, который она вносит в значение числа) не зависит от ее позиции в записи числа. Так, например, в римской системе счисления в числе XXIII (23) вес цифры X в каждой из двух позиций равен 10.

В позиционных системах счисления вес каждой цифры изменяется в зависимости от ее положения (позиции) в последовательности цифр в записи числа. Например, в числе 112 первая цифра 1 означает одну сотню, вторая цифра 1 означает один десяток. Подобная запись числа 112 означает сокращенную запись выражения

$$1*10^2 + 1*10^1 + 2*10^0$$

Любая позиционная система счисления характеризуется своим основанием. Основание позиционной системы счисления равно количеству различных цифр, используемых для записи чисел в данной системе счисления.

За основание системы счисления можно принять любое натуральное число — 2, 3, 4 и т.д. Таким образом, существует бесконечно много различных систем счисления.

Запись натуральных чисел в любой позиционной системе счисления с основанием q означает сокращенную форму следующей записи:

$$A_{n-1} * q^{n-1} + a_{n-2} * q^{n-2} + \dots + a_1 * q^1 + a_0 * q^0,$$

где a_i — цифры системы счисления, а n — число разрядов.

Например:

$$2*100^2 + 5*10^1 + 6*10^0 = 256$$

Кроме десятичной в информатике используются еще три позиционные системы счисления с основанием, являющимся степенью числа 2, а именно:

- Двоичная, использующая цифры 0 и 1;
- Восьмеричная, использующая цифры 0, 1, 2, 3, 4, 5, 6 и 7.
- Шестнадцатеричная, использующая для первых десяти целых чисел (от 0 до 9) цифры 0, 1, ..., 9, а для следующих шести чисел (от 10 до 15) — в качестве цифр используются буквы латинского алфавита A, B, C, D, E, F.

Например, число 12 в шестнадцатеричной системе записывается буквой С.

Обычно, после записи числа указывается основание системы счисления сниженными цифрами, в которой записано это число. По Брайлю, основание системы счисления пишется с помощью знака нижнего индекса (точки 16) сниженными цифрами без цифрового знака (в десятичной системе основание обычно не указывают). Например, шестнадцатеричные числа от 9 до 17 записываются следующим образом:

$$9 = 9_{16}$$

$$10 = A_{16}$$

$$11 = B_{16}$$

$$12 = C_{16}$$

$$13 = D_{16}$$

$$14 = E_{16}$$

$$15 = F_{16}$$

$$16 = 10_{16}$$

$$17 = 11_{16}$$

Из всех систем счисления наиболее простую техническую реализацию, как уже говорилось, имеет двоичная. Однако, для человека двоичная система счисления неудобна из-за большого количества разрядов при записи сколько-нибудь больших чисел. Но, чтобы профессионально использовать компьютер, следует научиться понимать его язык. Для этого и применяются восьмеричная и шестнадцатеричная системы. Числа в этих системах читаются почти так же легко, как десятичные, но требуют соответственно в три (восьмеричная) и в четыре (шестнадцатеричная) раза меньше разрядов, чем в двоичной.

Для перевода восьмеричного числа в двоичную систему достаточно каждую восьмеричную цифру заменить равной ей двоичной триадой (тройкой двоичных цифр). Например, триада 101_2 равна восьмеричной цифре 5_8 .

Для перевода шестнадцатеричного числа в двоичную систему следует каждую его цифру заменить равной ей двоичной тетрадой

(четверкой двоичных цифр). Например, тетрада 1101_2 равна шестнадцатеричной цифре D_{16} .

Чтобы перевести двоичное число в восьмеричную (или шестнадцатеричную) систему счисления, его нужно разбить, отсчитывая цифры справа налево на триады (тетрады) и каждую такую группу заменить соответствующей восьмеричной (шестнадцатеричной) цифрой.

Вы уже знакомы с алгоритмом перевода десятичного числа в двоичную, восьмеричную или шестнадцатеричную систему счисления. Напомним этот алгоритм в общем виде.

Для перевода натурального десятичного числа N в систему счисления с основанием q следует разделить N с остатком на q . Затем неполное частное нужно снова разделить с остатком на q , и т.д. Этот алгоритм следует продолжать пока получаемое неполное частное от-
лично от нуля. Как только в неполном частном будет получен нуль, алгоритм завершится. Представлением числа N в новой системе счисления будет последовательность остатков от деления, записанных в порядке, обратном порядку их получения. Все остатки записываются цифрами q -ичной системы счисления.

При решении задач, связанных с переводом десятичных чисел в двоичную систему счисления, часто пользуются более простым и быстрым способом. Он состоит в представлении переводимого в двоичную систему числа в виде суммы степеней двойки. По такому представлению сразу можно сделать вывод о том, на каком месте в двоичной записи стоит 0, а на каком 1.

Рассмотрим этот способ на примере. Пусть необходимо перевести в двоичную систему число 155. Будем действовать по следующему алгоритму:

1. Подберем максимальную степень числа 2, не превосходящую 155. Это $128 = 2^7$

2. Вычтем из числа 155 эту степень:

$$155 - 128 = 27$$

3. теперь подберем максимальную степень числа 2, не превосходящую 27.

$$\text{Это } 16 = 2^4$$

4. Вычтем из числа 27 эту степень:

$$27 - 16 = 11$$

5. Действуя аналогичным образом далее, получим представление:

$$155 = 2^7 + 2^4 + 2^3 + 2^1 + 2^0$$

6. Видим, что 155 представлено в виде суммы 7-ой, 4-ой, 3-ей, 1-ой и 0-ой степеней числа 2. Теперь запишем в двоичном представлении на месте разряда, соответствующего входящей в представление степени двойки число 1, а на месте разрядов, соответствующих не входящей в данное представление степени (например, на месте 6-ого разряда), запишем 0. Таким образом, получим:

$$155 = 10011011_2$$

Т.е. на месте 0-го разряда стоит 1, поскольку 2^0 входит в данное представление. На месте 1-ого разряда тоже 1, поскольку 2^1 также входит в представление. А на месте 2-ого разряда записан 0, поскольку 2^2 уже не входит в него. Напомним, что разряды нумеруются справа налево начиная с нуля.

Чтобы легко пользоваться этим алгоритмом, полезно выучить степени двойки до десятой включительно!

Перевод в десятичную систему счисления числа x , записанного в q -ичной системе, сводится к вычислению значения многочлена:

$$x = a_{n-1} * q^{n-1} + a_{n-2} * q^{n-2} + \dots + a_0 * q^0$$

средствами десятичной арифметики.

Правила выполнения арифметических операций в столбик (сложения, вычитания, умножения и деления) во всех позиционных системах счисления одинаковы. Следует только помнить, что таблицами сложения и умножения надо пользоваться своими для каждой системы счисления.

Рассмотрим несколько примеров решения задач на различные системы счисления.

Пример 1. Укажите целое число от 8 до 11, двоичная запись которого содержит ровно две единицы. Если таких чисел несколько, укажите наибольшее из них.

Решение. Указанному числовому промежутку принадлежат 4 целых числа. Переведём их в двоичную систему счисления. Для этого

воспользуемся способом, при котором число представляется в виде слагаемых, являющихся степенями числа 2.

$$8 = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 1000_2,$$

$$9 = 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 1001_2,$$

$$10 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 1010_2,$$

$$11 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 1011_2.$$

Указанному требованию соответствуют числа 9 и 10. Эти числа содержат в своей двоичной записи ровно две единицы.

Из чисел 9 и 10 выбираем наибольшее. Это Число 10.

Ответ: 10.

Пример 2. Вычислите сумму чисел x и y при $x = B3_{16}$, $y = 110110_2$. Результат представьте в десятичной системе счисления.

Решение. Выполним перевод каждого числа в десятичную систему счисления:

$$B3_{16} = 11 \cdot 16^1 + 3 \cdot 16^0 = 176 + 3 = 179.$$

$$110110_2 = 1 \cdot 2^5 + 1 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0 = 32 + 16 + 4 + 2 = 54.$$

Выполним сложение полученных чисел:

$$179 + 54 = 233.$$

Ответ: 233.

Пример 3. Даны 4 целых числа, записанных в шестнадцатеричной системе:

$A8$, AB , $B5$, CA .

Сколько среди них чисел, больших, чем 265_8 ?

Решение. Представим все числа в десятичной системе счисления:

$$A8_{16} = 10 \cdot 16^1 + 8 \cdot 16^0 = 160 + 8 = 168,$$

$$AB_{16} = 10 \cdot 16^1 + 11 \cdot 16^0 = 160 + 11 = 171,$$

$$B5_{16} = 11 \cdot 16^1 + 5 \cdot 16^0 = 176 + 5 = 181,$$

$$CA_{16} = 12 \cdot 16^1 + 10 \cdot 16^0 = 192 + 10 = 202,$$

Теперь переведем число, с которым будем проводить сравнения:

$$265_8 = 2 \cdot 8^2 + 6 \cdot 8^1 + 5 \cdot 8^0 = 128 + 48 + 5 = 181.$$

Сравнивая четыре первых результата с числом 181, приходим к выводу, что только одно число из четырех представленных больше, чем 265_8 .

Ответ: 1.

Пример 4. Укажите номер неравенства, которое выполняется для чисел:

$$a = 164_8,$$

$$b = A3,$$

$$c = 2200_4$$

Неравенства:

$$1) a < b < c;$$

$$2) a < c < b;$$

$$3) b < a < c;$$

$$4) c < b < a.$$

Решение. Выполним сравнение в десятичной системе счисления. Сделаем перевод чисел:

$$a = 164_8 = 1 \cdot 8^2 + 6 \cdot 8^1 + 4 \cdot 8^0 = 64 + 48 + 4 = 116,$$

$$b = A3_{16} = 10 \cdot 16^1 + 3 \cdot 16^0 = 163,$$

$$c = 2200_4 = 2 \cdot 4^3 + 2 \cdot 4^2 + 0 \cdot 4^1 + 0 \cdot 4^0 = 2 \cdot (64 + 16) = 160.$$

Так как, в условии задачи все неравенства одного знака, расположим результаты в порядке возрастания. Получим:

$$116 < 160 < 163.$$

Это соответствует неравенству:

$$a < c < b.$$

Данное неравенство расположено под номером 2.

Ответ: 2.

Контрольные вопросы

1. Что такое система счисления?
2. В чем разница между позиционной и непозиционной системами счисления?
3. Что такое «основание системы счисления»?
4. Сколько существует различных систем счисления? Почему?
5. Зачем используются восьмеричная и шестнадцатеричная системы счисления?
6. Какие алгоритмы перевода десятичного числа в двоичную систему счисления вы знаете?
7. Как перевести число из восьмеричной системы счисления в двоичную?

8. Как перевести число из двоичной системы счисления в шестнадцатеричную?

9. Как обозначаются цифры в шестнадцатеричной системе счисления?

§1.4. Представление чисел в компьютере

В компьютере числа можно представлять различными способами. Рассмотрим наиболее распространенные из них. Для описания этих способов представления чисел необходимо познакомиться со следующими терминами:

- Машинное слово;
- Фиксированная запятая или фиксированная точка;
- Прямой код;
- Дополнительный код;
- Обратный код.

Мы не будем давать строгого определения этих терминов, а поймём их смысл из текста ниже.

Как вы уже знаете, компьютер обрабатывает информацию в двоичном коде. Целые числа хранятся в памяти компьютера в формате с фиксированной запятой или фиксированной точкой. В этом случае каждому разряду ячейки памяти соответствует разряд двоичной записи числа, а запятая (или точка) находится справа после младшего разряда.

Множество целых чисел, которые можно записать в память компьютера, ограничено и зависит от размера ячеек памяти (машинного слова), используемых для их хранения. В k -разрядной ячейке может храниться 2_k различных значений целых чисел.

Чтобы представить, как в памяти компьютера хранятся целые неотрицательные числа, рассмотрим алгоритм получения внутреннего представления целого числа N , хранящегося в k -разрядном машинном слове:

1. Перевести число N в двоичную систему счисления.
2. Полученное двоичное представление дополнить слева незначащими нулями до k разрядов.

3. Записать полученную последовательность двоичных цифр (бит) в ячейку памяти, соответствующую машинному слову.

При размере ячейки памяти в 1 байт имеется возможность представить все числа в диапазоне от 00000000 до 11111111 (в двоичной системе) или от 0 до 255 (в десятичной системе). Большой размер ячейки памяти позволяет закодировать большой диапазон чисел. В дальнейшем будем считать, что машинное слово состоит из 8 бит (1 байт).

Итак, для хранения целого неотрицательного числа отводится одно машинное слово (ячейка памяти), состоящее из 8 бит. Для хранения целых чисел со знаком старший (левый) бит отводится для знака числа, а 7 оставшихся для хранения его модуля. Если число положительное, то в знаковый разряд записывается 0, если число отрицательное записывается 1. Такое представление целых чисел называется прямым кодом числа.

Например, число – 5 в прямом коде будет представлено двоичной записью 10000101. Проверьте это самостоятельно.

Заметим, что в ячейку размером 1 байт прямым кодом можно записать лишь числа, модуль которых не превосходит $2^7 - 1$. Поэтому в компьютере используются машинные слова большего размера (2, 4 или больше байт).

Существует и другой способ представления целых чисел. Рассмотрим алгоритм получения внутреннего представления целого отрицательного числа N , хранящегося в k -разрядном машинном слове :

1. Перевести число N в двоичную систему счисления.
2. Полученное двоичное представление дополнить слева незначащими нулями до k разрядов.
3. Получить обратный код числа N заменой во всех позициях 0 на 1 и 1 на 0, т.е. инвертировать значения всех бит.
4. К полученному числу прибавить 1.
5. Записать полученную последовательность двоичных цифр (бит) в ячейку памяти, соответствующую машинному слову.

Данная форма представления отрицательных целых чисел называется дополнительным кодом. Удобно хранить неотрицательные целые числа в прямом коде, а отрицательные в дополнительном.

Рассмотрим подробнее два следующих примера.

Пример 1. *Некоторый компьютер работает только с целыми неотрицательными числами. Каков диапазон изменения чисел, если для их представления в памяти компьютера отводится 1 байт?*

Решение: В 1 байт можно записать $2^8 = 256$

Следовательно искомый диапазон от 0 до 255.

Ответ: от 0 до 255.

Пример 2. *Каков диапазон изменения целых чисел (положительных и отрицательных), если в памяти компьютера для представления такого числа отводится 1 байт.*

Решение: Поскольку 1 байт может содержать максимум 256 чисел, то диапазон значений положительных и отрицательных чисел в равном количестве рассчитаем так:

$$256/2 = 128$$

Минимальное отрицательное число равно – 128. Так как число 0 также входит в этот диапазон, то максимальное положительное число будет равно 127

Ответ: от – 128 до 127.

Рассмотрим алгоритм перевода дополнительного кода в десятичное число:

1. Инвертировать дополнительный код.
2. Прибавить к полученному двоичному числу 1 и получить модуль отрицательного числа.
3. Перевести полученный результат в десятичное число и приписать знак отрицательного числа.

Действительные (вещественные) числа часто записываются в компьютере в экспоненциальной форме. Вы сталкивались с такой записью чисел при изучении программирования. Напомним, что в языке Python для записи очень больших или очень маленьких по модулю чисел используется форма с плавающей точкой. В этом случае число представляется в виде произведения некоторой десятичной дроби, называемой мантиссой, и степени числа 10, причём показатель степени является целым (возможно отрицательным) числом. Числа

с плавающей точкой в программах на языке Python записываются по следующему правилу:

1. Сначала записывается мантисса.
2. Затем идёт буква *e*.
3. Затем записывается целое число со знаком, являющееся показателем степени числа 10.

Пробелы внутри такой записи не допускаются. Например, расстояние от земли до солнца будет записываться как 1.496e11, а масса молекулы воды 2.99e-23.

Подчеркнем, что приведенное описание экспоненциальной формы записи действительных чисел относится к их представлению на экране компьютера и к вводу с помощью клавиатуры. Как такие числа хранятся в памяти компьютера здесь описываться не будет. Вы можете самостоятельно узнать это используя ресурсы сети Интернет.

Контрольные вопросы

1. Объясните что означают следующие термины:
 - А) Машинное слово;
 - Б) Фиксированная запятая или фиксированная точка;
 - В) Прямой код;
 - Г) Дополнительный код;
 - Д) Обратный код.
2. Почему множество целых чисел, которые можно записать в память компьютера, ограничено?
3. Расскажите как получить внутреннее представление целого числа со знаком в прямом коде.
4. Расскажите как получить внутреннее представление целого числа со знаком в дополнительном коде.
5. Что значит инвертировать биты в двоичной записи числа?
6. Расскажите как преобразовать дополнительный код в десятичное число.
7. Что такое экспоненциальная форма записи действительного числа?

8. Где используется экспоненциальная форма записи действительных чисел?

§1.5. Измерение объема информации

Как уже говорилось выше, дать строгое определение понятию «информация» нельзя. Однако, информацию можно измерить. Точнее, её количество (или объём) можно задать числом.

Общепринято за единицу измерения объёма информации брать бит. Это количество информации, которое можно передать в сообщении, состоящем из одного двоичного знака 0 или 1. Информация в один бит уменьшает неопределенность знания о предмете в два раза. Так, например, сообщение о том, что подброшенная монета упала «решкой» вверх, несет в себе один бит информации. Действительно, неопределенность знания о результате опыта с подбрасыванием монеты состоит в том, что возможно два равновероятных исхода. После того, как конкретный исход стал известен, неопределенность уменьшилась в два раза, что и соответствует одному биту информации.

На практике чаще используется более крупная единица – байт, равная 8 битам. Один байт информации можно передать с помощью одного символа кодировки ASCII (Об этом подробнее будет сказано в следующей главе). Используются также следующие кратные единицы измерения количества информации:

- 1 килобайт (1 Кб) равен 2^{10} байт (1024 байт);
- 1 мегабайт (1 Мб) равен 2^{20} байт (1024 Кб);
- 1 гигабайт (1 Гб) равен 2^{30} байт (1024 Мб);
- 1 терабайт (1 Тб) равен 2^{40} байт (1024 Гб).

Принцип измерения информации можно проиллюстрировать на следующем примере. Предположим имеется 8 монет, одна из которых фальшивая, а 7 настоящие. Эта фальшивая монета тяжелее настоящей. С помощью рычажных весов мы будем определять какая именно монета фальшивая.

Первый шаг. Разложим монеты на две кучки по 4 монеты в каждой и поместим каждую кучку на свою чашу рычажных весов. Перевесит та чаша, на которой находилась кучка монет, содержащая фальшивую. Теперь надо обнаружить фальшивую среди четырёх

монет. Таким образом неопределенность уменьшилась в 2 раза и мы получили информацию в 1 бит. Эта информация состоит в том, что фальшивая монета находится уже среди четырёх монет.

Второй шаг. Аналогично первому шагу разложим 4 монеты на чаши рычажных весов по две на каждую чашу. Опять перевесит чаша, на которой находится фальшивая монета и неопределенность уменьшится ещё в 2 раза. Теперь фальшивой является одна из двух монет. Мы получили ещё один бит информации.

Третий шаг. Положим каждую монету на свою чашу весов. Перевесит фальшивая монета и неопределенность исчезнет сократившись в 2 раза. В результате получаем третий бит информации.

Вывод. Объём информации о том, какая именно монета из восьми является фальшивой, равен 3 бита.

Рассмотрим ещё один пример. Пусть нам надо найти человека, проживающего в двухэтажном доме с двумя подъездами и двумя квартирами в каждом подъезде на каждом этаже. Легко подсчитать, что в доме всего 8 квартир. В какой именно квартире проживает этот человек знает консьерж. Однако, консьерж умеет отвечать на вопросы только «да» или «нет». В данном случае ответ на один вопрос несет один бит информации, так как из двух возможных исходов выбирается один, т.е. неопределенность уменьшается в два раза. Наша задача за наименьшее количество вопросов выяснить, где живет разыскиваемый человек.

Зададим первый вопрос: «живет ли этот человек в первом подъезде?» Получив, например, ответ «нет», делаем вывод, что он живет во втором подъезде.

Зададим второй вопрос: «живет ли этот человек на первом этаже?» Получив, например, ответ «да», мы ограничим количество вероятных квартир до двух.

Зададим третий вопрос: «живет ли этот человек в квартире слева?» Получив, например, ответ «нет», мы точно устанавливаем, что искомый человек живет в квартире справа.

Записав полученные ответы и заменив все «да» единицами, а «нет» – нулями, мы получим сообщение в виде последовательности из трех двоичных цифр. Таким образом, информация, которую мы

получили от консьержа, равна трем битам. Это минимальное количество вопросов, которые надо задать, чтобы получить данную информацию.

Описанный в этих примерах способ поиска объекта носит название метода деления пополам.

Приведем еще пример. Пусть в классе 32 ученика. Учитель решил спросить одного из них. Какое минимально возможное количество вопросов надо задать учителю, чтобы определить, кого именно он решил спросить?

Если в классе 4 ряда парт, то сначала зададим учителю вопрос: «Сидит ли задуманный ученик на парте в первом или втором рядах?» Получив ответ «да» или «нет», мы сократим количество «подозреваемых» до 16.

Вторым вопросом можно определить конкретный ряд, на котором сидит искомый школьник, сократив выбор до 8 человек. Далее будем поступать аналогично. После каждого ответа число «подозреваемых» сокращается вдвое.

После четвертого вопроса выбор останется сделать из двух учеников. Это можно осуществить, задав пятый вопрос.

Записав полученные ответы и заменив все «да» единицами, а «нет» – нулями, мы получим сообщение в виде последовательности из пяти двоичных цифр, т.е. в результате мы получаем 5 бит информации.

Таким образом, для отгадывания задуманного ученика из 32-х школьников достаточно задать 5 вопросов указанного выше вида. Но если задавать вопросы не оптимальным образом, то может понадобиться большее количество вопросов. при не оптимальном выборе вопросов возможно, что какие-либо два вопроса, ответ на каждый из которых несет один бит информации, в сумме содержат меньше двух бит информации. Так может получиться, если ответ на первый вопрос полностью или частично содержит ответ на второй. Например, если сначала спросить, сидит ли искомый ученик на первом или втором ряду, а затем спросить: «сидит ли искомый ученик на третьем или четвертом ряду?» Очевидно, что второй ответ не добав-

ляет никакой информации, и общее количество информации в двух ответах равно одному биту, а не двум.

Для того чтобы измерить количество информации в сообщении, надо закодировать сообщение в виде последовательности нулей и единиц наиболее рациональным способом, позволяющим получить самую короткую последовательность. Длина полученной последовательности нулей и единиц и является мерой количества информации в битах.

Предположим теперь, что надо выбирать задуманного ученика уже среди 24-х человек. В этом случае нам понадобится не меньше 4 и не больше 5 вопросов, если действовать методом деления пополам. После третьего вопроса у нас останется три “подозреваемых”. Их можно разделить на группу из одного и группу из двух учеников. Тогда после четвертого вопроса мы либо сразу найдем нужного школьника, либо придется задавать пятый вопрос. Значит, количество информации, требуемой, чтобы выбрать задуманного ученика из 24 человек, больше 4 бит и меньше 5.

В общем случае вычислить количество (или объём) информации можно с помощью формулы Шеннона-Хартли $I = \log_2 N$, где N число равновероятных событий, а I – количество информации.

Контрольные вопросы

1. В каких единицах измеряют объем информации?
2. Расскажите о жизненных ситуациях, в которых мы получаем ровно один бит информации.
3. Какие кратные единицы измерения объема информации вы знаете?
4. В чем состоит метод деления пополам?
5. Может ли объем информации выражаться дробным числом?
6. Для чего нужна формула Шеннона-Хартли?

Глава 2

Кодирование информации в компьютере

§2.1. Кодирование текстовой информации

Информация, выраженная с помощью естественных и формальных языков в письменной форме, обычно называется текстовой информацией. Текстовая информация представляется в виде последовательности знаков какого-либо алфавита и служебных символов. В компьютере каждому знаку сопоставляется двоичное число в соответствии с кодовой таблицей. Существуют различные кодовые таблицы, например, КОИ-7, ASCII, CP1251.

Для вывода на монитор текстовая информация подвергается декодированию. Вместо цифрового кода на экран дисплея выводится изображение символа. Набор изображений символов содержится в файле шрифта.

Как уже говорилось, для обработки текстовой информации на компьютере она должна быть представлена в двоичном коде. Код – набор символов (условных обозначений) для представления информации.

Кодирование – процесс представления информации в виде кода или, другими словами, процесс преобразования информации из одной формы в другую. Декодирование – процесс, обратный кодированию.

Человек различает знаки по их начертанию, а компьютер – по их двоичным кодам. При вводе в компьютер текстовой информации происходит ее двоичное кодирование, изображение знака преобразуется в его двоичный код. Пользователь нажимает на клавиатуре клавишу со знаком, и в компьютер поступает определенная последовательность электрических импульсов (двоичный код знака).

В процессе вывода знака на экран компьютера производится обратное перекодирование, т.е. преобразование двоичного кода знака в его изображение.

Присваивание знаку конкретного двоичного кода – это вопрос соглашения, которое фиксируется в кодовой таблице (или кодовой странице code page). В кодовой таблице ASCII первые 32 кода (де-

сятые коды с 0 по 31) соответствуют не знакам, а операциям (перевод строки, прокрутка страницы принтером и т.д.). Десятичным кодом 32 закодирован символ пробела. Пробел – это символ, а не его отсутствие.

Таблица ASCII (American standard code for information interchange) была разработана в США в 1963 году. Название «ASCII» по-русски часто произносится как «аски». Для кодирования каждого знака таблицы ASCII требуется количество информации, равное 8 бит, т. е. длина двоичного кода каждого знака составляет восемь двоичных знаков, т.е. один байт. Каждому знаку ставится в соответствие уникальный двоичный код из интервала от 00000000 до 11111111 (в десятичном представлении от 0 до 255).

Таблица ASCII определяет коды для символов:

- Десятичных цифр;
- Латинского алфавита (большие и малые буквы);
- Национального алфавита (например, русского);
- Знаков препинания;
- Специальных символов.

Десятичные коды с 32 по 127 являются интернациональными и соответствуют буквам латинского алфавита (большим и малым), цифрам, скобкам, специальным символам (например, @), знакам арифметических операций и знакам препинания.

Десятичные коды с 128 по 255 являются национальными, т. е. в различных национальных кодировках одному и тому же коду соответствуют разные знаки.

До недавнего времени существовало несколько различных кодовых таблиц для русских букв (Windows CP1251, MS-DOS CP866, КОИ-8, Mac, ISO), поэтому тексты, созданные в одной кодировке, не могли правильно отображаться в другой. Например, в кодировке Windows (кодовая страница CP1251) последовательность числовых кодов 221, 194, 204 образует слово «ЭВМ», тогда как в других кодировках это будет бессмысленный набор символов.

В большинстве случаев пользователь не должен заботиться о перекодировках текстовых документов, так как это делают специаль-

ные программы-конверторы, встроенные в операционную систему и приложения.

В последние годы широкое распространение получил новый международный стандарт кодирования текстовых символов Unicode, который отводит на каждый символ 2 байта (16 бит). Вычислим количество символов, которые можно закодировать согласно этому стандарту:

$$N = 2^{16} = 65536$$

Такого количества символов оказалось достаточно, чтобы закодировать не только русский и латинский алфавиты, цифры, знаки и математические символы, но и греческий, арабский, иврит и другие алфавиты.

В конце параграфа приведем пример решения задачи на кодирование текстовой информации.

Пример. Текстовый документ, состоящий из 3072 символов, хранился в 8-битной кодировке КОИ-8. Этот документ был преобразован в 16-битную кодировку Unicode. Укажите, какое дополнительное количество Кбайт потребуется для хранения документа. В ответе запишите только число.

Решение. Первоначально для кодирования текста была использована кодировка, при которой каждый символ занимает объём памяти, равный 8 бит или 1 байт. Вычислим объём информации в кодировке КОИ-8:

$$3072 * 1 \text{ байт} = 3072 \text{ байта.}$$

Вычислим объём информации в 16-битной кодировке (один символ занимает память, равную двум байтам):

$$3072 * 2 \text{ байта} = 6144 \text{ байта.}$$

Вычислим дополнительный объём памяти, который потребуется для хранения перекодированного файла:

$$6144 - 3072 = 3072 \text{ байта.}$$

Переведем данную величину в килобайты:

$$3072 / 1024 = 3 \text{ Кбайта.}$$

Ответ: 3.

Контрольные вопросы

1. Что такое текстовая информация?
2. Что означает термин «кодирование»?
3. Зачем нужно двоичное кодирование текстовой информации?
4. Что такое кодовая таблица?
5. Расскажите о кодировке ASCII.
6. С какой целью ввели стандарт кодирования Unicode?

§2.2. Кодирование звуковой информации

Как известно из курса физики, звук представляет собой распространяющуюся в воздухе, воде или другой среде волну с непрерывно меняющейся амплитудой и частотой. Человек воспринимает звуковые волны (колебания воздуха) с помощью слуха в форме звука различных громкости и тона. Чем больше амплитуда звуковой волны, тем громче звук, чем больше частота волны, тем выше тон звука. Человеческое ухо воспринимает звук с частотой от 20 колебаний в секунду (низкий звук) до 20000 колебаний в секунду (высокий звук).

Первичными преобразователями звукового сигнала являются датчики (микрофоны), которые преобразуют звук в аналоговый электрический сигнал. Далее этот сигнал дискретизируется по времени и амплитуде (оцифровывается) специальными устройствами – аналого-цифровыми преобразователями (АЦП), и в виде последовательности двоичных знаков поступает в компьютер. Обратное преобразование осуществляется цифро-аналоговыми преобразователями (ЦАП), выходной сигнал которых сглаживается, усиливается и подается на динамики, воспроизводящие звуковую информацию. Как правило, подобным образом в компьютер поступает информация от любых датчиков, преобразующих самую разнообразную техническую информацию.

Для того чтобы компьютер мог обрабатывать звук, непрерывный (аналоговый) звуковой сигнал должен быть преобразован в цифровую дискретную форму с помощью процесса дискретизации.

Напомним, что преобразование непрерывного объекта в множество отделимых друг от друга частей (т.е. в дискретное множество) называется дискретизацией.

Непрерывная звуковая волна разбивается на отдельные маленькие временные (временн'ые ударение на «ы») участки, для каждого такого участка устанавливается определенная величина амплитуды (интенсивности) звука. Таким образом, непрерывная зависимость громкости звука от времени $A(t)$ заменяется на дискретную последовательность уровней громкости. На графике это выглядит как замена гладкой кривой на последовательность «ступенек».

Качество полученного цифрового звука зависит от количества измерений уровня громкости звука в одну секунду, т. е. частоты дискретизации. Чем большее количество измерений производится за одну секунду (чем больше частота дискретизации), тем точнее «лесенка» цифрового звукового сигнала повторяет кривую аналогового сигнала.

Частота дискретизации звука – это количество измерений громкости звука за одну секунду.

Обычно в компьютерных программах обработки звуковой информации частота дискретизации лежит в диапазоне от 8000 до 48000 измерений громкости звука за одну секунду.

Каждой «ступеньке» присваивается определенное значение уровня громкости звука. Уровни громкости звука можно рассматривать как набор возможных состояний N , для кодирования которых необходимо определенное количество информации I , которое называется глубиной кодирования звука. Таким образом, глубина кодирования звука – это количество информации, которое необходимо для кодирования дискретных уровней громкости цифрового звука.

Если известна глубина кодирования, то количество уровней громкости цифрового звука можно рассчитать по формуле:

$$N = 2^I$$

Пусть глубина кодирования звука составляет 16 бит, тогда количество уровней громкости звука равно:

$$N = 2^I = 2^{16} = 65536$$

В процессе кодирования каждому уровню громкости звука присваивается свой 16-битовый двоичный код, наименьшему уровню звука будет соответствовать код 0000000000000000, а наибольшему – 1111111111111111.

Чем больше частота и глубина дискретизации звука, тем более качественным будет звучание оцифрованного звука. Самое низкое качество оцифрованного звука, соответствующее качеству телефонной связи, получается при частоте дискретизации 8000 раз в секунду, глубине дискретизации 8 бит и записи одной звуковой дорожки (режим «моно»). Самое высокое качество оцифрованного звука, соответствующее качеству аудио-CD, достигается при частоте дискретизации 48000 раз в секунду, глубине дискретизации 16 бит и записи двух звуковых дорожек (режим «стерео»).

Заметим, что чем выше качество цифрового звука, тем больше информационный объем звукового файла. Можно оценить информационный объем цифрового стереофонического звукового файла длительностью звучания 1 секунда при среднем качестве звука (16 бит, 24000 измерений в секунду). Для этого глубину кодирования необходимо умножить на количество измерений в одну секунду и умножить на 2 (стереозвук):

$$16 * 24000 * 2 = 768000 \text{ бит} = 96000 \text{ байт} = 93,75 \text{ Кбайт}$$

Звуковые редакторы позволяют не только записывать и воспроизводить звук, но и редактировать его. Оцифрованный звук представляется в звуковых редакторах в наглядной форме, поэтому операции копирования, перемещения и удаления частей звуковой дорожки можно легко осуществлять с помощью соответствующих клавиатурных команд, контролируя процесс с помощью программы невидимого доступа к информации (JAWS for Windows или NVDA). Кроме того, можно накладывать звуковые дорожки друг на друга (микшировать звуки) и применять различные акустические эффекты (эхо, воспроизведение в обратном направлении и др.).

Звуковые редакторы позволяют изменять качество цифрового звука и объем звукового файла путем изменения частоты дискретизации и глубины кодирования. Оцифрованный звук можно сохранять без сжатия в звуковых файлах в универсальном формате WAV или

в формате со сжатием mp3. При сохранении звука в форматах со сжатием отбрасываются «избыточные» для человеческого восприятия звуковые частоты с малой интенсивностью, совпадающие по времени со звуковыми частотами с большой интенсивностью. Применение такого формата позволяет сжимать звуковые файлы в десятки раз, однако приводит к необратимой потере информации, т.е. файлы не могут быть восстановлены в первоначальном виде.

Для кодирования инструментальной музыки в компьютерах и синтезаторах часто используют систему кодирования MIDI, основанную на нотной записи и отличающуюся чрезвычайной компактностью и простотой изменения темпа и тональности мелодии. Такое кодирование звуковой информации называют табличным.

В конце параграфа приведем пример решения задач на кодирование звуковой информации.

пример. Производилась двухканальная (стерео) звукозапись с частотой дискретизации 64 кГц и 24-битным разрешением (глубина кодирования). В результате был получен файл размером 48 Мбайт, сжатие данных не производилось. Определите приблизительно, сколько времени (в минутах) проводилась запись. В качестве ответа укажите ближайшее к времени записи целое число.

решение. Так как частота дискретизации 64 кГц, то за одну секунду запоминается 64000 значений сигнала. Глубина кодирования 24 бита, т. е. 3 байта, это объем памяти, который требуется для хранения данных одного замера сигнала. Объем памяти для хранения данных двух каналов в два раза больше памяти, которая требуется при одноканальной записи.

Для вычисления объема файла в байтах необходимо перемножить количество каналов на глубину звука в байтах на частоту дискретизации и на длительность звучания файла в секундах.

Для нахождения времени разделим объем файла на известные параметры звукового файла, переписанные для удобства в виде степени чисел 2, 3 и 5:

Частота дискретизации:

$$2^6 * (2 * 5)^3 = 2^9 * 5^3 \text{ байт}$$

Объем файла:

$$48 \text{ Мбайт} = 48 * 2^{20} \text{ байт} = 2^{24} * 3 \text{ байт}$$

Количество каналов:

$$2^1$$

$$t = (3 * 2^{24}) / (2^9 * 5^3 * 3 * 2^1) = (2^{24} / 2^{10}) * (1 / 5^3) = 2^{14} / 5^3 \text{ с}$$

Для выяснения количества минут, разделим эту величину на 60. заметим, что

$$60 = 2^2 * 3 * 5$$

$$t = 2^{14} / (2^2 * 3 * 5 * 5^3) = 2^{12} / (3 * 5^4) = 4096 / 1875 \text{ мин.}$$

Эта величина больше числа 2 и меньше 2,5. Округляя до целых, получаем результат 2 минуты.

Ответ: 2.

Контрольные вопросы

1. Что такое дискретизация?
2. Что такое частота дискретизации?
3. Что такое глубина кодирования?
4. Как частота дискретизации и глубина кодирования влияют на качество цифрового звука?

§2.3. Кодирование графической информации

Аналогично текстовой и звуковой информации графическая информация может быть представлена в аналоговой и цифровой форме. Примером аналогового представления графической информации может служить живописное полотно, цвет которого изменяется непрерывно, вид природных объектов и др. Цифровое изображение можно видеть на экране монитора, на листе бумаги с отпечатанной на принтере картинкой и др. Цифровое изображение состоит из отдельных точек разного цвета, аналогично рельефно-графическому изображению, отпечатанному на брайлевском принтере.

Для обработки на компьютере или с помощью какой-либо другой цифровой техники графические изображения необходимо закодировать – преобразовать из аналоговой (непрерывной) в цифровую (дискретную) форму. Кодирование графических изображений разделяется на два направления – растровая и векторная графика.

Растровые изображения получают путем пространственной дискретизации. Пространственную дискретизацию изображения можно

сравнить с построением изображения из мозаики (большого количества маленьких разноцветных стекол). Изображение разбивается на отдельные маленькие элементы – точки или пиксели (pixel – picture element), причем каждый элемент может иметь свой цвет. Чем больше пикселей, тем более детально представлено изображение.

Для каждого пикселя в памяти компьютера хранится код цвета. Наиболее распространенной является RGB-кодировка (Red, Green, Blue). При объеме памяти 24 бита для хранения цвета одного пикселя 8 бит используется для задания интенсивности красного цвета, 8 бит – зеленого и еще 8 бит – синего цвета. Таким образом, каждый цвет имеет 256 уровней интенсивности. Смешивание этих цветов в различных соотношениях дает 2^{24} различных цветов.

В результате пространственной дискретизации графическая информация представляется в виде растрового изображения, которое формируется из определенного количества строк, содержащих, в свою очередь, определенное количество точек (пикселей). В таком виде графическая информация хранится в файлах с расширением BMP. Для уменьшения размеров хранимых файлов применяют дополнительные методы сжатия графической информации. Получающиеся при этом файлы имеют расширение JPEG, GIF и т.п.

Важнейшей характеристикой качества растрового изображения является разрешающая способность. Разрешающая способность растрового изображения определяется количеством пикселей как по горизонтали, так и по вертикали на единицу длины изображения. Чем меньше размер пикселя (точки), тем больше разрешающая способность (больше строк раstra и точек в строке) и, соответственно, выше качество изображения. Величина разрешающей способности обычно выражается в dpi (dot per inch - точек на дюйм), т. е. в количестве точек в полоске изображения длиной один дюйм (1 дюйм = 2,54 см).

В процессе дискретизации используется не только RGB-кодировка, но и другие палитры цветов, т. е. наборы цветов, в которые могут быть окрашены точки изображения. Каждый цвет можно рассматривать как возможное состояние точки. Количество цветов N в палитре и количество информации I , необходимое для

кодирования цвета каждой точки, связаны между собой и могут быть вычислены по формуле:

$$N = 2^I$$

В простейшем случае (черно-белое изображение без градаций серого цвета) палитра цветов состоит всего из двух цветов (черного и белого). Каждая точка (пиксель) экрана может принимать одно из двух состояний – «черная» или «белая», следовательно, по приведенной формуле можно вычислить, какое количество информации необходимо, чтобы закодировать цвет каждой точки:

$$2 = 2^1 = 2^1$$

Следовательно $I = 1$ бит.

Количество информации, которое используется для кодирования цвета пикселя изображения, называется глубиной цвета. Так, в рассмотренном выше RGB-кодировании глубина цвета 24 бита. Зная глубину цвета, по выше приведенной формуле можно вычислить количество цветов в палитре.

При векторном представлении графической информации в компьютере вместо раstra (сетки) пикселей используется заданный набор графических примитивов – отрезок, кривая линия, окружность, многоугольник и др. Каждый графический примитив описывается с помощью специального языка математических уравнений.

Например, для представления окружности необходимо задать следующие параметры:

- Координаты центра окружности;
- Значение радиуса окружности;
- Цвет заполнения окружности;
- Цвет самой окружности;
- Толщина линии окружности.

Сложные графические объекты представляются в виде совокупности графических примитивов.

Основное отличие векторной графики от растровой состоит в том, что её можно легко увеличивать или уменьшать (масштабировать) без потери качества, поскольку векторное изображение задано в виде уравнений, а растровое изображение задано с определенным коли-

чеством пикселей и его увеличение приводит к возрастанию зернистости и потере качества.

Однако, не каждый объект может быть представлен в векторной форме. Для сложных объектов может понадобиться слишком большое количество графических примитивов и очень большое время для расчета изображения при выводе на растровый дисплей. В частности, перевод растровых изображений в векторную форму, как правило, требует очень большого объема вычислений и не всегда обеспечивает высокое качество изображения.

Таким образом, векторная графика эффективнее для представления чертежей, схем, геометрических рисунков и других простых изображений, не нуждающихся в фотореализме.

В конце параграфа приведем пример решения задач на кодирование графической информации.

Пример. Какой минимальный объём памяти (в Кбайт) нужно зарезервировать, чтобы можно было сохранить любое растровое изображение размером 128X128 пикселей при условии, что в изображении могут использоваться 256 различных цветов? В ответе запишите только целое число, единицу измерения писать не нужно.

решение. Информация о каждом пикселе содержит двоичный код, определяющий цвет пикселя. Для возможности сохранять информацию о различных 256 цветов, потребуется найти наименьшую степень числа 2, чтобы выполнялось неравенство:

$$2^x \leq 256$$

Такой степенью является число 8. Следовательно, для каждого пикселя выделяется объем памяти, равный 8 бит = 1 байт.

Подсчитаем количество пикселей:

$$128 * 128 = 2^7 * 2^7 = 2^{14} \text{ пикселей}$$

Перемножим количество пикселей на объем памяти, занимаемой информацией об одном пикселе:

$$2^{14} * 1 = 2^{14} \text{ байт}$$

Переведем объем памяти в килобайты, разделив число байт на 2^{10} :

$$2^{14} / 2^{10} = 2^4 \text{ Кбайт} = 16 \text{ Кбайт}$$

Ответ: 16.

Контрольные вопросы

1. Какие виды кодирования графической информации вы знаете?
2. Объясните, как с помощью пространственной дискретизации происходит формирование растрового изображения.
3. В каких файлах хранится растровое изображение?
4. В каких единицах выражается разрешающая способность растровых изображений?
5. Как связаны между собой количество цветов в палитре и глубина цвета?
6. Как кодируется графическая информация в векторном представлении?
7. Что такое графические примитивы?
8. В чём разница между векторной и растровой графикой?
9. В каких случаях векторная графика предпочтительнее, чем растровая?

§2.4. Коды с самоисправлениями

Как вы уже хорошо знаете, в компьютере для представления информации используется двоичное кодирование, поскольку технически стало возможным создавать устройства, которые могут с высокой надёжностью сохранять и распознавать не более двух различных состояний (цифр), например:

- Электромагнитные реле (замкнуто/разомкнуто), широко использовались в конструкциях первых ЭВМ;
- Участок поверхности магнитного носителя информации (намагничен/размагничен);
- Участок поверхности лазерного диска (отражает/не отражает);
- Триггер, может устойчиво находиться в одном из двух состояний, широко используется в оперативной памяти компьютера.

Все виды информации в компьютере кодируются на машинном языке, в виде последовательностей нулей и единиц. Алфавит такого кодирования состоит из двух символов 0 и 1.

В теории кодирования алфавитом называется набор символов, используемых для записи кода.

Цифры двоичного кода можно рассматривать как два равновероятных состояния (события). При записи двоичной цифры реализуется выбор одного из двух возможных состояний и, следовательно, она несет количество информации, равное 1 биту. Таким образом, две цифры несут информацию в 2 бита, три цифры – в 3 бита и так далее.

Количество информации в битах равно количеству цифр двоичного машинного кода. Подробнее о подходах к измерению количества информации говорилось в параграфе 1.5.

Двоичная система кодирования сообщений применяется также и при передаче по каналам связи. Это значит, что каждому символу, используемому для обычной записи сообщения (например, букве русского языка), сопоставлена последовательность из 0 и 1. Именно такое сопоставление представлено, например, кодовой таблицей ASCII. Более подробно о таблицах кодирования символов было рассказано в параграфе 2.1.

В кодовой таблице CP1251 символы «а», «и», «п», «р» задаются кодами 11100000, 11101000, 11101111, 11110000 соответственно. Таким образом, слово «пир» будет закодировано последовательностью 111011111110100011110000.

Предположим, что при передаче этого кодового сообщения произошла ошибка и вместо одной из единиц в результате каких-либо помех был передан 0. Т.е. на приемник информации поступила последовательность 111011111110000011110000. Тогда эта последовательность будет декодирована как «пар» (проверьте это!).

К подобной ошибке мог привести обыкновенный технический сбой, например, перепад в напряжении. Воздействие, приводящее к искажению передаваемой информации, обычно называют шумом.

Если сообщение носит «бытовой» характер, подобная ошибка может и не привести к тяжелым последствиям. Однако, если речь идет о передаче команд управления космическим кораблем или атомной электростанцией, то последствия могут быть весьма серьезными.

Существует метод, позволяющий исправлять ошибки, возникающие при передаче информации. Чтобы продемонстрировать идею кода, позволяющего обнаруживать ошибки, предположим, что все

передаваемые сообщения – это числовые данные, записываемые цифрами обычной десятичной системы счисления. Каждую цифру будем кодировать ее представлением в двоичной системе счисления. Результат такого кодирования представлен в двух первых столбцах следующей таблицы (на третий столбец пока не обращаем внимание):

0	0000	00000
1	0001	00011
2	0010	00101
3	0011	00110
4	0100	01001
5	0101	01010
6	0110	01100
7	0111	01111
8	1000	10001
9	1001	10010

Очевидно, что предложенное кодирование не позволяет обнаружить ошибку. Например, если вместо 0010 пришло ошибочное 0011, то в такое сообщение можно поверить как в правильное.

Теперь добавим в конце кода каждой цифры еще один двоичный символ. Получившийся таким образом расширенный код представлен в третьем столбце таблицы.

Последний символ в код приписывается по следующему правилу: если в исходном четырехсимвольном коде четное число единиц, то пишем 0, если нечетное, то пишем 1. В получившемся коде для любого символа количество единиц всегда четно. Поэтому, если при передаче сообщения в каком-то месте произошла ошибка, т.е. 0 заменился на 1 или наоборот, то количество единиц в таком коде соответствующего символа стало нечетным, и это легко обнаруживается. Например, пришло сообщение: 100100011010011. Разбиваем его на три группы по 5 символов: 10010 00110 10011. Сразу видно: первые две группы правильные, а третья – нет.

Чтобы описать способность кода к распознаванию ошибок, используют понятие расстояния между словами. Пусть даны слова над одним алфавитом. Тогда расстоянием между словами называется ко-

личество позиций, в которых символы одного слова не совпадают с символами второго. Например, расстояние между словами «стог» и «снег» равно 2 – они отличаются во второй и третьей позициях. А между словами 1001001 и 0100001 расстояние равно 3, поскольку они отличаются в первой, второй и четвертой позициях. Расстояние между словами называют расстоянием Хэмминга.

Обычно под расстоянием понимают некоторую геометрическую величину, характеризующую удаленность объектов друг от друга. Однако, на практике расстояние представляют различными величинами. На плоскости под расстоянием обычно понимают длину отрезка, соединяющего две точки. На поверхности земного шара, естественно расстоянием считать дугу большого круга, проходящего через две рассматриваемые точки. В городе расстояние между двумя точками измеряют вдоль улиц, по которым можно добраться из одной точки в другую и т.д.

Математики выяснили, что общими для всех разновидностей расстояний являются три свойства (три аксиомы метрики). Обозначим через $D(A, B)$ функцию, которая двум объектам (например, точкам или словам) сопоставляет неотрицательное число. Эта функция должна удовлетворять следующим аксиомам:

1. $D(A, B) = 0$ тогда и только тогда, когда $A = B$ (равенство здесь означает совпадение объектов);
2. $D(A, B) = D(B, A)$;
3. $D(A, B) \leq D(A, C) + D(C, B)$, каким бы ни был объект C .

Смысл первого свойства очевиден. Второе свойство утверждает, что объект A удален от объекта B так же, как объект B удален от объекта A . наконец, третье свойство говорит, что дорога через третий объект C всегда не короче, нежели прямой путь. Третье свойство обычно называют неравенством треугольника за его естественную геометрическую аналогию: сумма двух сторон треугольника больше третьей стороны.

В математике принято любую функцию, обладающую указанными тремя свойствами, называть расстоянием (или метрикой). Расстояние Хэмминга обладает всеми тремя свойствами. Попробуйте доказать это!

Теперь опять увеличим рассматриваемый выше код, добавив еще две двоичных цифры в конце. Приведем новую таблицу кодов, в которой два первых столбца те же, а в третьем приведен семибитовый расширенный код:

0	0000	0000000
1	0001	0001111
2	0010	0010110
3	0011	0011001
4	0100	0100101
5	0101	0101010
6	0110	0110011
7	0111	0111100
8	1000	1000011
9	1001	1001100

При таком кодировании минимальное расстояние между кодовыми словами равно 3. Это означает, что если при передаче сообщения произошло две ошибки, то все равно принятое слово не совпадет ни с одним кодовым словом. Таким образом, будет выявлено ошибочно переданное слово. Более того, весьма маловероятно, чтобы в семибитовом слове оказалось сразу две ошибки, поэтому, получив слово с ошибкой, можно найти ближайшее к нему слово (т.е. отличающееся только на один символ) и исправить ошибку. Заметим, что поскольку расстояние между любыми двумя кодовыми словами не меньше 3, кодовое слово, ближайшее к ошибочному, будет единственным.

Пусть, например, получено сообщение 001011101100101010111. Разобьем его на группы по 7 символов, получим: 0010111, 0110010, 1010111. Во второй таблице нет кодового слова, соответствующего первой группе символов. Но на расстоянии 1 от него находится код 0010101, значит, допущена одна ошибка, а исходно была передана цифра 2. Вторая группа является кодовым словом, она соответствует цифре 6. А третья группа снова ошибочна. Ближайшее к ней кодовое слово – 1000111. Это код цифры 8. Значит, было передано число 268. Проверьте это по таблице!

Таким образом, построенный код гарантированно исправляет одну ошибку в закодированном слове. Причем, исправление производит-

ся по четкому алгоритму, что позволяет автоматизировать процесс исправления ошибок. Такой код получил название кода Хэмминга.

Разработанная математиками теория кодирования позволяет строить коды с заданным минимальным расстоянием между кодовыми словами. Так, в европейских системах связи широко используется 235-битовый код, расширенный с помощью дополнительных 20 двоичных символов. Минимальное расстояние между словами этого кода равно 7. Такой код гарантированно обнаруживает 6 ошибок и исправляет слова, в которых допущено не более 3 ошибок. В течение многих лет эксплуатации этих систем не было случая, чтобы ошибка прошла незамеченной.

В конце параграфа приведем два примера решения задач на кодирование информации и на перевод из одной системы счисления в другую.

Пример 1. Для кодирования букв О, В, Д, П, А решили использовать двоичное представление чисел 0, 1, 2, 3 и 4 соответственно (в записи используется минимум 2 разряда с сохранением одного незначащего нуля в случае одноразрядного представления). Закодируйте слово «ВОДОПАД» таким способом и результат запишите в восьмеричном представлении.

Решение. Сначала следует представить числа от 0 до 4 в двоичном коде:

$$\begin{aligned}0 &= 0_2; \\1 &= 1_2; \\2 &= 10_2; \\3 &= 11_2; \\4 &= 100_2.\end{aligned}$$

Выпишем двоичный код каждой буквы, согласно условию задачи:

$$\begin{aligned}\text{О} &= 00; \\ \text{В} &= 01; \\ \text{Д} &= 10; \\ \text{П} &= 11; \\ \text{А} &= 100.\end{aligned}$$

Кодируем последовательность букв «ВОДОПАД» и получаем код: 010010001110010.

Теперь разобьём этот код на тройки справа налево и переведём полученные группы в десятичный код, зная, что восьмеричное представление совпадает с десятичным при разбиении тройками.

$$010\ 010\ 001\ 110\ 010 = 2_{10}\ 2_{10}\ 1_{10}\ 6_{10}\ 2_{10} = 2_8\ 2_8\ 1_8\ 6_8\ 2_8$$

Собирая восьмеричные знаки вместе получаем результат: 22162_8 .

Ответ: 22162_8 .

Пример 2. Для передачи по каналу связи сообщения, состоящего только из символов А, Б, В и Г, используется посимвольное кодирование: А – 10, Б – 11, В – 110, Г – 0. Через канал связи передаётся сообщение: «ВАГБААГВ». Закодируйте сообщение данным кодом. Полученное двоичное число переведите в шестнадцатеричный вид.

Решение. Закодируем последовательность букв ВАГБААГВ и получим:

$$1101001110100110.$$

Теперь разобьём это представление на четвёрки справа налево и переведём полученный набор чисел сначала в десятичный код, затем в шестнадцатеричный:

$$1101\ 0011\ 1010\ 0110 = 13\ 3\ 10\ 6 = D3A6$$

Ответ: $D3A6_{16}$.

Контрольные вопросы

1. Какие способы технической реализации хранения информации в компьютере вы знаете?

2. Что такое шум с точки зрения процесса передачи информации?

3. Что называют алфавитом в теории кодирования?

4. Приведите примеры процессов передачи информации, в которых:

А) Ошибка не приведет к серьезным последствиям;

Б) Ошибка приведет к серьезным последствиям.

5. Что называется метрикой (расстоянием)?

6. Как можно вычислить расстояние между словами?

7. Как можно исправить ошибочное слово используя расстояние между словами в коде Хэмминга?

§2.5. Минимальные коды

Как известно, шесть точек брайлевского шеститочия позволяют образовать 63 комбинации. Эти комбинации дают возможность записать (закодировать) буквы русского языка, знаки препинания и некоторые служебные символы. Использование особых признаков и специальных знаков рельефно-точечной системы Брайля позволяет записывать не только буквы различных естественных алфавитов, но и математические, химические и другие знаки, а также и знаки нотного письма.

Записывая номера точек, образующих буквы русского алфавита, можно закодировать любое текстовое сообщение. Например, слово «информатика» будет закодировано следующей последовательностью цифр:

24 1345 124 135 1235 134 1 2345 24 13 1 (проверьте это).

Брайлевские дисплеи и принтеры могут отображать информацию в компьютерном восьмиточечном брайле. Используя 8 точек можно получить уже 255 различных комбинаций. В восьмиточечие седьмая точка располагается под третьей, а восьмая под шестой. Брайлевские принтеры используют систему кодирования, в которой каждый бит байта отвечает за одну конкретную точку. Т.е., если на месте некоторого бита стоит 1, то принтер пробивает соответствующую точку, а если 0, то точка не пробивается. Очевидно, что такой способ хранения «брайлевской» информации в компьютере значительно более эффективен, чем код, основанный на номерах точек.

Из истории тифлопедагогики известны неоднократные попытки оптимизации системы Брайля. Желание сделать это было обусловлено с одной стороны, потребностью создать интернациональную систему письма для слепых на основе системы Брайля. С другой стороны, предпринимались попытки «минимизировать код», т.е. изменить систему Брайля так, чтобы буквы, чаще всего встречающиеся в текстах, обозначались наименьшим количеством точек. Предполагалось, что, поскольку, буквы, чаще всего встречающиеся, обозначаются минимальным количеством точек, то затраты времени и физических усилий при письме будут минимальны.

На основе современных алгоритмов можно разработать наиболее экономичный код для записи по рельефно-точечной системе. Самым экономичным будет код, в котором каждый элементарный символ будет передавать максимальную информацию. Элементарный символ в случае системы Брайля это одна точка шеститочия (т.е. один двоичный символ).

Задачу минимизации можно решить, например, с помощью «кода Шеннона-Фано». Принцип построения его в том, что кодируемые символы (буквы или комбинации букв) разделяются на две приблизительно равновероятные группы: для первой группы символов на первом месте комбинации ставится 0, для второй 1. далее каждая группа снова делится на две приблизительно равновероятные подгруппы. Для символов первой подгруппы на втором месте ставиться 0; для второй подгруппы – единица и т. д.

Алгоритм Шеннона-Фано – один из первых алгоритмов сжатия, который сформулировали американские учёные Шеннон и Фано. Данный метод сжатия имеет большое сходство с алгоритмом Хаффмана, который появился на несколько лет позже и является логическим продолжением алгоритма Шеннона. Алгоритм использует коды переменной длины: часто встречающийся символ кодируется кодом меньшей длины, редко встречающийся – кодом большей длины.

Коды Шеннона – Фано являются префиксными, т.е. никакое кодовое слово не является началом какого-либо другого. Это свойство называется условием Фано и его соблюдение позволяет однозначно декодировать любую последовательность кодовых слов. В теории кодирования условие Фано формулируется следующим образом:

Для того, чтобы сообщение, записанное с помощью неравномерного по длине кода, однозначно декодировалось, достаточно, чтобы никакой код не был началом другого (более длинного) кода.

Подробнее алгоритмы построения минимальных кодов здесь рассматриваться не будут. Вы можете изучить их самостоятельно используя источники информации из сети Интернет.

Контрольные вопросы

1. Как кодируется информация в брайлевском принтере?

2. В чём состоит алгоритм кодирования Шеннона-Фано?
3. Какие коды называются префиксными?
4. В чём состоит условие Фано?
5. Является ли код, состоящий в кодировании букв номерами точек их брайлевского написания префиксным?
6. Как вы думаете, почему система рельефно-точечных обозначений Брайля так и не была минимизирована?

Глава 3

Алгебра логики

§3.1. Логические функции

Алгебра логики – это раздел математики, изучающий высказывания, рассматриваемые с точки зрения их логических значений (истинности или ложности), а также логические операции над ними. В алгебре логики предполагается, что высказывания могут быть только истинными или ложными, т.е. используется бинарная (двоичная) логика. Каждая переменная может иметь только два значения – 0 или 1 («истина» или «ложь»).

Алгебра логики возникла в середине XIX века в трудах английского математика Джорджа Буля. Ее создание представляло собой попытку решать традиционные логические задачи алгебраическими методами. Буль положил в основу своей идеи аналогию между алгеброй и логикой. Алгебра логики стала первой системой математической логики, в которой алгебраическая символика была применена к логическим выражениям и операциям. Буль создал систему, в которой стало возможно решать логические задачи с помощью алгебраических методов. Любое логическое утверждение он выражал форме уравнений, в которых действуют логические законы, подобные законам классической алгебры.

В 1938 году Клод Шеннон применил алгебру логики для описания процесса функционирования релейно-контактных и электронно-ламповых схем. Логика высказываний послужила основным математическим инструментом при создании компьютеров. Она легко преобразуется в битовую логику – истинность высказывания обозначается одним битом (0 – ЛОЖЬ, 1 – ИСТИНА), а это легко реализуется технически.

Алгебра логики занимается исследованием операций с высказываниями, т.е. с предложениями, которые характеризуются только истинностным значением. Другими словами, в алгебре логики высказывание может иметь только одно из двух значений: «истина» или «ложь».

Логическое высказывание – это некоторое утверждение, в отношении которого можно однозначно сказать, истинно оно или ложно. Например, высказывание «Джордж Буль создал алгебру логики» является истинным высказыванием; утверждение «В сутках на земле 25 часов» будет ложным высказыванием, а фраза «прямоугольный треугольник» высказыванием не является, поскольку не указывает ни на какой конкретный треугольник и не может быть истинным или ложным.

Высказывания – это базовые понятия алгебры логики, которыми она оперирует. Высказывания обозначают большими латинскими буквами (переменными), например, А, В, С и т.д. Возможными значениями логической переменной могут быть только 1 (истина) или 0 (ложь). Высказывания строятся из логических переменных и логических констант 1 и 0 при помощи логических операций.

Как правило, в математических выражениях для краткости записи используются обозначения 0 и 1. В дальнейшем изложении материала мы будем использовать именно 0 и 1, подразумевая их логическое значение.

В алгебре логики определяются три основные базовые операции: отрицание (не), конъюнкция (и) и дизъюнкция (или). Смысл этих операций совпадает с их общеупотребительным значением.

Отрицание (инверсия, логическое «не») является унарной операцией, т.е. имеет один аргумент. Отрицание ставит в соответствие высказыванию новое высказывание, значение которого противоположно исходному. Например, отрицание высказывания «два умножить на два равно пять» будет истинное высказывание «два умножить на два не равно пять».

По брайлю знак отрицания обозначается точками 146. Во многих языках программирования отрицание обозначается английским словом not. Примером символической записи может служить следующая формула:

$$\text{not } A = B$$

В дальнейшем изложении мы будем использовать подобные обозначения. Заметим, что в языке программирования Python, который вы изучаете, используются именно такие обозначения.

Конъюнкция (логическое умножение, логическое «и») является бинарной операцией, т.е. объединяет два аргумента. Конъюнкция двух высказываний истинна тогда и только тогда, когда истинны оба исходных высказывания. Например, конъюнкция высказываний « $5 < 7$ » и «луна вращается вокруг земли» истинна, а конъюнкция высказываний « $2 = 3$ » и «в неделе 7 дней» – ложна.

По брайлю конъюнкция обозначается двухклеточным знаком 56 26. В этой книге будем обозначать конъюнкцию английским словом `and`, как это принято в языке Python. Например:

`A and B =not C and D`

Заметим, что конъюнкцию часто обозначают как умножение. Действительно, если опираться на числовые обозначения лжи и истины (0 и 1), то конъюнкция в точности соответствует умножению. Проверьте это самостоятельно!

Дизъюнкция (логическое сложение, логическое «или») также является бинарной (от двух аргументов) операцией. Дизъюнкция двух высказываний ложна тогда и только тогда, когда ложны оба исходных высказывания. Например, дизъюнкция высказываний «2 четное число» или «3 четное число» истинна, а дизъюнкция высказываний «4 нечетное число» или «5 четное число» ложна.

По брайлю дизъюнкция обозначается двухклеточным знаком 56 35. В этой книге в стиле предыдущих обозначений будем использовать для обозначения дизъюнкции английское слово `or`. Например:

`A or B and not C`

Понять, каким образом дизъюнкция играет роль сложения несколько сложнее. Дело в том, что в алгебре логики мы будем считать любое положительное число соответствующим истине и представлять его единицей. Таким образом, $1 + 1 = 1$. Проверьте соответствие сложения с описанным изменением и дизъюнкции самостоятельно!

С помощью описанных выше трёх базисных функций алгебры логики можно выразить любую другую функцию. Доказательство этого утверждения выходит за рамки этой книги.

В школьном курсе информатики используют еще две бинарные логические операции (функции) – импликацию и эквиваленцию. Ка-

ждуя из этих функций можно выразить через три описанных выше базисных операции.

Импликация (следовательно) также представляет собой бинарную операцию. Это логическая операция, ставящая в соответствие двум высказываниям новое, являющееся ложным тогда и только тогда, когда первое высказывание (посылка) истинно, а второе (следствие) – ложно. Таким образом, импликация ложна только в одном случае – из истины ложь не следует. например, высказывание «Если идёт дождь, то на улице сыро» представляет собой истинную импликацию, а высказывание «если шесть положительное число, то шесть простое число» – ложное.

По брайлю импликация обозначается двухклеточным знаком 2356 345 (знак равенства и круглая скобка закрыта). Это изображение соответствует двойной стрелке вправо и в дальнейшем мы будем обозначать импликацию $=$). Например:

$$A =) B = \text{not } A \text{ or } B$$

Эквиваленция (тогда и только тогда, когда) – бинарная логическая операция, ставящая в соответствие двум высказываниям новое, являющееся истинным только тогда, когда оба исходных высказывания истинны или оба исходных высказывания ложны. Например, высказывание «А равно В» равносильно «В равно А».

По брайлю эквиваленция обозначается в трёх клетках 126 2356 345 (знак равенства в круглых скобках). Этот знак соответствует двойной двунаправленной стрелке. Таким обозначением мы будем пользоваться в дальнейшем. Например:

$$A \text{ or } B (=) B \text{ or } A$$

С помощью описанных логических операций можно конструировать достаточно сложные логические функции (выражения). Для вычисления значения логических выражений необходимо учитывать порядок выполнения основных операций.

Приоритет логических операций:

1. Если в выражении присутствуют скобки, то сначала выполняются действия в них;
2. После этого выполняются все операции отрицания;
3. Затем выполняются операции конъюнкции;

4. После конъюнкции выполняются операции дизъюнкции;
5. В последнюю очередь выполняются операции импликации, а затем эквиваленции.

Операции одного приоритета выполняются в порядке их следования, слева направо. Как в математике, скобки меняют порядок выполнения операций.

Высказывания, образованные из других высказываний, называются составными. Высказывание, никакая часть которого не является высказыванием, называется элементарным.

Например:

$A \text{ or } B \text{ and not } C$ – составное высказывание при любых A , B и C , а высказывание « $2 > 0$ » – простое.

Любое составное логическое высказывание можно представить в виде логического выражения (формулы), состоящего из логических констант (0, 1), логических переменных, знаков логических операций и скобок. Такие составные высказывания в алгебре логики часто называют функциями.

Контрольные вопросы

1. Что изучает алгебра логики?
2. Кто стал основоположником алгебры логики? В чём была его цель?
3. Почему именно алгебра логики стала основным математическим инструментом создания компьютеров?
4. Что такое высказывание?
5. Приведите примеры истинных и ложных высказываний.
6. Как в математических выражениях обозначают истину и ложь?
7. Какие базовые логические операции определяются в алгебре логики? Дайте определение каждой из них.
8. Что такое бинарная операция?
9. Как работает логическая операция импликация?
10. Как работает логическая операция эквиваленция?
11. В каком порядке выполняются логические операции?
12. Что такое простое высказывание?
13. Что такое составное высказывание?

§3.2. Таблицы истинности

Мы знаем, что алгебра логики оперирует на множестве из двух чисел – 0 и 1. На этом множестве можно задать всего четыре унарных операции:

1. Каждому высказыванию операция ставит в соответствие истинное высказывание (тождественная истина);
2. Каждому высказыванию операция ставит в соответствие ложное высказывание (тождественная ложь);
3. Каждому высказыванию операция ставит в соответствие высказывание того же значения, т.е. истинному высказыванию ставится в соответствие истинное, а ложному – ложное (тождественная операция);
4. Каждому высказыванию ставится в соответствие обратное по значению высказывание (отрицание).

Очевидно, что на множестве из двух чисел никаких других унарных операций не существует.

Роль первой из перечисленных унарных операций может играть константа 1 (истина), а роль второй – константа 0 (ложь). Третья операция никак не влияет на высказывание (на значение переменной в составном высказывании), поэтому не используется. Четвертая операция представляет собой уже знакомое отрицание (not). Таким образом, из четырёх возможных унарных операций используется в алгебре логики лишь одна, что соответствует здравому смыслу.

Каждую логическую операцию (или функцию) можно задать таблицей истинности. В таблице истинности перечисляются все возможные значения аргументов и соответствующие им значения операции (функции).

Например, таблица истинности унарной операции имеет два столбца – столбец значений аргумента и столбец значений функции. Очевидно, что строк с данными в такой таблице будет две, поскольку существует всего два различных значения 0 или 1 аргумента.

В верху таблицы записывается строка заголовков. В заголовке первого столбца указывается имя переменной, а в заголовке второго столбца – имя операции (функции). Таким образом, для записи таблицы истинности унарной операции необходимо три строки.

Рассмотрим таблицы истинности описанных выше унарных операций:

Тождественная истина

A 1

1 1

0 1

Тождественная ложь

A 0

1 0

0 0

Тождественная операция

A A

1 1

0 0

Отрицание

A not

1 0

0 1

Обратите внимание, что особые обозначения первых трёх операций не используются, поскольку в этом нет смысла.

Для бинарных операций, т.е. операций с двумя аргументами, в таблице истинности будет три столбца – столбец значений первого аргумента, столбец значений второго аргумента, и столбец значений самой операции. Кроме строки с заголовками в таблице истинности будет четыре строки с данными, поскольку различных наборов значений аргументов всего четыре.

Рассмотрим в качестве примера таблицы истинности четырёх описанных в предыдущем параграфе бинарных операций:

Конъюнкция

A B and

1 1 1

1 0 0

0 1 0

0 0 0

Дизъюнкция

A B or

1 1 1

1 0 1

0 1 1

0 0 0

Импликация

A B \Rightarrow

1 1 1

1 0 0

0 1 1

0 0 1

Эквиваленция

A B (\Leftrightarrow)

1 1 1

1 0 0

0 1 0

0 0 1

Обратите внимание, что два первых столбца во всех таблицах истинности одинаковые. Эти столбцы содержат все варианты значений аргументов и для удобства сравнения таблиц истинности их принято писать именно в таком виде:

1 1

1 0

0 1

0 0

Поскольку у всех таблиц истинности бинарных функций два первых столбца совпадают, то отличаться такие таблицы могут только третьим столбцом значений. Отсюда следует, что различных бинарных функций столько, сколько существует различных вариантов для третьего столбца, т.е. наборов из четырёх бинарных значений (0 и 1). Из комбинаторики известно, что таких наборов может быть

$$2^4 = 16$$

Таким образом, мы получили, что всего существует 16 бинарных логических функций. Обычно используют только четыре бинарные логические функции, описанные в предыдущем параграфе. При этом, импликацию и эквиваленцию можно выразить через отрицание, конъюнкцию и дизъюнкцию.

Часто в логической формуле (функции) присутствуют более, чем две переменные. Очевидно, что таблица истинности такой функции будет содержать большее количество столбцов и строк. Например, таблица истинности логической функции от трёх переменных будет содержать 4 столбца и 8 строк.

Обычно в качестве определения логической операции берут её таблицу истинности. Например, чтобы задать операцию конъюнкция следует выписать её таблицу истинности, а затем положить по определению, что операция, соответствующая данной таблице истинности, называется конъюнкцией.

Доказательства логических тождеств также проводят опираясь на таблицы истинности. Т.е. вычисляют таблицу истинности левой части тождества, затем вычисляют таблицу истинности правой части и сравнивают их. Если таблицы совпадают, то тождество доказано.

Пример. Докажем, что импликацию можно выразить через отрицание, конъюнкцию и дизъюнкцию по формуле:

$$A \Rightarrow B = \text{not } A \text{ or } B$$

Решение: Таблица истинности импликации приведена выше. Вычислим таблицу истинности правой части, подставляя в неё значения аргументов из двух первых столбцов таблицы:

$$\text{Not } 1 \text{ or } 1 = 1$$

$$\text{Not } 1 \text{ or } 0 = 0$$

$$\text{Not } 0 \text{ or } 1 = 1$$

$$\text{Not } 0 \text{ or } 0 = 0$$

Запишем результат в виде таблицы истинности:

$$A \quad B \quad \text{not } A \text{ or } B$$

$$1 \quad 1 \quad 1$$

$$1 \quad 0 \quad 0$$

$$0 \quad 1 \quad 1$$

$$0 \quad 0 \quad 1$$

Сравнивая полученную таблицу с таблицей истинности импликации, расположенной в этом параграфе ранее, видим, что они совпадают и тождество доказано.

Контрольные вопросы

1. Сколько унарных операций можно задать на множестве из двух чисел? Почему?
2. Почему из всех возможных унарных операций в алгебре логики используется лишь одна?
3. Что такое таблица истинности?
4. Сколько строк и столбцов содержит таблица истинности унарной операции? Почему?
5. Сколько строк и столбцов содержит таблица истинности бинарной операции? Почему?
6. Сколько столбцов и строк содержит таблица истинности логической функции от трёх аргументов? Почему?
7. Чем отличаются таблицы истинности различных бинарных операций?
8. Зачем первые два столбца таблицы истинности бинарной операции пишут всегда одинаково?
9. Сколько всего существует логических бинарных операций? Почему?
10. Что является определением бинарной операции?
11. Как проводят доказательства логических тождеств?
12. Как выражается импликация через базовые логические операции?

§3.3. Законы алгебры логики

В алгебре логики существует несколько законов, часть из которых объединяются в группы двойственных. Многие из них аналогичны известным вам законам классической алгебры. Например, законы коммутативности (переместительные) или законы дистрибутивности (распределительные). Как и в классической (обычной) алгебре, эти законы позволяют упрощать выражения, т.е. приводить сложные в записи логические выражения к эквивалентным, но более простым.

Приведём формулировки основных законов алгебры логики.

Закон тождества.

Этот закон утверждает, что суть каждого высказывания остается неизменной на протяжении всего рассуждения, в котором это высказывание фигурирует. В алгебраической форме закон тождества выглядит гораздо проще:

$$A = A$$

Закон противоречия.

Данный закон утверждает, что никакое высказывание не может быть истинно одновременно со своим отрицанием, например, высказывания «это прямоугольный треугольник» и «это не прямоугольный треугольник» одновременно истинными быть не могут. Т.е. A и $\text{not } A$ не могут быть одновременно истинными.

Закон исключенного третьего.

Утверждение этого закона состоит в том, что для каждого высказывания существуют только две возможности - это высказывание либо истинно, либо ложно. третьего варианта не существует. Например, «Сегодня я либо получу пять, либо не получу». Истинно либо высказывание, либо его отрицание, т.е. либо истинно A , либо истинно $\text{not } A$.

Закон снятия двойного отрицания.

Данный закон заключается в том, что отрицать отрицание какого-либо высказывания означает то же самое, что утверждать это высказывание, например, высказывание «неверно, что два умножить на два не равно 4» эквивалентно высказыванию «два умножить на два равно 4». Формулой это выражается так:

$$\text{not } A = A$$

Законы идемпотентности.

Это два парных закона, утверждающих, что:

- Конъюнкция одинаковых высказываний эквивалентна любому из них;
- Дизъюнкция одинаковых высказываний эквивалентна любому из них.

На языке формул эти законы имеют вид:

$$A \text{ and } A = A$$

$$A \text{ or } A = A$$

Из законов идемпотентности следует, что в алгебре логики нет степеней и коэффициентов. Чтобы понять это следствие, вспомним, что конъюнкция аналогична умножению, а дизъюнкция – сложению. Тогда $A \text{ and } A$ есть произведение A на A , а это представляет собой степень. Дизъюнкция $A \text{ or } A$ аналогична выражению $A + A = 2A$.

Законы коммутативности (законы перестановочности).

Два эти закона аналогичны хорошо знакомым законам из классической алгебры (от перестановки слагаемых сумма не меняется). Они утверждают, что высказывания можно менять местами как в конъюнкции, так и в дизъюнкции:

- $A \text{ and } B = B \text{ and } A$
- $A \text{ or } B = B \text{ or } A$

Законы ассоциативности (сочетательные законы).

Как и в предыдущем пункте здесь также существует прямая аналогия с классической алгеброй:

- $(A \text{ and } B) \text{ and } C = A \text{ and } (B \text{ and } C)$
- $(A \text{ or } B) \text{ or } C = A \text{ or } (B \text{ or } C)$

Законы дистрибутивности (распределительные законы).

Аналогия с обычной алгеброй прослеживается и в этих парных законах, однако, некоторые отличия здесь имеют место. Найдите эти отличия самостоятельно! Логические законы дистрибутивности утверждают, что дизъюнкция и конъюнкция равноправны по отношению к дистрибутивности:

- $(A \text{ or } B) \text{ and } C = A \text{ and } C \text{ or } B \text{ and } C$
- $(A \text{ and } B) \text{ or } C = (A \text{ or } C) \text{ and } (B \text{ or } C)$

Законы поглощения.

Законы поглощения констант утверждают, что тождественная ложь не влияет на значение дизъюнкции, а тождественная истина не влияет на значение конъюнкции:

- $A \text{ or } 1 = 1$
- $A \text{ or } 0 = A$
- $A \text{ and } 1 = A$

- $A \text{ and } 0 = 0$

Законы поглощения показывают, как упрощать логические выражения при повторе операнда.

Законы де Моргана.

Эти два парных закона можно выразить следующим образом:

- Отрицание конъюнкции эквивалентно дизъюнкции отрицаний;
- Отрицание дизъюнкции эквивалентно конъюнкции отрицаний.

На языке формул законы де Моргана выглядят так:

- $\text{Not } (A \text{ and } B) = \text{not } A \text{ or not } B$
- $\text{Not } (A \text{ or } B) = \text{not } A \text{ and not } B$

Законы де Моргана связывают с помощью отрицания конъюнкцию и дизъюнкцию. Названы эти законы именем английского логика XIX века А. де Моргана. Их доказательство вы провели решая упражнение 3.2.2.

Контрольные вопросы

1. Почему многие логические законы аналогичны законам классической алгебры?
2. Как на практике используются законы алгебры логики?
3. Сформулируйте основные законы алгебры логики:
 - А) Закон тождества;
 - Б) Закон противоречия;
 - В) Закон исключенного третьего;
 - Г) Закон снятия двойного отрицания;
 - Д) Законы идемпотентности;
 - Е) Законы коммутативности;
 - Ж) Законы ассоциативности;
- 3) Законы дистрибутивности;
- И) Законы поглощения;
- К) Законы де Моргана.
4. В чём отличие логических законов дистрибутивности от аналогичных законов классической алгебры?
5. Какое следствие из законов идемпотентности вы знаете? Поясните его.

6. Как вы думаете, почему логические законы, образующие группы, называются двойственными?

§3.4. Преобразование логических выражений

Одним из методов решения логических задач является применение аппарата алгебры логики. Этот метод состоит из последовательного применения пяти следующих шагов:

1. Изучить условие задачи и выделить из него простые высказывания, обозначив их переменными (большими латинскими буквами);
2. Выразить каждое утверждение (посылку) задачи на языке алгебры логики;
3. Объединить конъюнкцией (логическим умножением) получившиеся формулы и приравнять к единице конечную формулу;
4. Используя законы алгебры логики упростить формулу;
5. Проанализировать получившийся результат.

Рассмотрим применение данного метода на примере решения следующей задачи.

Задача «Кто преступник». Пусть необходимо с достоверностью определить одного из участников преступления, обозначенных буквами русского алфавита А, Б, В, исходя из двух посылок:

1. Если А не участвовал или Б участвовал, то В участвовал;
2. Если А не участвовал, то В не участвовал.

Решение. 1. Составим по условию задачи простые логические высказывания и обозначим их переменными:

А – А участвовал в преступлении;

В – Б участвовал в преступлении;

С – В участвовал в преступлении.

2. Запишем посылки задачи в виде формул:

$\text{not } A \text{ or } B \Rightarrow C$;

$\text{not } A \Rightarrow \text{not } C$.

3. Объединим конъюнкцией получившиеся две формулы и приравняем к единице:

$(\text{not } A \text{ or } B \Rightarrow C) \text{ and } (\text{not } A \Rightarrow \text{not } C) = 1$.

4. Используя законы алгебры логики преобразуем полученную формулу к более простому виду:

Выразим в обеих скобках импликацию через конъюнкцию:

$(\text{not}(\text{not } A \text{ or } B) \text{ or } C) \text{ and } (A \text{ or } \text{not } C) = 1;$

Применим законы де Моргана

$(A \text{ and } \text{not } B \text{ or } C) \text{ and } (A \text{ or } \text{not } C) = 1;$

Применим закон дистрибутивности для раскрытия второй скобки (первая скобка рассматривается как один сомножитель):

$(A \text{ and } \text{not } B \text{ or } C) \text{ and } A \text{ or } (A \text{ and } \text{not } B \text{ or } C) \text{ and } \text{not } C = 1;$

Ещё раз применим закон дистрибутивности для раскрытия оставшихся скобок:

$A \text{ and } \text{not } B \text{ and } A \text{ or } C \text{ and } A \text{ or } A \text{ and } \text{not } B \text{ and } \text{not } C \text{ or } C \text{ and } \text{not } C = 1;$

По законам поглощения, имеем:

$A \text{ and } \text{not } B \text{ and } A = A \text{ and } \text{not } B;$

$C \text{ and } \text{not } C = 0;$

Подставляя, получим:

$A \text{ and } \text{not } B \text{ or } C \text{ and } A \text{ or } A \text{ and } \text{not } B \text{ and } \text{not } C \text{ or } 0 = 1;$

Применив закон коммутативности ко второму слагаемому и ещё раз закон поглощения к последним двум, получим:

$A \text{ and } \text{not } B \text{ or } A \text{ and } C \text{ or } A \text{ and } \text{not } B \text{ and } \text{not } C = 1;$

Используя всё тот же закон дистрибутивности, вынесем за скобки общий множитель A:

$A \text{ and } (\text{not } B \text{ or } C \text{ or } \text{not } B \text{ and } \text{not } C) = 1;$

Опять применим вторую формулу закона дистрибутивности (используя её справа на лево):

$A \text{ and } (\text{not } B \text{ or } (C \text{ or } \text{not } B) \text{ and } (C \text{ or } \text{not } C)) = 1;$

Несколько раз последовательно применим закон поглощения, а также закон коммутативности:

$A \text{ and } (\text{not } B \text{ or } (C \text{ or } \text{not } B) \text{ and } 1) = 1;$

$A \text{ and } (\text{not } B \text{ or } C \text{ or } \text{not } B) = 1;$

$A \text{ and } (\text{not } B \text{ or } C) = 1;$

5. Проанализируем получившуюся формулу, представляющую собой конъюнкцию переменной A и скобки, содержащей дизъюнкцию $\text{not } B \text{ or } C$. Поскольку значение этой формулы равно 1, то каждый операнд должен равняться 1 и значит $A = 1$.

Ответ: Иванов участвовал в преступлении.

При решении подобных задач могут быть полезны следующие формулы:

- $A \text{ or } (A \text{ and } B) = A$;
- $A \text{ and } (A \text{ or } B) = A$.

Контрольные вопросы

1. Какие методы решения логических задач вы знаете?
2. Какие результаты алгебры логики можно применять для решения логических задач?
3. Какие шаги следует предпринять для решения логической задачи методом алгебры логики?
4. Объясните решение задачи «Кто преступник».

Глава 4

Теория алгоритмов

§4.1. Алгоритмизация процессов

Как вы уже знаете, алгоритм – это строго детерминированная (определённая) последовательность команд, предназначенная для преобразования некоторого объекта из начального состояния в конечное. При этом все команды алгоритма обязаны быть понятными исполнителю. В математике для выполнения типовых операций используются алгоритмы (правила), описывающие последовательности действий. Например, правила сложения дробных чисел, решения квадратных уравнений и т. д. Обычно любые инструкции и правила представляют собой последовательность действий, которые необходимо выполнить в определенном порядке.

Правило (инструкция), определяющее порядок выполнения действий над данными с целью получения искомых результатов, и является алгоритмом.

Понятие алгоритма такое же основополагающее для информатики, как и понятие информации. Поэтому важно понять теорию алгоритмов и научиться алгоритмизировать необходимые операции.

Название «алгоритм» произошло от латинской формы имени величайшего среднеазиатского математика Мухаммеда ибн Муса ал-Хорезми (Alhorithmi), жившего в 783–850 гг. В своей книге «Об индийском счете» он изложил правила записи натуральных чисел с помощью арабских цифр и правила действий над ними «столбиком», знакомые теперь каждому школьнику. В XII веке эта книга была переведена на латынь и получила широкое распространение в Европе.

Алгоритмы могут описывать процессы преобразования самых разных объектов. Например, алгоритм сборки мебели, который обычно приводится в инструкции. Этот алгоритм преобразует объект «набор деталей» в объект «готовая мебель».

В современном информационном обществе широкое распространение получили вычислительные алгоритмы, которые описывают

преобразование информационных объектов (данных). Алгоритм позволяет формализовать выполнение информационного процесса, реализуемого на компьютере или на каком-либо другом цифровом устройстве.

Алгоритм – это точное предписание, однозначно определяющее вычислительный процесс, ведущий от начальных (входных) данных к результату (выходным данным).

Как видно из нескольких вариантов определений, приведённых выше, для выполнения алгоритма нужен исполнитель. Исполнитель понимает команды алгоритма и выполняет их. При разработке алгоритма надо следить за тем, чтобы все его команды были понятны исполнителю.

Например, если вы складываете два числа в столбик, значит вы являетесь исполнителем алгоритма сложения. Если числа складывает калькулятор, то исполнитель он. Очевидно, что для человека и калькулятора алгоритмы должны быть разными.

Одним из самых распространённых исполнителей является персональный компьютер. Алгоритм для компьютера представляет собой четкую инструкцию по выполнению некоторой операции (процесса). В общем случае разработка алгоритма является сложным и трудоемким процессом. Однако, уметь алгоритмизировать свои действия крайне важно для эффективного использования компьютера. Представление информационного процесса в форме алгоритма позволяет автоматически выполнить его компьютеру.

Алгоритмизация – это техника разработки алгоритма для решения какой-либо задачи на компьютере.

Напомним, что исполнитель алгоритма – это некоторая абстрактная или реальная (техническая или биологическая) система, способная выполнить команды (указания, инструкции), предписываемые алгоритмом.

Каждый исполнитель обладает определенным набором команд, которые он может выполнить. Алгоритм должен быть понятен исполнителю, содержать только понятные ему команды.

Например, редактирование текста состоит из удаления, копирования, перемещения или замены какого-либо фрагмента. Исполнитель

любого алгоритма редактирования текста должен быть в состоянии выполнить эти операции.

Компьютер выполняет алгоритмы формально, «не зная» содержание поставленной задачи, а только строго выполняя последовательность действий, предусмотренную алгоритмом.

Заметим, что существуют алгоритмы, которые выполняет человек работая на компьютере, а существуют алгоритмы, которые автоматически выполняет компьютер.

В качестве иллюстрации понятий «алгоритм», «исполнитель» и «объект» рассмотрим процедуру редактирования текста. У объекта «текст» есть свойства, например, «шрифт», которым он набран. Рассмотрим алгоритм изменения свойства «шрифт» объекта «текст».

Алгоритм изменения свойства текста необходимо разбить на операции, которые должны выполняться с помощью отдельных команд, понятных исполнителю.

Если алгоритм записан на бумаге и вы читая команды этого алгоритма выполняете их, то вы являетесь исполнителем. Когда вы используя клавиатуру даёте команды компьютеру, то исполнителем в этой ситуации уже является он. Точнее, здесь исполнителем является компьютер с запущенным редактором, в котором происходит обработка текста. Исполнитель алгоритма изменения шрифта должен быть способен выполнить образующие алгоритм команды.

Итак, пусть имеется компьютер с запущенным редактором Word и в окне редактора есть произвольный текст, набранный шрифтом Areal. Приведем алгоритм для человека (пользователя), для изменения шрифта на Times New Roman.

Название алгоритма: Изменения шрифта.

Описание входных данных: Текст в редакторе Word, набранный шрифтом Areal.

Описание выходных данных: Тот же текст в редакторе Word, набранный шрифтом Times New Roman.

1. Начало.
2. Выделить весь текст введя команду Ctrl + A.
3. Вызвать диалог изменения шрифта введя команду Ctrl + D.
4. В списке шрифтов выбрать шрифт Times New Roman.

5. Нажать клавишу Enter.

6. Конец.

Исполнитель этого алгоритма – это человек, который работает на компьютере. Обратите внимание, что не каждый человек может быть исполнителем этого алгоритма. Выполнить алгоритм может только тот, кто имеет определенные пользовательские навыки.

Команды, которые с клавиатуры подавал человек, исполнял компьютер с помощью программы Word. Компьютер понимал какие клавиши нажаты и выполнял соответствующее действие, т.е. был исполнителем. Очевидно, что компьютер исполнял команды только из списка клавиатурных команд.

При реализации вышеприведенного алгоритма каждую команду подаёт человек, а компьютер выполняет. Это происходит потому, что алгоритм составлен из команд, которые понимает человек, но не понимает компьютер. Исполнителем такого алгоритма является человек, а вводимые им клавиатурные команды исполняет уже компьютер. Т.е., компьютер понимает не сами команды алгоритма, а клавиатурные команды вводимые человеком.

При разработке алгоритма (алгоритмизации) необходимо чётко представлять какие данные являются начальными, что необходимо получить и какие команды для этого следует использовать. Например, для решения квадратного уравнения необходимо знать его коэффициенты (входные данные), получить нужно корни (выходные данные, результат), а использовать можно любые арифметические действия.

Наряду с приведённым выше алгоритмом изменения шрифта, существуют автоматически исполняемые компьютером. Представление алгоритма с помощью особых, понятных компьютеру команд, позволяет выполнять ему такие алгоритмы самостоятельно. В этом случае компьютер будет работать автоматически, без участия человека. Говорят, что компьютер исполняет программу, реализующую алгоритм. Алгоритм, записанный на понятном компьютеру языке, называется программой, а такой язык – языком программирования. Таким образом, программа – это последовательность команд, понятных исполнителю «компьютер».

Контрольные вопросы

1. Что такое алгоритм? Приведите примеры.
2. Какие основополагающие понятия информатики вы знаете?
3. Как произошло слово «алгоритм»?
4. Какие алгоритмы получили распространение в современном обществе? Почему?
5. Что такое исполнитель?
6. Чем характеризуется исполнитель?
7. Что такое алгоритмизация?
8. Каждый ли человек может быть исполнителем приведённого в параграфе алгоритма изменения шрифта? Почему?
9. Что необходимо понимать перед составлением алгоритма?
10. Что такое компьютерная программа?

§4.2. Основные алгоритмические конструкции

Каждый алгоритм состоит из отдельных базовых конструкций (структур). Освоение принципов алгоритмизации опирается на использование этих базовых конструкций. Итак, при разработке алгоритмов используют три основные алгоритмические конструкции:

- Линейная (следование);
- Ветвление (условный переход);
- Циклическая (повторение операций).

Рассмотрим каждую из этих трёх алгоритмических конструкций.

Большинство алгоритмов содержат фрагменты, в которых команды должны быть выполнены последовательно одна за другой. Некоторые алгоритмы полностью образованы последовательно выполняемыми командами. Такие алгоритмы (или фрагменты алгоритмов) называются линейными. Например, алгоритм сложения правильных дробей является линейным.

Таким образом, линейный алгоритм (или фрагмент) – это такой алгоритм, в котором все команды выполняются последовательно одна за другой и только один раз. Другими словами, линейный фрагмент алгоритма образуется последовательностью команд, следующих одна за другой от начала к концу.

На практике часто встречаются алгоритмы, в которых в зависимости от первоначальных условий или результатов промежуточных операций необходимо выполнять те или иные команды. Такие алгоритмы содержат структуру ветвления, т.е. фрагмент, в котором осуществляется выбор одного из двух возможных направлений дальнейшей работы.

Базовая структура ветвления обеспечивает в зависимости от результата проверки условия (истина или ложь) выбор одного из альтернативных путей работы алгоритма. Каждый из путей ведет к общему выходу, поэтому работа алгоритма продолжится независимо от того, какой путь будет выбран в этом фрагменте.

Структуру ветвления можно представить схематически следующим образом:

Если условие истинно, то выполняются команды группы 1, а если условие ложно, то выполняются команды группы 2; Затем основной алгоритм продолжается.

Например, структуру ветвления содержит алгоритм решения квадратного уравнения. Если вычисленное при работе алгоритма значение дискриминанта меньше нуля, то такое уравнение не имеет действительных корней и должно быть выдано сообщение об этом, а работа алгоритма прекращена. В случае, когда дискриминант больше или равен нулю, уравнение имеет два (возможно совпадающих) корня и далее должны быть применены формулы для их вычисления.

Для решения многих задач применяются алгоритмы с отдельными многократно повторяющимися участками. Такие алгоритмы (или фрагменты) называются циклическими, а соответствующая конструкция – циклом.

Базовая структура цикл обеспечивает многократное выполнение группы команд, которая называется телом цикла.

Таким образом, цикл – это последовательность команд алгоритма, которая повторяется до тех пор, пока не будет выполнено заданное условие. Использование такой конструкции значительно снижает трудоемкость разработки алгоритмов и написания программ.

Например, хорошо вам известный Алгоритм перевода десятичного числа в двоичное является циклическим. Вспомним, что получив

одну двоичную цифру (первую или последнюю?) как остаток от деления на 2, мы переходим к делению на два следующего числа - целой части от предыдущего деления. Так продолжается пока в качестве целой части будет получено число 0.

Каждая из этих трёх базовых алгоритмических структур имеет единственный вход и единственный выход. Другими словами, всегда есть единственная команда (или начало алгоритма), предшествующая какой-либо структуре. И всегда есть единственная команда (или конец алгоритма), идущая после базовой структуры.

Основные (базовые) алгоритмические структуры могут быть частями друг друга. Например, тело цикла может содержать структуру ветвления, а группы команд, выполняющихся при истинности или ложности условия в этой структуре ветвления, могут быть достаточно длинными линейными фрагментами. Также могут использоваться однотипные вложенные конструкции, например, цикл внутри цикла.

Контрольные вопросы

1. Сколько существует основных алгоритмических структур?
2. Используя известные вам математические алгоритмы, приведите примеры:
 - А) линейного алгоритма;
 - Б) ветвящегося алгоритма;
 - В) циклического алгоритма.
3. Объясните принципы работы каждой алгоритмической структуры.
4. Каким может быть взаимное расположение базовых алгоритмических структур в основном алгоритме?
5. Как используя описанные алгоритмические конструкции организовать выбор из трёх и более вариантов?
6. Приведите пример алгоритма, использующего вложенные циклы?

§4.3. Свойства алгоритмов

Каждый человек в повседневной деятельности постоянно сталкивается с необходимостью выполнения какого-либо алгоритма.

С алгоритмами мы сталкиваемся в учебе, быту, профессиональной деятельности. При этом все алгоритмы подчиняются некоторым общим законам. В этом параграфе будут рассмотрены пять основных свойств алгоритма.

Итак, алгоритм – это понятное и точное предписание исполнителю выполнить конечную последовательность команд, приводящую от исходных данных к искомому результату. Каждый алгоритм должен отвечать следующим пяти свойствам:

- Понятность;
- Дискретность;
- Детерминированность;
- Результативность;
- Массовость.

Поясним смысл каждого из этих свойств.

Понятность. Это свойство обязывает алгоритм быть понятным для исполнителя. Очевидно, что исполнитель алгоритма должен понимать каждую команду алгоритма, быть способным его выполнять. Иными словами, имея алгоритм и произвольный вариант исходных данных, исполнитель должен знать, как надо действовать для выполнения этого алгоритма. Например, алгоритм сложения обыкновенных дробей будет понятен ученику 7 класса, поскольку он умеет выполнять все необходимые арифметические операции, а первокласснику этот алгоритм выполнить не удастся. С другой стороны, алгоритм решения дифференциального уравнения не сможет выполнить и ученик 7 класса. Алгоритм может выполнять свою функцию только тогда, когда он понятен исполнителю.

Таким образом, при разработке алгоритма нужно четко представлять для какого исполнителя он создается. В алгоритм можно включать только команды из списка команд того исполнителя, для которого разрабатывается алгоритм. Например, при работе на каком-либо языке программирования в текст программы можно включать только конструкции данного языка и ничего другого. Даже если некоторая запись кажется вам понятной, она может быть совершенно непонятной компьютеру и алгоритм не заработает.

Дискретность (прерывность, раздельность). Как вы уже знаете, алгоритмы состоят из отдельных команд, которые исполнитель выполняет одну за другой в строгой последовательности. Разделение алгоритма на отдельные команды является важным его свойством и называется дискретностью. Иначе говоря, дискретность состоит в том, что прежде, чем приступить к выполнению следующего шага алгоритма, необходимо завершить выполнение предыдущего.

По смыслу слова дискретность – это прерывность или раздельность. Алгоритм должен представлять процесс решения задачи как последовательное выполнение отдельных друг от друга простых шагов. Нельзя приступить к выполнению следующего шага пока не завершён предыдущий. Например, в алгоритме решения квадратного уравнения нельзя приступить к проверке дискриминанта на неотрицательность, пока значение дискриминанта не вычислено.

Детерминированность (определённость). Алгоритм должен быть разработан таким образом, чтобы, выполнив очередную команду, исполнитель точно знал какую команду необходимо исполнять следующей. Это свойство алгоритма называется детерминированностью (определённостью). Также, будучи понятным, алгоритм не должен содержать команды, смысл которых может восприниматься исполнителем неоднозначно. Выполнение каждой команды должно приводить к конкретному результату.

Иными словами, каждая команда алгоритма должна быть четкой, однозначной и не оставлять места для произвола. Благодаря этому свойству выполнение алгоритма носит механический характер и не требует никаких дополнительных указаний или сведений о решаемой задаче.

Например, в алгоритме умножения десятичных дробей недопустима команда: «отделите запятой две–три цифры в результате». Исполнитель не поймёт сколько цифр необходимо отделить запятой в результате.

Результативность (или конечность). Алгоритм должен обеспечивать преобразование объекта из начального состояния в конечное за конечное число шагов. Такое свойство алгоритма называется ре-

зультативностью или конечностью. Это свойство состоит в том, что за конечное число шагов алгоритм либо должен приводить к решению задачи, либо останавливаться из-за невозможности получить решение с выдачей соответствующего сообщения.

Смысл этого требования к алгоритмам состоит в том, что при точном исполнении всех команд алгоритма процесс решения задачи должен прекратиться за конечное число шагов и должен быть получен определенный постановкой задачи ответ.

Например, алгоритм перевода десятичного числа в двоичное завершится не зависимо от того, насколько большое число вы переводили в двоичную систему.

Массовость. Это свойство означает, что алгоритм решения задачи разрабатывается в общем виде, т.е. он должен быть применим для некоторого класса задач, различающихся лишь исходными данными. При этом исходные данные могут выбираться из некоторой области, которая называется областью применимости алгоритма.

Свойство массовости предполагает, что алгоритм должен быть пригоден для решения всех задач данного типа. Практическую пользу будут иметь только алгоритмы, разработанные для целого класса задач данного типа. Например, алгоритм решения квадратного уравнения применим к любому уравнению, являющемуся квадратным.

Таким образом, если разработанная Вами последовательность команд не обладает хотя бы одним из перечисленных выше свойств, то она не может считаться алгоритмом!

Контрольные вопросы

1. Перечислите основные свойства алгоритма.
2. Поясните, что означает свойство:
 - А) Понятность;
 - Б) Дискретность;
 - В) Детерминированность;
 - Г) Результативность;
 - Д) Массовость.
3. Каким свойством алгоритмов не обладают следующие последовательности команд обработки натурального числа:

А)

1. Возведите исходное число в квадрат.
2. Перейдите к шагу 1.

Б)

1. Прибавьте к исходному числу число 7.
2. Вычтите из результата число 2 или 3.

В)

1. Умножьте исходное число 6 на число 2.
2. Разделите результат 12 на число 4.

§4.4. Формы представления алгоритмов

Записывать алгоритмы принято по определённой схеме:

1. Каждый алгоритм должен иметь имя, которое раскрывает его смысл.
2. Необходимо обозначить начало и конец алгоритма.
3. Описать входные и выходные данные.
4. Указать команды, которые позволяют выполнять определенные действия над входными данными.

При этом алгоритм можно записывать несколькими способами, среди которых на практике наиболее распространены следующие:

- словесный (запись на естественном языке);
- графический (блок-схема);
- псевдокод (формализованное описание алгоритмов на условном алгоритмическом языке);
- программный (тексты на языках программирования).

Словесный способ записи алгоритмов представляет собой описание последовательных этапов обработки данных. Алгоритм задается в произвольном изложении на естественном языке.

Запишем алгоритм нахождения наибольшего общего делителя (НОД) двух натуральных чисел (алгоритм Евклида):

1. Ввод двух чисел.
2. Если числа равны, то взять первое из них в качестве ответа и остановиться, в противном случае продолжить выполнение алгоритма.
3. Определить большее из чисел.
4. Заменить большее число разностью большего и меньшего.

5. Повторить алгоритм с шага 2.

Описанный алгоритм применим к любым натуральным числам и приводит к решению поставленной задачи. Убедитесь в этом самостоятельно, определив с помощью этого алгоритма наибольший общий делитель чисел 125 и 75.

Словесный способ представления алгоритма не имеет строгой формализации, достаточно многословен и в некоторых случаях допускает неоднозначность толкования инструкций. Эти причины обуславливают его редкое использование. Однако, в учебных целях и в случае использования рельефно-точечной системы Брайля словесный способ записи алгоритма будет весьма эффективен.

Графический способ представления алгоритмов в виде блок-схем является более компактным и наглядным по сравнению со словесным, что определяет его частое использование зрячими обучающимися. Наряду с этим, существующие на настоящий момент технологии ручного изготовления рельефных рисунков достаточно сложны и трудоёмки, поэтому блок-схемы не применяются обучающимися с глубоким нарушением зрения. Однако, блок-схемы могут встретиться на экзамене по информатике или в заданиях олимпиад, поэтому следует выучить значение основных блоков.

При графическом представлении алгоритм изображается в виде последовательности связанных между собой стрелками блоков (геометрических фигур), каждый из которых соответствует реализации какой-либо алгоритмической структуры или выполнению одной или нескольких команд (операторов). Такое графическое представление алгоритма называется блок-схемой. В блок-схеме каждому типу действий (вводу исходных данных, вычислению значений выражений, проверке условий, управлению повторением команд, окончанию обработки и т.п.) соответствует геометрическая фигура – блок. Блоки соединяются линиями со стрелками, определяющими очередность выполнения команд. Т.е. элементы алгоритма изображаются с помощью заданных геометрических фигур.

На блок-схеме хорошо видна структура алгоритма, по которой исполнителю (человеку) удобно отслеживать процесс его выполнения. Ниже приведено описание наиболее часто употребляемых блоков:

1. Овал – начало или конец алгоритма. Часто обозначается прямоугольником со скруглёнными углами. Ставится в самом верху и в самом низу блок-схемы для обозначения начала и конца алгоритма.

2. Параллелограмм – ввод начальных данных или вывод результата. Внутри параллелограмма указываются имена переменных, в которые осуществляется ввод или в которых хранятся значения для вывода. Также в параллелограмме могут указываться фразы для вывода. Например, «Решений нет».

3. Прямоугольник – обозначает выполнение команд. В прямоугольнике записывается присваивание значений переменным и вычислительные формулы. В некоторых случаях в одном прямоугольнике записывают несколько действий. Например, формулы для вычисления обоих корней квадратного уравнения можно записать в один прямоугольник.

4. Ромб – проверка условия. Ромб служит для обозначения алгоритмической конструкции «Ветвление». Внутри ромба записывается условие, в зависимости от истинности или ложности которого определяется направление дальнейшего выполнения алгоритма. Из ромба выходят две стрелки, одна имеет подпись «Да», а другая – «Нет». Если условие истинно, то дальнейшее выполнение алгоритма идет по стрелке, помеченной «Да», а если ложно, то по стрелке «Нет».

5. Шестиугольник – модификация. Слово «модификация» означает видоизменение, преобразование. Этот блок обозначает алгоритмическую циклическую структуру. Иначе говоря, используется для организации циклических конструкций. Внутри блока записывается параметр цикла, для которого указываются его начальное значение, конечное значение и шаг изменения значения параметра для каждого прохождения цикла. На уровне школьной программы этот блок используется очень редко, только в случае решения задач повышенной сложности.

Псевдокод представляет собой формализованную систему обозначений и правил, предназначенную для единообразной записи алгоритмов. Псевдокод занимает промежуточное место между естественным и формальным языками. С одной стороны, он близок

к обычному естественному языку, поэтому алгоритмы могут на нем записываться и читаться как обычный текст. С другой стороны, в псевдокоде используются некоторые формальные конструкции и математическая символика, что приближает запись алгоритма к общепринятой математической записи.

В псевдокоде не приняты строгие синтаксические правила для записи команд, присущие формальным языкам, что облегчает запись алгоритма на стадии его проектирования и дает возможность использовать более широкий набор команд, рассчитанный на абстрактного исполнителя. Однако в псевдокоде обычно имеются некоторые конструкции, присущие формальным языкам, что облегчает переход от записи на псевдокоде к записи алгоритма на формальном языке (языке программирования).

Единого или формального определения псевдокода не существует, поэтому возможны различные псевдокоды, отличающиеся набором служебных слов и основных (базовых) конструкций.

Примером псевдокода является школьный алгоритмический язык в русской нотации. Приведем список некоторых служебных слов этого языка:

- алг (алгоритм);
- дано (входные данные);
- надо (описание результата);
- сим (символьный);
- лог (логический);
- цел (целый);
- вещ (вещественный);
- арг (аргумент);
- нач (начало);
- ввод (ввод данных);
- вывод (вывод результата);
- если то иначе (ветвление);
- нц (начало цикла);
- кц (конец цикла);
- кон (конец).

Общий вид алгоритма можно записать с помощью этого псевдокода:

алг (название алгоритма)

дано (область применимости алгоритма)

надо (цель выполнения алгоритма)

нач (начало алгоритма)

последовательность команд (тело алгоритма)

кон (конец алгоритма)

Часть алгоритма от слова алг до слова нач называется заголовком, а часть, заключенная между словами нач и кон-телом алгоритма.

Заметим, что слова дано и надо не обязательны, их часто опускают.

При записи алгоритма в словесной форме, в виде блок-схемы или на псевдокоде допускается определенный произвол при изображении команд. Вместе с тем такая запись точна настолько, что позволяет человеку понять суть алгоритма и исполнить его. Однако на практике в качестве исполнителей алгоритмов используются компьютеры. Поэтому алгоритм, предназначенный для исполнения на компьютере, должен быть записан на понятном ему языке и строго соответствовать описанным в параграфе 4.3. свойствам, что требует использования формализованного языка. Такой язык принято называть языком программирования, а запись алгоритма на этом языке – программой для компьютера. С программированием вы уже знакомы. Следующая глава будет посвящена более сложным его вопросам.

Контрольные вопросы

1. Какие способы записи алгоритма вы знаете?
2. Какой из известных вам способов записи алгоритма удобен при работе по брайлю? Почему?
3. Какие геометрические фигуры используются на блок-схеме?
4. Для чего на блок-схеме служит:
 - Овал;
 - Прямоугольник;
 - Параллелограмм;
 - Ромб;
 - Шестиугольник?

5. Что на блок-схеме означают стрелки с надписями «Да» и «Нет», выходящие из ромба?

6. Какую структуру имеет алгоритм, блок-схема которого состоит только из овалов, параллелограммов и прямоугольников?

7. Что такое псевдокод?

8. Чем среди других способов записи алгоритмов выделяется язык программирования?

§4.5. Машина Тьюринга

В теоретической информатике существует понятие универсального исполнителя. Это такой формальный исполнитель, который может имитировать любой другой формальный исполнитель. Очевидно, что как физическое устройство универсальный исполнитель не существует, поскольку теоретически информация может кодироваться сколь угодно длинными сообщениями, а любой физический носитель информации ограничен (конечен).

Идею универсального формального исполнителя предложили почти одновременно и независимо друг от друга два выдающихся ученых А. Тьюринг и Э. Пост. Предложенные ими исполнители различались между собой, но оказалось, что они могут имитировать друг друга.

Универсальный исполнитель принято называть Машиной. Также принято давать машинам имена их изобретателей. Соответственно, универсального исполнителя, придуманного А. Тьюрингом, называют Машиной Тьюринга, а исполнителя, описанного Э. Постом, называют Машиной Поста. Позже появились и другие универсальные исполнители, например, машина Минского. Ниже будет рассмотрена Машина Тьюринга.

В 1936 г. Аланом Тьюрингом был предложен абстрактный универсальный исполнитель вычислительных алгоритмов, который стали называть Машиной Тьюринга. Это не реальное устройство, а математическая модель (идеальный объект). Существуют программные реализации Машины Тьюринга на компьютере, но считать их воплощением самой Машины нельзя, поскольку объём обрабатываемой ими информации ограничен. С подобными объектами вы сталкива-

лись в физике – идеальный газ, математический маятник, материальная точка и т.п.

Машина Тьюринга состоит из бесконечной в обе стороны ленты, разделенной на ячейки (клетки), и каретки (автомата, считывающей головки), способной двигаться вдоль ленты и считывать из ячейки или записывать туда символы некоторого алфавита. Движения каретки управляются программой.

Программы для Машины Тьюринга записываются в виде таблицы, где первый столбец и строка содержат буквы внешнего алфавита и возможные внутренние состояния автомата (внутренний алфавит). В ячейках таблицы записываются команды для Машины Тьюринга.

Буква, которую считывает каретка в ячейке в данный момент, и внутреннее состояние каретки определяют, какую команду нужно выполнить. Команда определяется пересечением строки и столбца, озаглавленных символами внутреннего и внешнего алфавитов.

Каждая Машина Тьюринга имеет свой внутренний и внешний алфавит. Внутренний алфавит представляет собой конечное множество символов $A = \{a_1, a_2, \dots, a_n\}$, которые можно записывать в ячейки ленты. Например, внешний алфавит может состоять из двух символов $\{0, 1\}$.

Внутренний алфавит – это также конечное множество, каждый элемент которого соответствует некоторому состоянию каретки. Среди этих состояний должно быть начальное, с которого Машина начинает работу, и конечное, которым работа заканчивается (состояние останова). Внутренний алфавит принято обозначать $Q = \{q_0, q_1, q_2, \dots, q_m\}$.

Таким образом, внутренний и внешний алфавиты определяют конкретную Машину Тьюринга. Изменив один или оба алфавита мы получим другую Машину Тьюринга.

В начале работы Машина Тьюринга находится в состоянии q_0 и каретка располагается на некоторой клетке ленты, которую считают начальной.

В дальнейшем для упрощения понимания мы будем рассматривать внешний алфавит из двух символов 0 и 1.

Машина Тьюринга в процессе своей работы может выполнять следующие действия:

- Записывать символ внешнего алфавита в ячейку ленты, заменяя находившийся в ней;
- Передвигаться на одну ячейку влево или вправо;
- Менять свое внутреннее состояние.

Одна команда для Машины Тьюринга представляет собой конкретную комбинацию этих трех составляющих:

- Указания, какой символ внешнего алфавита (0 или 1) записать в ячейку над которой стоит каретка (при этом находившийся там ранее символ будет затёрт);
- В какую из соседних клеток (слева или справа) переместиться;
- В какое состояние Q_i перейти.

При этом команда может содержать и не все составляющие, например, не менять символ, не передвигаться или не менять внутреннего состояния.

Каждая команда Машины Тьюринга содержит не более одного действия из каждой группы. Приведём несколько примеров команд для Машины Тьюринга.

$0 \rightarrow q_1$

Эта команда записывает символ 0 в текущую ячейку, смещается на одну ячейку вправо (на это указывает стрелка вправо) и переходит в состояние q_1 .

$1 \leftarrow q_3$

Такая команда записывает в текущую ячейку символ 1, смещается на ячейку влево и переходит в состояние q_3 .

$0 S q_1$

Записать в текущую ячейку 0, остановиться и перейти в состояние q_1 .

Поскольку действие Машины Тьюринга зависит от того, в каком состоянии в настоящее время она находится и какой символ записан в текущей ячейке, программу для неё, как уже говорилось, удобно записывать в таблице. Строки этой таблицы озаглавлены символами внешнего алфавита, а столбцы озаглавлены символами внутреннего. В клетки этой таблицы записываются команды.

Если в текущей ячейке на ленте записан символ a_i , а Машина находится в состоянии q_j , то выполняется команда, стоящая на пересечении строки a_i и столбца q_j .

Такую таблицу называют функциональной схемой данной Машины. Функциональная схема и является программой для Машины Тьюринга. Из неё, в частности, видно, каковы внешний и внутренний алфавиты данной Машины.

Пусть, например, на ленте записана последовательность из некоторого количества единиц. Такую последовательность можно считать записью натурального числа, соответствующего данному количеству единиц. Тогда функциональная схема, приведенная ниже, заставляет Машину Тьюринга увеличить на одну количество единиц в этой последовательности, т.е. прибавить к данному натуральному числу единицу. Заметим, что при начале работы каретка должна стоять на любой ячейке с единицей.

A/Q	q_0
0	1 S q_0
1	1 $\leftarrow q_0$

Доказать, что Машина Тьюринга является универсальным исполнителем, нельзя. Так же, например, как нельзя доказать закон сохранения энергии в физике. Но практика составления алгоритмов показывает, что любую задачу, которую сегодня умеет решать человек, можно запрограммировать на Машине Тьюринга. Этот экспериментальный факт, называемый тезисом Тьюринга, формулируют так:

для задачи существует алгоритм решения тогда и только тогда, когда имеется подходящая Машина Тьюринга, посредством которой можно решить эту задачу.

Контрольные вопросы

1. Какого формального исполнителя называют универсальным?
2. Почему универсальный исполнитель не может существовать как физическое устройство?
3. Кто первым предложил идею универсального исполнителя?
4. Из чего состоит Машина Тьюринга?

5. Почему лента Машины Тьюринга должна быть бесконечной?
6. Что такое внутренний и внешний алфавиты Машины Тьюринга?
7. Какие действия может выполнять Машина Тьюринга?
8. Приведите примеры команд для Машины Тьюринга.
9. Как задаётся программа для Машины Тьюринга?
10. Что называют функциональной схемой Машины Тьюринга?
11. Как по функциональной схеме определить какую команду должна выполнить Машина Тьюринга?
12. С какой стороны будет приписана единица к последовательности единиц на ленте в примере увеличения натурального числа на 1?
13. В чём состоит тезис Тьюринга?

Глава 5

Программирование и анализ данных без визуального контроля

Материал этой главы предполагает, что читатель уже знаком с основами программирования на языке Python.

§5.1. Структуры данных

«Структуры данных» в программировании предназначены для хранения различных связанных между собой данных в удобной форме. Наличие нескольких форм хранения обеспечивает быстрый и эффективный способ доступа к необходимым данным. В чём состоит это удобство станет понятным после знакомства со структурами данных языка Python.

В Python существуют четыре разновидности структур данных:

- Список;
- Кортеж;
- Словарь;
- Множество.

Рассмотрим каждую из этих структур.

Список (list) – это структура данных, которая содержит упорядоченный набор элементов, т.е. хранит последовательность элементов. При создании список элементов должен быть заключён в квадратные скобки, чтобы интерпретатор Python мог отличить его от других структур данных. Если в квадратных скобках ничего не указывать, будет создан пустой список. После того, как список создан, можно добавлять, удалять или искать его элементы. В отличие от строк, списки являются изменяемыми объектами, в них можно удалять и добавлять элементы.

При работе со списком используются понятия объектно-ориентированного программирования. Одним из важнейших таких понятий является класс. У каждого класса существуют свои методы, т.е. функции, определённые для использования применительно к данному классу. Эти функции будут доступны только при работе с объек-

том данного класса. Например, Python предоставляет метод `append()` для класса списков (`list`), который позволяет добавлять элемент к концу списка.

Например:

```
my_list.append('Новый элемент списка')
```

Добавляет указанную в апострофах строку в конце списка `my_list`. Обратите внимание, что имя метода отделяется от имени класса точкой «.».

Список содержит элементы, которые представляют собой переменные, определённые для использования с данным классом. Эти переменные можно использовать только после того, как объект данного класса будет создан.

Пример 1. Программа обрабатывает список предметов для подготовки домашнего задания.

```
homework = ['Физика', 'Геометрия', 'Английский язык']
print('Я должен сделать', len(homework), 'задания:')
for item in homework:
    print(item, end=' ')
print('\n')
print('В электронном журнале добавили ещё 2 задания и теперь
нужно сделать:')
homework.append('Русский язык')
homework.append('Алгебра')
for item in homework:
    print(item, end=' ')
print('\n')
print('Я сделал физику и осталось сделать:')
del homework[0]
for item in homework:
    print(item, end=' ')
print('\n')
print('Буду делать уроки в алфавитном порядке:')
homework.sort()
print(homework)
```

В результате выполнения этой программы на экран будет выведено:

Я должен сделать 3 задания:

Физика Геометрия Английский язык

В электронном журнале добавили ещё 2 задания и теперь нужно сделать:

Физика Геометрия Английский язык Русский язык Алгебра

Я сделал физику и осталось сделать:

Геометрия Английский язык Русский язык Алгебра

Буду делать уроки в алфавитном порядке:

['Алгебра', 'Английский язык', 'Геометрия', 'Русский язык']

В первой строке программы создаётся список `homework` и первые три его элемента заполняются данными. Затем выводится сообщение, содержащее информацию о количестве элементов в этом списке и далее с помощью цикла выводится его содержимое. Обратите внимание, что функция `print()` в теле цикла содержит второй аргумент `end = ' '`, который указывает, что вывод следует заканчивать не переводом строки, а символом пробела. Благодаря этому список предметов на экране располагается в строку. Следующая функция `print()` содержит единственный аргумент `'\n'`, который обеспечивает переход на новую строку и следующий вывод начнётся с новой строки.

Следующие несколько строк программы добавляют два элемента к списку с помощью метода `append()` и выводят на экран новый список с добавленными элементами.

Далее идёт часть программы, демонстрирующая удаление элемента из списка с помощью инструкции `del` и выводящая результат на экран аналогично предыдущему.

Последняя часть программы используя метод `sort()` упорядочивает элементы списка в алфавитном порядке и выводит их на экран одной функцией `print()` с именем списка в качестве аргумента. Обратите внимание, что в этом случае названия учебных предметов выводятся на экран в квадратных скобках через запятую.

В этом примере в списке хранятся только строки (названия учебных предметов), однако в список можно добавлять любые объекты, включая числа или другие списки.

Кортежи (tuple) служат для хранения нескольких объектов вместе. Их можно рассматривать как аналог списков, но без такой обширной функциональности, которую предоставляет класс списка. Одна из важнейших особенностей кортежей заключается в том, что они неизменяемы, так же, как и строки. Т.е. модифицировать кортежи невозможно.

Кортежи обозначаются указанием элементов, разделённых запятыми; по желанию их можно ещё заключить в круглые скобки.

Кортежи обычно используются в тех случаях, когда в процессе работы программы набор значений не должен изменяться.

Пример 2. Программа описывает наличие автомобилей в торговом салоне.

```
car=('Мерседес', 'Фольксваген', 'Опель')
print('Количество автомобилей в салоне:', len(car), 'это', car)
print('В салон привезли ещё несколько японских автомобилей')
all_car='Хонда', 'Мицубиси', car
print('Теперь количество автомобилей в салоне :', len(all_car) -1
+len(all_car[2]))
print('Все марки автомобилей в салоне:', all_car)
print('Немецкие автомобили :', all_car[2])
print('Последний немецкий автомобиль:', all_car[2][2])
```

В результате выполнения этой программы на экран будет выведено:
Количество автомобилей в салоне: 3 это ('Мерседес', 'Фольксваген', 'Опель')

В салон привезли ещё несколько японских автомобилей
Теперь количество автомобилей в салоне : 5
Все марки автомобилей в салоне: ('Хонда', 'Мицубиси', ('Мерседес', 'Фольксваген', 'Опель'))
Немецкие автомобили : ('Мерседес', 'Фольксваген', 'Опель')
Последний немецкий автомобиль: Опель

Поясним как работает эта программа. Переменная car обозначает кортеж элементов, каждый из которых является строкой с названием марки автомобиля. Функция len() позволяет получить длину кортежа и эта информация вместе с перечнем имеющихся в салоне автомобилей выводится на экран.

Далее создаётся новый кортеж `all_car`, содержащий в качестве последнего элемента кортеж `car`. Обратите внимание, что при создании этого кортежа круглые скобки не использовались.

Затем вычисляется и выводится на экран общее количество автомобилей в салоне. Заметим, что доступ к кортежу `car` осуществляется по индексу (по его номеру в кортеже).

Далее выводится на экран содержимое нового кортежа, а потом содержимое последнего его элемента (старого кортежа).

В конце программы с помощью двойного индекса выводится последний элемент старого кортежа с названиями немецких автомобилей.

Итак, доступ к элементам кортежа осуществляется указанием позиции элемента, заключённой в квадратные скобки, аналогично доступу к элементам списка.

Хотя при создании кортежа круглые скобки не являются обязательными, лучше всегда их указывать. Явное использование круглых скобок позволит избежать неоднозначных случаев. Например:

```
print(1, 2, 3)
print((1, 2, 3))
```

Эти функции выполняют различные действия: первая функция `print()` выводит три числа, а вторая – кортеж, содержащий эти три числа.

Кортеж, не содержащий элементов (пустой) или кортеж, содержащий 1 элемент, создаётся с помощью обязательных круглых скобок. Причём при создании одноэлементного кортежа в скобках должна присутствовать запятая «`,`». Например, пустой кортеж создаётся следующим образом:

```
Empty =()
```

А кортеж с одним элементом создаётся инструкцией:

```
Once =(5,)
```

Так необходимо поступать, чтобы Python мог отличить кортеж от скобок, окружающих объект в выражении.

Списки, кортежи и строки являются примерами последовательностей. Для последовательностей реализуются некоторые специфические для этих структур возможности. Например, проверка при-

надлежности (операторы “in” и “not in”) и оператор индексирования (квадратные скобки), позволяющий получить необходимый элемент последовательности.

Словарь (dict) похож на телефонную или адресную книгу, т.е. в нём есть ключи (фамилии абонентов), связанные с некоторой информацией (номерах их телефонов или адресами). В словарях в качестве ключей могут использоваться только неизменяемые объекты (такие, как строки), а в качестве значений можно использовать как неизменяемые, так и изменяемые объекты.

Пары ключ-значение указываются в словаре следующим образом:

```
My_dict = {key1 : value1, key2 : value2}
```

Ключ и значение разделяются двоеточием «:», а пары друг от друга отделяются запятой «,», и все пары заключаются в фигурные скобки «{ }».

Пример 3. Программа обрабатывает адресную книгу, содержащую фамилию и электронный адрес абонента.

```
# Создание словаря
adress_book = {
    'Иванов' : 'ivan123@gmail.com',
    'Николаев' : 'nick_w6@yandex.ru',
    'Петров' : 'xxx777@mail.ru',
    'Сидоров' : 'summer2020@hotmail.com'
}

# Вывод адреса на экран
print(«Адрес Иванова:», adress_book['Иванов'])

# Вывод на экран всех адресов
print('В адресной книге {0} контакта'.format(len(adress_book)))
for name, email in adress_book.items():
    print('Контакт {0} с адресом {1}'.format(name, email))

# Удаление записи из словаря
del adress_book['Николаев']
print('После удаления в адресной книге {0} контакта'.format(len(adress_book)))
for name, email in adress_book.items():
```

```

    print('Контакт {0} с адресом {1}'.format(name, email))
# Добавление записи
adress_book['Сергеев'] = 'serj2021@python.org'
print('После добавления контакта в адресной книге {0} контак-
тов'.format(len(adress_book)))
for name, email in adress_book.items():
    print('Контакт {0} с адресом {1}'.format(name, email))
# Проверка наличия адреса
if 'Сергеев' in adress_book:
    print(«\nАдрес Сергеева:», adress_book['Сергеев'])
else:
    print('Адрес Сергеева не найден')

```

В результате работы программы на экран будет выведено:

Адрес Иванова: ivan123@gmail.com

В адресной книге 4 контакта

Контакт Иванов с адресом ivan123@gmail.com

Контакт Николаев с адресом nick_w6@yandex.ru

Контакт Петров с адресом xxx777@mail.ru

Контакт Сидоров с адресом summer2020@hotmail.com

После удаления в адресной книге 3 контакта

Контакт Иванов с адресом ivan123@gmail.com

Контакт Петров с адресом xxx777@mail.ru

Контакт Сидоров с адресом summer2020@hotmail.com

После добавления контакта в адресной книге 4 контактов

Контакт Иванов с адресом ivan123@gmail.com

Контакт Петров с адресом xxx777@mail.ru

Контакт Сидоров с адресом summer2020@hotmail.com

Контакт Сергеев с адресом serj2021@python.org

Адрес Сергеева: serj2021@python.org

Поясним как работает эта программа. В начале программы после соответствующего комментария создаётся словарь с именем `adress_book` и заполняется четырьмя парами ключ-значение. Затем выводится на экран один из адресов.

Далее с помощью цикла на экран выводятся фамилии и адреса всех абонентов. Обратите внимание, как работает этот цикл. Используя метод `items()` элементы пары ключ-значение присваиваются двум переменным `name` и `email`, перебирая все такие пары. Затем в теле цикла осуществляется форматированный вывод на экран.

В следующем фрагменте программы в адресную книгу добавляется информация о новом абоненте и для контроля вновь выводится на экран.

В последнем фрагменте программы с помощью оператора `in` в адресной книге проверяется наличие фамилии определённого абонента и, если фамилия обнаружена, выводится на экран его электронный адрес.

Множество (`set`) – это неупорядоченный набор простых объектов. Они необходимы в том случае, когда присутствие объекта в наборе важнее порядка или того, сколько раз данный объект там встречается. Используя множества, можно осуществлять проверку принадлежности, определять, является ли данное множество подмножеством другого множества, находить пересечения множеств и так далее.

Для создания множества используется функция `set()`. В скобках в качестве будущих элементов множества перечисляются необходимые данные.

Пример 4. Программа обрабатывает список стран.

```
# Создание множества
countries = set(['Бразилия', 'Россия', 'Индия'])
print(countries, '\n')

# Проверка наличия элемента во множестве
if 'Индия' in countries:
    print('Индия присутствует в перечне стран')
else:
    print('Индия не найдена в перечне стран')
if 'США' in countries:
    print('США присутствуют в перечне стран')
else:
    print('США не найдены в перечне стран')
```

```

# Копирование множества
countries_new = countries.copy()
print('\nСкопированное множество:')
print(countries_new)

# Добавление элемента
countries_new.add('Китай')
print('\nНовое множество с добавленным элементом')
print(countries_new, '\n')

# Является ли множество подмножеством другого
if countries_new.issuperset(countries):
    print('Старое множество является подмножеством нового')
else:
    print('Нет, это не подмножество')

# Удаление элемента
countries.remove('Россия')
print('\nСтарое множество с удаленным элементом')
print(countries)

print('\nПересечение старого и нового множеств:')
print(countries.intersection(countries_new))

```

В результате работы программы на экран будет выведено:

```
{'Индия', 'Бразилия', 'Россия'}
```

Индия присутствует в перечне стран

США не найдены в перечне стран

Скопированное множество:

```
{'Индия', 'Бразилия', 'Россия'}
```

Новое множество с добавленным элементом

```
{'Индия', 'Бразилия', 'Россия', 'Китай'}
```

Старое множество является подмножеством нового

Старое множество с удаленным элементом

```
{'Индия', 'Бразилия'}
```

Пересечение старого и нового множеств:

```
{'Индия', 'Бразилия'}
```

Этот пример содержит необходимые комментарии и достаточно понятен, разобраться в его работе вы сможете самостоятельно.

В конце параграфа приведём несколько общих замечаний.

После создания объекта и присваивания его какой-либо переменной, эта переменная будет только ссылаться на созданный объект, а не представлять собой сам этот объект. Т.е. имя переменной указывает на ту часть памяти компьютера, где хранится объект. Это называется привязкой имени к объекту.

Пример 5. Программа создаёт список фруктов и демонстрирует различные способы присваивания его другой переменной.

```
# Создание списка
fruits=['яблоки', 'манго', 'морковь', 'бананы']
print('Простое присваивание')
my_fruits=fruits
print('fruits:', fruits)
print('my_fruits:', my_fruits)
# my_fruits является ещё одним именем, указывающим на тот же
объект
print('Удаление первого элемента списка fruits')
del fruits[0]
print('fruits:', fruits)
print('my_fruits:', my_fruits)
# Теперь вместо присваивания копируем с помощью среза
print('Копирование полного среза')
my_fruits=fruits[:]
print('fruits:', fruits)
print('my_fruits:', my_fruits)
print('Удаление второго элемента в списке my_fruits')
del my_fruits[1]
print('fruits:', fruits)
print('my_fruits:', my_fruits)
```

В результате работы этой программы на экран будет выведено:

Простое присваивание

fruits: ['яблоки', 'манго', 'морковь', 'бананы']

```
my_fruits: ['яблоки', 'манго', 'морковь', 'бананы']
```

Удаление первого элемента списка fruits

```
fruits: ['манго', 'морковь', 'бананы']
```

```
my_fruits: ['манго', 'морковь', 'бананы']
```

Копирование полного среза

```
fruits: ['манго', 'морковь', 'бананы']
```

```
my_fruits: ['манго', 'морковь', 'бананы']
```

Удаление второго элемента в списке my_fruits

```
fruits: ['манго', 'морковь', 'бананы']
```

```
my_fruits: ['манго', 'бананы']
```

Обратите внимание, что и fruits, и my_fruits в результате удаления первого элемента после простого присваивания содержат один и тот же список без пункта «яблоко». Это демонстрирует тот факт, что обе переменные fruits и my_fruits указывают на один объект.

Таким образом, если нужно сделать копию списка или подобной последовательности, следует воспользоваться срезом.

Контрольные вопросы

1. Зачем нужны различные структуры данных?
2. Какие структуры данных языка Python вы знаете?
3. Как можно создать список?
4. Какие операции можно выполнять со списком?
5. Чем список отличается от строки?
6. Что собой представляют методы класса?
7. Как можно добавить элемент к списку?
8. Как можно создать кортеж?
9. Чем кортеж отличается от списка?
10. Как осуществляется доступ к элементам кортежа?
11. Расскажите что представляет собой словарь в языке Python.
12. Как создать словарь?
13. Чем характеризуется множество в языке Python?
14. Как можно создать множество?
15. Как следует создавать копию списка? Почему?

§5.2. Массивы и списки

Списки в языке Python являются аналогом массивов в других языках программирования, при этом списки обладают более широкими возможностями. Размер списков не ограничен, благодаря чему они могут увеличиваться и уменьшаться по мере необходимости в результате использования методов, определённых для списков:

- `append()` – добавление элемента в список;
- `pop()` – удаление элемента по его индексу (аналогично инструкции `del`).

Как правило для обработки и анализа данных в массиве используются циклы. Далее в этой главе будем употреблять термин «массив», чтобы обеспечить универсальность дальнейшего изложения материала.

Массив – это определенное количество элементов (переменных) одного типа, которые имеют общее имя, и у каждого элемента есть свой индекс (порядковый номер). Массивы бывают одномерными и многомерными. Размер массива ограничивается только объёмом рабочей памяти компьютера. В информатике массив называется одномерным, если для получения доступа к его элементам достаточно одной индексной переменной.

Получить доступ к каждому элементу массива можно через его индекс. Например, если есть массив с именем `x`, то его первому элементу (с индексом 0) можно присвоить значение 5 следующим образом: `x[0] = 5`.

Обратите внимание, что индексы элементов массива заключаются в квадратные скобки и начинаются с 0.

Наряду с конкретным значением (константой) в качестве индекса может быть использована переменная, например, при обработке массива поэлементно с помощью цикла `for` с параметром `i` допустимо присваивание `x[i] = i*i`. Такое присваивание в теле цикла поместит в каждый элемент массива квадрат его индекса.

При создании массива и заполнении его элементами числами используют метод `append()`, а для удаления элементов – метод `pop()`. Напомним, что имя метода отделяется от имени массива точкой.

Изучим приемы создания и заполнения массивов на примерах.

Пример 1. Программа запрашивает размер (количество элементов) массива и заполняет его числами, вводимыми с клавиатуры, а затем выводит на экран элементы массива в обратном порядке.

```
n=int(input('Введите число элементов массива '))
x=[]
print('Введите', n, 'чисел')
for i in range(n):
    x.append(int(input()))
print('Вот эти числа в обратном порядке')
for i in range(n -1, -1, -1):
    print(x[i])
```

Обратите внимание на следующие особенности работы данного примера:

1. Вторая строка `x=[]` говорит, что в программе будет использоваться массив (список) с именем `x`. Пустые квадратные скобки означают, что `x` – это массив (список).

2. Внутри круглых скобок метода `append` заключена вложенная конструкция функций для ввода чисел с клавиатуры.

3. В управляющей строке второго цикла функция `range()` использует три параметра, причем третий параметр (шаг) есть число отрицательное, что позволяет изменять счетчик цикла в обратном порядке от значения `n -1` до 0.

Пример 2. Программа запрашивает размер массива, заполняет его случайными целыми числами от -100 до 100, отыскивает максимальный элемент массива, выводит его на экран и удаляет этот элемент из массива.

```
import random
# Подключение модуля для генерации случайных чисел
n=int(input('Введите число элементов массива '))
x=[]
for i in range(n):
    x.append(random.randint(-100, 100))
    print(x[i], end = ' ')
# Заполнение массива случайными числами и вывод его на экран
```



```

mx_x = x[0]
index_mx_x = 0
for i in range(1, n):
    if mx_x < x[i]:
        mx_x = x[i]
        index_mx_x = i
# Отыскание максимального элемента и его индекса
print('\nМаксимальный элемент', mx_x, 'с индексом', index_mx_x)
x.pop(index_mx_x)
print('Теперь в массиве', len(x), 'элементов')
for i in range(len(x)):
    print(x[i], end = ' ')
# Вывод массива без максимального элемента

```

Если после запуска программы на запрос количества элементов в массиве ввести число 4, то вывод на экран может быть следующим:

```

Введите число элементов массива 4
Создан массив из 4 элементов
7 -17 5 -4
Максимальный элемент 7 с индексом 0
Теперь в массиве 3 элементов
-17 5 -4

```

Обратите внимание, что во втором цикле для определения числа шагов используется функция `len()`, возвращающая количество элементов массива после удаления максимального.

Перед следующим примером рассмотрим ещё одну весьма удобную особенность языка Python. В нём можно за одну инструкцию присваивания изменять значение сразу нескольких переменных. Делается это с помощью множественного присваивания:

```
a, b = 0, 1
```

Слева от знака «`=`» (присвоить) в множественном присваивании должны стоять через запятую имена переменных, а справа должны стоять произвольные выражения, разделённые запятыми. Главное, чтобы слева и справа от знака присваивания было одинаковое число элементов. Присваивание будет осуществляться по порядку: первой

переменной слева будет присвоено значение, стоящее первым справа, второй переменной – второе значение и т.д.

Множественное присваивание удобно использовать, когда нужно обменять значения двух переменных. Рассмотрим пример, использующий подобную конструкцию.

Пример 3. Программа запрашивает размер массива, заполняет его случайными целыми числами от -100 до 100 и упорядочивает его по возрастанию.

```
import random
n = int(input('Введите число элементов массива '))
mas = []
print('Не сортированный массив')
for i in range(n):
    mas.append(random.randint(1, 100))
    print(mas[i], end = ' ')
for i in range(n):
    for j in range(1, n):
        if mas[j - 1] > mas[j]:
            mas[j - 1], mas[j] = mas[j], mas[j - 1]
print('\nУпорядоченный массив')
for i in range(n):
    print(mas[i], end = ' ')
```

Если после запуска программы на запрос количества элементов в массиве ввести число 5, то вывод на экран может быть следующим:

```
Введите число элементов массива 5
Не сортированный массив
38 89 41 81 94
Упорядоченный массив
38 41 81 89 94
```

В этом примере для упорядочивания (сортировки) массива используется алгоритм, обладающий собственным именем – «Пузырьковый алгоритм». Он состоит в том, что все элементы массива перебираются столько раз, сколько их в массиве, и на каждом проходе проверяются пары идущих подряд элементов. Если в проверяемой

паре элементы стоят правильно, т.е. предыдущий элемент меньше или равен последующему, то алгоритм переходит к проверке следующей пары. Если же предыдущий элемент больше последующего, то они меняются местами. Таким образом, после многократного прохождения цикла все его элементы будут расположены по возрастанию.

Пример 4. Программа выполняет сортировку элементов массива по возрастанию, как и в предыдущем примере, но вместо «Пузырькового алгоритма» используется метод `sort()`.

```
print('Введите число элементов')
n=int(input())
mas =[1]*n
# Создание массива из n элементов
print('Введите', n, 'чисел')
for i in range(n):
    mas[i] =int(input())
# Сортировка элементов массива
mas.sort()
print('Упорядоченный массив')
for i in range(n):
    print(mas[i], end = ' ')
```

После ввода с клавиатуры пяти чисел в произвольном порядке программа выведет на экран:

```
Введите число элементов 5
Введите 5 чисел
3
5
4
2
1
Упорядоченный массив
1 2 3 4 5
```

Обратите внимание, как в этом примере создаётся и заполняется данными массив `mas`. Присваивание `mas =[1]*n` создаёт массив из `n` элементов, каждый из которых равен 1.

основная часть программы, сортирующая элементы массива, здесь состоит всего из одной строки. В этой строке используется метод `sort()` для сортировки элементов.

Метод `sort()`, использованный в этом примере, по умолчанию упорядочивает элементы списка по возрастанию, а метод `reverse()` - по убыванию. В обоих случаях происходит непосредственное изменение самого списка.

Приведём ещё один пример обработки массивов.

Пример 5. Программа создаёт десятиэлементный массив из случайных чисел от -100 до 100. Затем запрашивает контрольное число и проверяет его наличие в массиве выводя на экран соответствующее сообщение.

```
import random
answer = 'Контрольное число не обнаружено'
x = []
print('Создание десятиэлементного массива')
for i in range(10):
    x.append(random.randint(-100, 100))
    print(x[i], end = ' ')
c = int(input('\nВведите контрольное число: '))
for i in range(10):
    if c == x[i]:
        answer = 'Контрольное число присутствует в массиве'
print(answer)
```

В результате работы этой программы на экран может быть выведено:

Создание десятиэлементного массива

29 12 14 47 30 -33 73 -44 85 -76

Введите контрольное число: 3

Контрольное число не обнаружено

Самостоятельно подробно разберите как работает эта программа.

В заключении параграфа заметим, что элементами массивов могут быть данные и других типов, например, строки. Сама строка так-

же может быть рассмотрена как массив символов, из которых она образована.

Хотя списки не имеют фиксированного размера, язык Python, тем не менее, не допускает возможности обращаться к несуществующим элементам списка. Обращение к элементам списка по индексам, значения которых выходят за пределы списка, всегда является ошибкой.

Контрольные вопросы

1. Что такое массивы?
2. Что называют элементом массива?
3. Что называют индексом элемента массива?
4. Что значит «одномерный массив»?
5. Сколько индексов может быть у одного элемента массива?
6. Сколько элементов может содержать массив?
7. Какие методы имеет массив?
8. Расскажите как можно заполнить числами массив.
9. Зачем используется метод `append`?
10. Расскажите как работает «Пузырьковый алгоритм».
11. Расскажите как работают программы, приведённые в каждом из примеров этого параграфа.

§5.3. Реализация рекурсивных алгоритмов

В начале параграфа будет рассмотрена процедура создания собственных функций, т.е. функций, которые разрабатывает сам программист. По способу использования они ничем не отличаются от изученных ранее встроенных в Python функций, например, таких, как `print()`, `input()` или `len()`.

Предположим, что некоторый алгоритм требует несколько раз вычислять наибольшее (максимальное) из трёх чисел. В этом случае программа, реализующая этот алгоритм, должна содержать соответствующий фрагмент кода для определения наибольшего числа. Приведём пример программы, содержащей подобный фрагмент.

Пример 1. Программа вычисляет наибольшее из трёх введенных целых чисел и выводит его на экран.

```
a = int(input('Введите первое число '))
```

```
b=int(input('Введите второе число '))
c=int(input('Введите третье число '))
if a >=b and a >=c:
    maximum =a
if b >=a and b >=c:
    maximum =b
if c >=b and c >=a:
    maximum =c
print('Максимальное число', maximum)
```

В этом примере программа запрашивает три целых числа и находит максимальное из них, но даже если предположить, что при реализации упомянутого выше алгоритма нет необходимости каждый раз запрашивать эти числа (они будут вычисляться самой программой), то многократно включать в текст программы фрагмент из трёх инструкций ветвления достаточно трудоёмко. В этом случае можно воспользоваться операцией копирования фрагмента текста в текстовом редакторе. Однако, если данный фрагмент содержал ошибку, то при копировании она также продублируется. К тому же, подобное копирование фрагментов программного кода существенно увеличивает размер самой программы и ухудшает её читаемость.

Чтобы избежать многократного использования в тексте программы одного и того же фрагмента кода, в языках программирования существуют функции. Функции – это такие участки кода, которые изолированы от остальной программы и выполняются только тогда, когда вызываются.

Функции обладают некоторыми общими свойствами:

1. Функции принимают один или несколько аргументов (в некоторых случаях функции не имеют аргументов).
2. Функции возвращают одно или несколько значений (в некоторых случаях функции не возвращают значение).

Например, функция `len()` принимает один аргумент (строку) и возвращает одно значение (её длину). Функция `print()` принимает переменное число аргументов и ничего не возвращает, а в качестве результата своей работы выводит значение аргументов на экран.

Рассмотрим функцию, которая принимает три числа и возвращает максимальное из них.

Пример 2. Функция с тремя числовыми аргументами, возвращающая максимальное значение из этих трёх чисел.

```
def maximum(a, b, c):  
    if a >=b and a >=c:  
        return a  
    if b >=a and b >=c:  
        return b  
    if c >=b and c >=a:  
        return c
```

Обратите внимание на следующие правила оформления функций, разрабатываемых программистом:

1. Функция начинается с ключевого слова `def` (definition – определение), после которого идёт имя функции с перечисленными в круглых скобках переменными, являющимися аргументами. В конце строки ставится двоеточие «:».

2. Тело функции оформляется в виде блока, т.е. пишется с отступа (4 пробела). В случае вложенных конструкций отступ соответствующим образом увеличивается.

3. Для возвращения значения используется ключевое слово `return` с указанием переменной (или конкретной константы), содержащей возвращаемое значение. Если функция ничего не возвращает, то ключевое слово `return` отсутствует.

Для того, чтобы продемонстрировать работу описанной выше функции её следует включить в законченную программу.

Пример 3. Программа вычисляет наибольшее из трёх введенных целых чисел и выводит его на экран (аналогично примеру 1, но с использованием функции).

```
def maximum(a, b, c):  
    if a >=b and a >=c:  
        return a  
    if b >=a and b >=c:  
        return b
```

```
if c >=b and c >=a:
```

```
    return c
```

```
a=int(input('Введите первое число '))
```

```
b=int(input('Введите второе число '))
```

```
c=int(input('Введите третье число '))
```

```
print('Максимальное число', maximum(a, b, c))
```

В этом примере основная программа отделена от функции пустой строкой. Обратите внимание, что описание функции должно идти до её использования в коде основной программы.

Заметим, что в языке программирования Python уже встроена функция `max()` для определения максимума из любого количества чисел, т.е. эта функция с переменным числом аргументов.

Приведем ещё один пример создания функций.

Пример 4. Программа запрашивает натуральное число и выводит его факториал на экран.

```
def factorial(x):
```

```
    res =1
```

```
    for i in range(2, x +1):
```

```
        res *=i
```

```
    return res
```

```
n=int(input('Введите натуральное число '))
```

```
print('Факториал числа', n, 'равен', factorial(n))
```

С помощью инструкции `return` функция может возвращать несколько значений, при этом возвращаемые значения заключаются в квадратные скобки и разделяются запятой, т.е. оформляются в виде списка:

```
Return [a, b, c]
```

В этом случае результат вызова функции можно будет использовать во множественном присваивании:

```
K, l, m =f()
```

Предположим, что требуется прочитать некоторый текст, содержащий незнакомые слова. Для выяснения значения этих незнакомых слов необходимо пользоваться толковым словарем. Когда в тексте встречается незнакомое слово, то его объяснение ищется в словаре.

В объяснении слова могут, в свою очередь, встретиться незнакомые слова, которые также нужно будет найти в словаре и так далее.

Такую задачу можно решить с помощью следующего алгоритма:

1. Если встретилось незнакомое слово, Найдите его в словаре.
2. Прочитайте статью, объясняющую значение этого слова.
3. Если объяснение понятно, то есть статья не содержит слов, вызывающих затруднение, вернитесь в предыдущий текст и продолжите чтение с места, где было прервано чтение.
4. Если в объяснении встречается незнакомое слово, то прекратите его чтение, запомните место прекращения и выясните значение слова, придерживаясь правил 1 - 4.

Такой алгоритм называют рекурсивным, т.е. использующим себя. В рекурсивном алгоритме содержится ссылка на самого себя.

Обратите внимание на следующие две важнейшие особенности рекурсивного алгоритма:

1. Рекурсивный алгоритм должен содержать, по крайней мере одну, терминальную ветвь, т.е. условие окончания алгоритма (пункт 3 примера, приведённого выше).

2. Когда алгоритм доходит до рекурсивной ветви, т.е. обращения к самому себе (пункт 4 из примера, приведённого выше), то процесс выполнения приостанавливается, и запускается новый аналогичный алгоритм, но уже на следующем уровне. Прерванный же алгоритм запоминается. Он будет ждать и начнет исполнение лишь по окончании более высокого по уровню алгоритма. В свою очередь новый алгоритм может приостановиться, войти в режим ожидания и т.д. Таким образом образуется цепочка прерванных процессов, из которых выполняется лишь последний в настоящий момент времени процесс, а по окончании его работы продолжает выполняться предшествовавший ему.

Рекурсия – это такой способ организации вычислительного процесса, при котором функция в ходе выполнения составляющих ее инструкций обращается сама к себе.

При каждом очередном входе в рекурсивную функцию ее локальные параметры размещаются в особом образом организованной области памяти – стеке. Целиком весь процесс выполнен, когда стек

опустеет или, другими словами, все прерванные процессы выполнятся.

При использовании рекурсии необходимо обращать особое внимание на условие выхода из рекурсивной функции в нужный момент.

Ещё одним примером рекурсивного алгоритма может служить алгоритм поиска файлов с заданным расширением находящихся в заданном каталоге и всех его подкаталогах:

1. Вывести список всех файлов, находящихся в текущем каталоге и имеющих заданное расширение.
2. Если в текущем каталоге есть вложенные подкаталоги, то обработать каждый из них.

Приведенный алгоритм является рекурсивным, поскольку для обработки подкаталога алгоритм вызывает сам себя.

Рекурсия полезна, прежде всего, в случаях, когда основную задачу можно разделить на подзадачи, имеющие ту же структуру, что и первоначальная задача. Функции, реализующие рекурсивный алгоритм, называются рекурсивными.

Приведём несколько примеров реализации рекурсивных алгоритмов на языке Python.

Пример 1. Программа запрашивает натуральное число и используя рекурсивную функцию вычисляет сумму всех натуральных чисел, не превосходящих введенного.

```
def sum_n(n):  
    if n == 1:  
        return 1  
    else:  
        return sum_n(n - 1) + n  
  
k = int(input('Введите натуральное число '))  
print('Сумма натуральных чисел равна', sum_n(k))
```

Действительно, сумма первых n натуральных чисел равна сумме первых $(n - 1)$ натуральных чисел плюс n , а при $n = 1$ сумма равна 1. Условие $n == 1$ представляет собой условие выхода из рекурсии.

Пример 2. Программа запрашивает натуральное число и, используя рекурсивную функцию, вычисляет его факториал.

```
def factorial(n):
    if n == 1:
        return 1
    else:
        return factorial(n - 1) * n

k = int(input('Введите натуральное число '))
print('Факториал числа', k, 'равен', factorial(k))
```

Действительно, факториал числа n равен произведению n на факториал числа $(n - 1)$. В свою очередь, факториал числа $(n - 1)$ равен произведению $(n - 1)$ на факториал числа $(n - 2)$ и т.д.

Обратите внимание, что функция вызывает сама себя только в том случае, если значение полученного аргумента не равно единице. Если значение аргумента равно единице, то функция себя не вызывает, а возвращает значение, и рекурсивный процесс завершается.

Не следует путать рекурсию с итерацией – повторяющимся выполнением определённой последовательности операций до тех пор, пока не будет удовлетворяться некоторое условие. Цикл с проверкой условия реализует итерационный алгоритм. Большинство алгоритмов можно реализовать двумя способами: итерацией и рекурсией.

Обратите внимание, что использование рекурсивной формы организации алгоритма обычно выглядит изящнее итерационной и дает более компактный текст программы. Это является основным достоинством рекурсивной формы реализации алгоритма.

Однако, у рекурсии есть и существенные недостатки:

1. Если глубина рекурсии очень велика, то программа будет требовать во время выполнения много памяти – это может привести к переполнению стека.

2. Рекурсивные алгоритмы, как правило, выполняются более медленно.

3. При программировании рекурсивных алгоритмов велика вероятность ошибок, способных вынудить программиста к перезагрузке компьютера.

Таким образом, если задача имеет очевидное не рекурсивное решение, то следует выбрать именно его. Рекурсивные функции являют-

ся мощным механизмом в программировании, однако, они не всегда эффективны. Часто использование рекурсии приводит к ошибкам. Наиболее распространенная из таких ошибок состоит в том, что рекурсия никогда не заканчивается, т.е. цепочка вызовов рекурсивной функции никогда не завершается и продолжается, пока не закончится свободная память в компьютере.

Укажем две наиболее распространенные ошибки, приводящие к бесконечной рекурсии:

1. Неправильное оформление выхода из рекурсии. Например, если в рекурсивной программе вычисления факториала не организовать проверку $n == 1$, то функция `factorial(1)` вызовет функцию `factorial(0)`, далее будет вызвана функция `factorial(-1)` и т.д. Таким образом, процесс никогда не завершится.

2. Рекурсивный вызов с неправильными аргументами. Например, если функция `factorial(n)` будет вызывать `factorial(n)`, то также получится бесконечная цепочка вызовов, поскольку значение аргумента не изменилось.

Итак, при разработке рекурсивной функции необходимо прежде всего предусматривать условие завершения рекурсивного процесса и следить за тем, чтобы это условие когда-либо выполнялось и процесс завершался.

Приведём ещё один пример рекурсивного решения задачи вычисления членов последовательности чисел Фибоначчи. Эта последовательность была впервые предложена В 1202 году итальянским математиком Леонардом Пизанским, известным под именем Фибоначчи. Напомним, что первый член последовательности равен 1, второй равен 1, а каждый последующий равен сумме двух предыдущих:

1, 1, 2, 3, 5, 8, 13, 21, 34, ...

Числа Фибоначчи вычисляются так:

$$f(1) = 1$$

$$f(2) = 1$$

$$f(n) = f(n - 1) + f(n - 2)$$

Это позволяет построить рекурсивный алгоритм вычисления чисел Фибоначчи.

Пример 3. Рекурсивная программа вычисления числа Фибоначчи по его номеру.

```
def fib(n):  
    if n == 1 or n == 2:  
        return 1  
    else:  
        return fib(n - 1) + fib(n - 2)  
  
k = int(input('Введите номер числа Фибоначчи '))  
print('Число Фибоначчи с номером', k, 'равно', fib(k))
```

Контрольные вопросы

1. Что такое функция в программировании?
2. Какие встроенные функции языка Python вам известны?
3. Расскажите, как следует создавать собственную функцию в языке Python.
4. Могут ли в описании функции встречаться вложенные блоки инструкций?
5. Может ли инструкция return встречаться в описании функции несколько раз?
6. Может ли инструкция return вернуть сразу несколько значений?
7. Что такое рекурсия?
8. Какую функцию называют рекурсивной?
9. Что такое итерация?
10. Какие преимущества имеет рекурсия?
11. Какие недостатки имеет рекурсия?
12. Что следует обязательно предусматривать при разработке рекурсивных функций?
13. Приведите примеры рекурсивных алгоритмов.

§5.4. Работа с файлами

Рассмотренные в первом параграфе этой главы структуры данных способны хранить информацию только при работающей программе, в которой они были созданы. После завершения программы все списки, кортежи, словари и множества прекратят своё существование и будут созданы вновь только при следующем её запуске. Чтобы

сохранить информацию после завершения программы и даже после выключения компьютера её записывают в файлы. При следующем запуске программа может прочитать информацию из файла и создать необходимую структуру данных, содержащую эту информацию. В этом параграфе будут коротко рассмотрены основные способы работы с файлами на языке Python.

Работа с файлом в Python состоит из трёх этапов:

- Открытие файла;
- Запись/чтение данных;
- Заккрытие файла.

Для открытия файла используется функция `open()`. Эта функция возвращает объект класса `io.TextIOWrapper`, имеющий несколько полезных методов. У функции `open()` достаточно много аргументов, но чаще всего используются два:

- Имя файла;
- Режим открытия файла.

Пример открытия файла:

```
f=open('text.txt', 'r')
```

Первый параметр функции `open()` является строкой, содержащей имя открываемого файла. Если в этой строке путь к файлу не указан, то поиск файла будет осуществляться в той папке, в которой была запущена программа.

Вторым аргументом является режим открытия файла. Он может быть следующим:

- `'r'` - открытие файла на чтение (этот режим является значением по умолчанию);
- `'w'` - открытие файла на запись, при этом содержимое файла удаляется, если файла не существует, то создается новый;
- `'a'` - открытие файла на добавление данных (информация добавляется в конец файла);
- `'b'` - открытие файла в двоичном режиме;
- `'t'` - открытие файла в текстовом режиме (также является значением по умолчанию);
- `'+'` - открытие файла на чтение и запись (используется в сочетаниях `'w+'` или `'r+'`).

Режимы могут быть объединены, например, 'rb' – чтение в двоичном режиме. По умолчанию установлен режим 'rt' – чтение в текстовом режиме.

Для чтения информации из файла можно использовать несколько способов. Первый из них состоит в применении метода read(), читающего весь файл целиком, если у него будут отсутствовать аргументы. Если метод read() вызвать с целым числом n в качестве аргумента, то будут прочитаны n символов. Например, следующий фрагмент программы будет читать из файла и выводить на экран один символ:

```
f=open('text.txt')
ch=f.read(1)
print(ch)
```

Обратите внимание, что если метод read() вызывать с аргументом 1 несколько раз, то внутренний указатель будет сдвигаться по файлу каждый раз на один символ и будут читаться идущие подряд символы. Таким образом можно прочитать посимвольно весь файл.

Другой способ заключается в чтении файла по строкам. Для этого служит метод readlines(). При использовании readlines() внутренний указатель в файле сдвигается по строкам. Например, следующий фрагмент программы прочитает одну строку из файла и выведет её на экран:

```
f=open('text.txt')
line=f.readlines()
print(line)
```

Заключив метод readlines() в цикл, можно прочитать весь файл и записать его строки в список для последующей обработки.

Python даёт ещё один способ построчного чтения файла с помощью цикла for. Например, следующий фрагмент программы прочитает построчно весь файл и выведет текст на экран:

```
f=open('text.txt')
for line in f:
    print(line)
```

Теперь рассмотрим способы записи информации в файл. Аналогично чтению есть два метода для записи текста в файл. Запись может осуществляться с помощью метода write(). Этот метод будет

возвращать число записанных в файл символов. В качестве аргумента метода `write()` могут использоваться только строки. Если необходимо записать в файл числовую информацию, то ее необходимо предварительно привести к строковому типу, используя функцию `str()`. Например, следующий фрагмент программы записывает в файл одну строку:

```
f=open('text.txt', 'w')
f.write('Это строка для записи')
```

Обратите внимание, что в этом примере файл был открыт для записи (режим `'w'`).

Второй способ записи реализуется с помощью метода `writelines()`, записывающего текст по строкам. Метод `writelines()` может записывать в файл не только строки, но и другие структуры данных. Например, следующий фрагмент программы записывает в файл список `city_list`:

```
city_list=['Москва', 'Волгоград', 'Новосибирск', 'Саратов']
f=open('city.txt', 'w')
f.writelines(city_list)
```

Методы `write()` и `writelines()` автоматически не ставят символ переноса строки, и это программисту нужно контролировать самостоятельно. Так, приведённый выше фрагмент программы, записывающий список `city_list`, поместит в файл перечень городов в одну строку без пробелов между ними:

```
МоскваВолгоградНовосибирскСаратов
```

Обратите внимание, что все приведённые выше в этом параграфе примеры являются лишь фрагментами, а не законченными программами! В них не хватает последнего этапа работы с файлами! После окончания работы с файлом его следует закрыть с помощью метода `close()`. Например:

```
f.close()
```

В конце параграфа приведём несколько примеров завершённых программ работы с файлами.

Пример 1. Программа записывает в файл список городов, но так, чтобы название каждого города начиналось с новой строки. Затем файл прочитывается и его содержимое выводится на экран.


```

# Создание списка
city_list = [
    'Москва',
    'Волгоград',
    'Новосибирск',
    'Саратов']
# Открытие файла
f = open('city.txt', 'w+')
# Запись списка в файл
for line in city_list:
    f.write(line + '\n')
# Вывод списка из файла на экран
print('В файле записаны города:')
f.seek(0)
for line in f.readlines():
    print(line.strip())
# Заккрытие файла
f.close()

```

После выполнения этой программы на экран будет выведено:

В файле записаны города:

Москва

Волгоград

Новосибирск

Саратов

Обратите внимание, что перед выводом списка городов на экран внутренний указатель в файле смещается к началу с помощью метода seek(). Если этого не сделать, то к моменту чтения из файла внутренний указатель будет находиться в конце и чтения не произойдёт.

Пример 2. Программа для создания англо-русского словаря.

```

# Создание пустого словаря
words = {}
# Чтение ранее сохранённого словаря из файла
f = open('dictionary.txt', 'r')
old_words = f.readlines()

```

```

f.close()
for line in old_words:
    words[line.split(' ')[0]] = line.split(' ')[1]
# Вывод на экран ранее сохранённого словаря
print('В словаре {0} словарных записей'.format(len(words)))
for eng_word, rus_word in words.items():
    print('{0} - {1}'.format(eng_word, rus_word), end = '')
# Добавление новых словарных статей
p = 'y'
print('Добавьте в словарь записи')
while p == 'y':
    eng_word = input('Введите английское слово')
    rus_word = input('Введите соответствующее русское слово')
    words[eng_word] = rus_word
    p = input('Хотите ввести ещё одну словарную статью (y/n)?')
# Вывод на экран дополненного словаря
print('Теперь в словаре {0} словарных записей'.format(len(words)))
for eng_word, rus_word in words.items():
    print('{0} - {1}'.format(eng_word, rus_word), end = '')
# Запись словаря в файл
f = open('dictionary.txt', 'w')
for eng_word, rus_word in words.items():
    f.writelines(eng_word + ' ' + rus_word)
f.writelines('\n')
f.close()

```

Опираясь на комментарии в тексте программы, самостоятельно пошагово разберите её работу.

Контрольные вопросы

1. Что произойдёт с информацией, записанной в кортеж, после завершения программы? А с информацией, записанной в словарь?
2. Как сохранить информацию до следующего запуска программы?
3. Какие этапы необходимо выполнить при работе с файлами?
4. Какая функция открывает файл? Какие у неё аргументы?

5. Какие режимы открытия файлов существуют в Python?
6. Как можно прочитать информацию из файла?
7. Как можно записать информацию в файл?
8. Данные какого типа записывает метод `write()`? А метод `writelines()`?
9. Предложите способ, с помощью которого можно записываемую в файл информацию разделить на строки.
10. Какой метод служит для закрытия файла?
11. Расскажите о том, как работают приведённые в примерах 1 и 2 программы.

Глава 6

Файловые менеджеры и программы невизуального доступа к информации

§6.1. Операции над файлами и папками

Как вы уже знаете, основной единицей хранения информации на компьютере является файл. Исполняемые программы, текстовые документы, фотографии, музыкальные произведения и т.п. представляют собой файлы.

Файл – это поименованная область дискового пространства или другого носителя электронной информации. Каждый файл имеет имя, состоящее из двух частей: собственно названия (или просто имени) и расширения (обычно 3 или 4 символа). По расширению файла операционная система Windows определяет какой программой этот файл следует открывать (определяет его тип). Например, текстовый файл имеет расширение TXT и открывается редактором Блокнот, текстовый документ имеет расширение DOCX и открывается редактором (текстовым процессором) Word, а программные файлы имеют расширение EXE и выполняются (а не открываются) операционной системой.

Над файлом можно выполнять следующие операции:

- Копирование;
- перемещение;
- Удаление;
- переименование.

Также файл можно создавать. Операция создания строго говоря не является операцией над объектом, так как на момент начала этой операции объекта еще не существует, он возникает в результате ее выполнения.

Файл или несколько файлов можно поместить в некоторое хранилище. Это хранилище называется папкой или каталогом и имеет определенное имя. Сама папка не занимает места на диске, поэтому ее размер определяется суммарным размером содержащихся в ней

файлов. Над папками можно производить такие же операции, как и над файлами включая операцию создания.

Заметим, что операции, производимые над папкой, относятся к файлам, содержащимся в ней. Например, если удалить какую-либо папку, то вместе с ней удалятся и все файлы находящиеся в ней. Если же скопировать папку на другой носитель, то вместе с ней копируются все файлы, находящиеся в этой папке.

В одной папке не может быть двух файлов с одинаковыми именами. То есть, имена двух файлов, находящихся в одной папке, должны отличаться друг от друга хотя бы одним символом. В разных папках файлы с одинаковыми именами могут встречаться. Это могут быть копии одного файла, или одноименные файлы с разным содержанием.

Внутри папки могут находиться не только файлы, но и другие папки. Такие папки называются вложенными или подпапками.

Файлы и папки хранятся на носителях цифровой информации: жестких дисках (внутри компьютера), компакт дисках (оптических носителях информации), флешках и т.д. Папки имеют структуру, называемую «дерево». Причем, это «дерево» расположено «корнем» вверх. Самый верхний уровень, на котором могут располагаться папки и файлы, называется корневой папкой диска, или корневым каталогом. Здесь находятся самые большие папки-хранилища (самые толстые ветви дерева), в которые вложены все остальные папки этого диска. В корневой папке диска могут находиться и просто файлы. Чтобы определить, где находится файл, нужно указать путь к нему. Путь начинается с имени диска, а затем содержит последовательный перечень всех папок, в которые вложен этот файл.

Пусть, например, на диске D: (это раздел жёсткого диска внутри компьютера) находится папка «Учебная информация». В ней находится папка «Стихотворения». А в папке «Стихотворения» есть файл «Пушкин.docx». Тогда полный путь к этому файлу будет таким:

D:\Учебная информация\Стихотворения\Пушкин.docx

Путь к файлу позволяет быстро отыскать нужный объект. Например, указанный выше путь означает, что для отыскания файла Пушкин.docx, нужно сначала открыть локальный диск D: затем войти

папку «Учебная информация» и, наконец, войти в папку «Стихотворения», в которой будет находиться искомый файл.

Для выполнения операций над файлами и папками служат специальные программы- файловые менеджеры. В стандартный комплект приложений операционной системы Windows входит файловый менеджер Проводник. Для запуска Проводника можно воспользоваться клавиатурной командой Win +E или начать набирать слово «проводник» в строке поиска Главного Меню. Когда Проводник будет запущен программа невидимого доступа сообщит заголовок окна «Этот компьютер». Обратите внимание, что в заголовке окна нет названия программы Проводник. Вместо этого окно имеет заглавие, содержащее имя той папки, содержимое которой отображает в данный момент Проводник. Этим удобно пользоваться для того, чтобы определить имя отображаемой папки. Напомним, что заголовок окна озвучивается командой Ins +T.

Если файловый менеджер Проводник отображает элементы папки в виде списка, то они располагаются в несколько столбцов. При перемещении вниз и достижении конца первого столбца фокус перейдёт на первый элемент второго столбца. И так далее, пока не будет достигнут последний элемент списка.

В дальнейшем будем считать, что в Проводнике в качестве режима отображения папок установлен режим Таблица. По умолчанию в таблице присутствуют 4 столбца:

- Имя;
- Дата изменения;
- Тип;
- Размер.

При входе в папку фокус находится в первом столбце. Для получения информации из остальных столбцов, используйте стрелку вправо или для чтения сразу всей строки клавиатурную комбинацию Ins +8 (цифра 8 набирается на дополнительной клавиатуре).

Принципы выбора объекта в Проводнике аналогичны используемым для поиска объекта на Рабочем Столе. Можно перемещаться по элементам таблицы с помощью стрелок вертикального управления курсором, а можно использовать перемещение по первым бук-

вам названия объекта. Можно набрать несколько первых букв имени объекта при условии, что пауза между нажатиями не превысит секундного интервала. Если используется этот способ, то учитывайте, что при вводе букв имеет значение раскладка клавиатуры. JAWS при этом будет сообщать вводимые буквы и имя активного объекта.

Для входа в папку используется клавиша Enter, а для выхода – клавиша BackSpace.

Если запустив программу Проводник ввести команду Shift +Tab, то фокус окажется в области отображения, где содержимое диска представлено в виде Дерева. Структура Дерево отображает объекты согласно уровню иерархии. Чем больше уровень иерархии, тем больше смещение элемента вправо от левой границы области отображения.

При переходе в область отображения структуры «дерево» JAWS сообщит об этом, произнеся слово «дерево», а затем назовет имя выделенного объекта. Если выделенный объект является подпапкой, то после её имени JAWS добавит, либо «открыто», либо «закрыто», что соответствует отображению или не отображению содержимого этой подпапки. Для открытия подпапки используется клавиша Стрелка Вправо, а для закрытия – Стрелка Влево. В частности, аналогично будут открываться логические диски и внешние накопители информации.

Для перемещения по Дереву используются клавиши вертикального управления курсором.

При изменении уровня просматриваемого объекта JAWS сообщит номер уровня этого объекта. Если при перемещении фокуса сообщение о номере уровня не было, то текущий и предыдущий объекты находятся на одном уровне иерархии.

Рассмотрим алгоритм создания папки в программе Проводник:

1. Запустите Проводник сочетанием КЛАВИШ Win +E.
2. Выберите необходимый носитель информации (локальный диск или внешний накопитель).
3. Перейдите в папку, в которой необходимо создать новую папку.
4. Убедитесь, что никакой объект в этой папке не выделен. Обратите внимание, что если вы только вошли в папку и не совершали

никаких действий, то объект, на котором будет находиться фокус, не выделен. В противном случае снимите выделение с текущего объекта с помощью команды Ctrl +Пробел.

5. Откройте Контекстное Меню клавишей Application.

6. нажимая стрелку вниз Найдите подменю «создать». JAWS должен сообщить, что этот пункт содержит подменю. Войдите в него нажав Стрелку Вправо.

7. В раскрывшемся подменю Выберите команду «папку».

8. После нажатия клавиши Enter раскроется окно диалога создания папки, фокус будет находиться в поле редактирования, в которое следует ввести имя создаваемой папки. Обратите внимание на то, что в поле редактирования уже будет введено имя «Новая папка», причем это предложенное имя будет выделено. Можно нажав клавишу Delete удалить его, а можно просто начать ввод желаемого имени папки, при этом предложенное имя удалится.

9. После ввода нужного имени нажмите клавишу Enter. Фокус окажется на только что созданной папке.

В меню «создать» есть также команды для создания пустых документов разных типов. Это один из способов создания новых документов, который состоит из следующих этапов:

1. Запустите программу Проводник.
2. Перейдите в нужную папку.
3. Добейтесь того, чтобы никакой объект не был выделен.
4. В контекстном меню выберите подменю «Создать».
5. В раскрывшемся подменю выберите необходимый объект.
6. Введите имя создаваемого объекта и нажмите Enter.

Обратите внимание, что если вы создаете файл, то его расширение вводить не надо.

Для того, чтобы Скопировать или переместить объект, можно воспользоваться следующим алгоритмом:

1. Запустите программу Проводник.
2. Переместите фокус на нужный объект, при этом он станет выделен как активный. Если это первый объект в папке, то его потребуются выделить командой Ctrl +Пробел.

3. Введите команду Ctrl +C для копирования выделенного объекта (файла или папки) в буфер обмена или Ctrl +X для вырезания текущего объекта.

4. Перейдите в папку, в которую нужно поместить объект. Она может находиться и на другом носителе информации.

5. Введите команду Ctrl +V для вставки объекта из буфера обмена в текущую папку.

При копировании и перемещении объектов удобно запустить две копии программы Проводник, чтобы в первой из них открыть папку источник объектов, а во второй открыть папку назначения. Напомним, что переключаться между окнами (т.е. в данном случае между копиями Проводника) можно командой Alt +Tab.

Если при вставке файла в некоторую папку, там уже существует файл с таким именем, то будет выдано сообщение об этом. В появившемся диалоге необходимо выбрать заменять ли новым файлом старый или отказаться от копирования. Если копируется группа файлов, то в вариантах ответа добавится кнопка «Да для всех». Это позволяет ускорить процесс копирования и замены для всей группы файлов и избежать аналогичных вопросов для каждого файла из копируемой группы. Копирование папок происходит аналогично.

В операционной системе Windows существует универсальный буфер обмена информацией, в который можно при необходимости временно поместить (скопировать или вырезать) файлы или папки, чтобы потом их вставить в папку назначения. При выполнении рассмотренных выше операций копирования и перемещения объекты переносились через буфер обмена.

Если объекты помещаются в буфер обмена с помощью команды копирования, то после вставки этих объектов в другую папку, они размножаются, т.е. остаются в той папке, где были и появляются в той, куда их вставили. Причем, после вставки объекты остаются в буфере обмена, и их можно вставить еще много раз в другие папки.

Если файл или папка попадают в буфер обмена с помощью команды «Вырезать», то после их вставки они удаляются из первоначального места, т.е. вырезание с последующей вставкой – это перемещение файлов и папок из одной папки в другую.

Ранее помещенные в буфер объекты замещаются новыми только при следующем копировании или вырезании.

Отметим, что файлы и папки в буфер обмена может скопировать или вырезать одна программа, а выполнить вставку другая. Буфер используется для обмена информацией не только внутри одной программы, но и для обмена информацией между разными программами.

Для удаления объекта с помощью программы Проводник можно воспользоваться следующим алгоритмом:

1. Запустите программу Проводник.
2. Установите фокус на объект подлежащий удалению.
3. Нажмите клавишу Del для ввода команды «удалить».
4. Подтвердите своё решение удалить этот объект, нажатием кнопки «Да» в появившемся диалоге.

При удалении папки и файлы не сразу удаляются с носителя информации. Сначала они помещаются в особую папку, которая называется «Корзина». Из «Корзины» удаленные объекты можно восстановить.

Ярлык «Корзины» находится на «Рабочем Столе». Для возвращения ошибочно удаленных файлов откройте «Корзину» с помощью ярлыка на «Рабочем Столе» и в Контекстном Меню выберите команду «Восстановить». После чего выделенный объект будет восстановлен. Для окончательного удаления объектов снимите выделение с текущего объекта и в Контекстном Меню «Корзины» выберите команду «Очистить Корзину». После выполнения этой операции восстановить папки и файлы будет уже невозможно.

Рекомендуется периодически выполнять процедуру очистки «Корзины».

Переименование папки или файла удобнее всего осуществлять с помощью клавиатурной команды F2. Хотя есть и другие способы выполнения этой операции. После нажатия клавиши F2, JAWS сообщит, что фокус оказался в поле «Редактор» и добавит к этому текущее имя объекта. В это поле следует ввести новое имя объекта. Как и в случае ввода имён для создаваемых объектов, текущее имя будет выделено. Это означает, что ввод любого символа заменит выделенное имя введённым символом. Если вы хотите исправить один или

несколько символов, а не вводить имя заново целиком, нажимайте клавиши горизонтального управления курсором для снятия выделения и перемещения по текущему названию объекта. Подтвердите внесенные изменения нажатием клавиши Enter. Если вы хотите отказаться от изменения имени объекта, то воспользуйтесь клавишей Escape. В любом из двух случаев режим редактирования будет завершен и фокус вернется в список объектов папки.

Для изменения имени объекта в поле редактирования удобно использовать брайлевский дисплей. Найдите на дисплее символ, который необходимо изменить, и нажмите над ним кнопку перемещения курсора (Кнопку Роутинга). Текстовый курсор сразу окажется на нужном символе и его можно будет удалить, а затем ввести новый символ. Ввод символа можно осуществлять как с основной клавиатуры, так и с клавиатуры Перкинса на брайлевском дисплее.

В конце параграфа коротко рассмотрим алгоритмы выполнения операций над файлами и папками с помощью файлового менеджера Total Commander. В отличие от Проводника Total Commander не включен в число приложений, входящих в дистрибутив операционной системы Windows, т.е. его надо устанавливать отдельно. Процедура установки Total Commander здесь описываться не будет. Будем считать, что эта программа уже установлена на ваш компьютер и на «Рабочий Стол» помещен ярлык для ее запуска.

Total Commander является одним из самых популярных файловых менеджеров. Он позволяет выполнять все, рассмотренные выше, операции над файлами и папками, а также и многое другое.

Интерфейс Total Commander разделен по вертикали на две независимые части (панели), на которых отображается список папок и файлов какого-либо носителя. В самом верху окна Total Commander расположено строка меню, как и у большинства приложений Windows. Внизу окна программы есть «Командная Строка», но это средство для опытных пользователей и здесь работа с ней рассматриваться не будет.

Способ отображения папок и файлов на левой и правой панелях Total Commander можно настраивать независимо. Вариантов настройки здесь значительно больше, чем в Проводнике Windows. Рас-

смотрим только некоторые из возможных вариантов настройки режима представления папок и файлов.

Подробный – имена объектов расположены столбцом. После каждого имени объекта указан его тип, дата создания, размер и т.д. При перемещении стрелками вертикального управления курсором JAWS будет озвучивать всю эту информацию. Установить режим «Подробный» можно с помощью клавиатурной команды Ctrl +F2.

Краткий – в этом случае всё содержимое панели выводиться в несколько столбцов, в которых отображены только имена объектов без какой-либо дополнительной информации. Поскольку имена объектов представлены в виде списка, то при перемещении фокуса вниз, JAWS будет последовательно озвучивать все объекты автоматически переходя из столбца в следующий столбец. С точки зрения пользователя программ невизуального доступа имена объектов расположены в один столбец. Установить этот вид можно клавиатурной командой Ctrl +F1.

Комментарии – этот вид панели похож на режим Подробный, но здесь еще появится столбец с комментарием к каждому файлу или папке (если он имеется). Создать комментарий можно с помощью клавиатурной команды Ctrl +Z. Обратите внимание, что при вводе этой команды JAWS произнесет «отмена», поскольку реагирует на подаваемую клавиатурную команду, а не на ее выполнение. Установить режим «Комментарии» можно клавиатурной командой Ctrl +Shift +F2.

Дерево каталогов – объекты будут представлены в виде уже знакомой структуры Дерево. Этот режим включается командой Ctrl +F8.

Выбрать любой из этих режимов можно через меню Вид строки меню программы Total Commander. В этом меню представлены и другие режимы, их вы можете попробовать самостоятельно.

В меню Вид также можно выбрать режим сортировки объектов. Например, введя клавиатурную команду Ctrl +F4 можно установить режим сортировки По типу. В этом режиме в самом верху будут располагаться имена папок, а затем имена файлов, упорядоченные в алфавитном порядке по расширению. Т.е. сперва будут идти файлы с расширением com, затем с расширением docx, а затем exe. Если

файлы имеют одинаковое расширение, то они будут упорядочены в алфавитном порядке относительно имени.

Можно упорядочить файлы по имени. Для этого следует ввести клавиатурную команду Ctrl +F3. В этом случае приоритет в алфавитном порядке будет отдан имени файла. Если же имена будут одинаковые, то файлы упорядочатся относительно расширения.

Заметим, что устанавливаться режимы будут для той панели, на которой находился фокус в момент ввода соответствующей команды. Если повторить команду сортировки вторично, то порядок объектов изменится на обратный. Перемещать фокус с панели на панель можно клавишей Tab.

На каждой панели можно независимо выбрать содержимое какого-либо носителя. Для выбора носителя левой панели используйте клавиатурную команду Alt +F1, а для правой панели – Alt +F2. Раскроется список подключенных к компьютеру носителей информации. Сперва в нем будут идти логические диски C:, D: и т.д. За логическими дисками пойдут имена подключенных к компьютеру флешек. Перемещаясь стрелками горизонтального управления курсором можно выбрать желаемый носитель информации и нажать клавишу Enter. На соответствующей панели будет отображено его содержимое.

Скопировать или переместить объект можно действуя по следующему алгоритму:

1. Запустите программу Total Commander.
2. Откройте на любой панели тот носитель, на который надо скопировать или переместить объекты. Используйте команды Alt +F1 или Alt +F2.
3. Откройте папку, в которую следует скопировать или переместить объект. Для входа в папку используйте клавишу Enter, а для выхода BackSpace.
4. Откройте на противоположной панели тот носитель, на котором находится подлежащий копированию объект. Используйте команды Alt +F1 или Alt +F2. Обратите внимание, что при открытии какого-либо носителя на данной панели фокус переходит на эту панель автоматически, т.е. нажимать клавишу Tab нет необходимости.

5. Установите фокус на папку или файл, подлежащий копированию или перемещению. Используйте для этого клавиши вертикального управления курсором и клавишу Enter для входа в подкаталоги.

6. Нажмите клавишу F5 для копирования объекта или F6 для его перемещения.

7. Подтвердите свое желание выполнить операцию нажатием клавиши Enter.

Для переименования объектов удобно использовать клавиатурную команду Shift +F6 . После её ввода на подлежащем переименованию объекте в появившемся поле редактирования следует ввести новое имя объекта и нажать Enter. В этом случае объект (файл или папка) никуда не переместится, а будет переименован и останется в том же месте.

Для создания папки следует перейти в то место, где она будет создаваться и нажать клавишу F7. Затем в раскрывшемся диалоге в поле редактирования ввести имя создаваемой папки и нажать клавишу Enter для подтверждения. Обратите внимание, что здесь в поле редактирования тоже будет присутствовать предлагаемое имя. В отличие от Проводника это будет не «Новая папка», а имя той папки или того файла, на котором стоял фокус в момент нажатия клавиши F7.

С помощью клавиши F8 в Total Commander удаляются папки и файлы. Чтобы удалить какой-либо объект следует установить на него фокус, нажать клавишу F8 и Enter для подтверждения. Для окончательного удаления папок и файлов без помещения их в «Корзину» надо нажимать сочетание клавиш Shift +F8. Если вы случайно нажали F8, закрыть диалог удаления можно клавишей Escape и файлы удалены не будут.

Просмотреть текстовый файл можно с помощью клавиши F3. В этом случае файл можно будет только читать, возможности его редактирования не будет. Для редактирования файла следует пользоваться клавишей F4. При этом файл откроется для редактирования в стандартном редакторе Windows «Блокнот». Редактор по умолчанию можно заменить на любой другой с помощью настроек Total Commander.

Контрольные вопросы

1. Что такое файл?
2. Что такое папка?
3. Какие типы файлов вы знаете?
4. Могут ли существовать файлы с одинаковыми именами? В каком случае?
5. Что такое путь к файлу?
6. Какие операции можно выполнять над файлами и папками?
7. Какие программы называются файловыми менеджерами?
8. Какие способы отыскания объекта в Проводнике вы знаете?
9. Какие команды служат для перемещения по файловой структуре в Проводнике?
10. Расскажите, как можно создать папку в Проводнике.
11. Расскажите, как можно скопировать объект в Проводнике.
12. Для чего нужен буфер обмена?
13. Что такое «Корзина»?
14. Как можно переименовать объект в Проводнике?
15. Для чего служит программа Total Commander?
16. Опишите внешний вид интерфейса программы Total Commander.
17. Как выполняются файловые операции в Total Commander?

§6.2. Групповые операции

Встречаются задачи, при выполнении которых необходимо производить какие-либо операции над группой объектов (файлов и папок), находящихся в одной папке. В этом случае нужно выделить эту группу объектов и произвести точно такие же действия, как при работе с одним объектом.

Если требуется выделить все объекты в некоторой папке, то можно воспользоваться клавиатурной командой **Ctrl + A**. Эта команда сработает в любом файловом менеджере, как Проводник, так и Total Commander.

Чтобы отменить выделение группы объектов (в частности всех) достаточно переместить фокус в любом направлении с помощью клавиш вертикального перемещения курсора. После перемещения

будет выделен только тот объект, на который указывает фокус, а с остальных объектов выделение будет снято. Обратите внимание, что снять выделение подобным образом можно только в Проводнике, как снять выделение в Total Commander будет рассказано ниже.

Существует общий для Проводника и Total Commander алгоритм выделения идущих подряд объектов:

1. Запустите файловый менеджер (Проводник или Total Commander).
2. Установите фокус на объект, с которого начнётся выделение.
3. Удерживая клавишу Shift нажатой, переместите фокус до последнего объекта, подлежащего выделению.

Будут выделены идущие подряд объекты, с которыми можно проводить те же операции, что и с одним объектом, за исключением переименования. Помните, что любое перемещение фокуса в Проводнике снимет выделение, а в Total Commander выделение на объектах останется!

Если выделен лишний файл, то, не отпуская Shift, следует нажать Стрелку вверх. С нижнего объекта выделение снимается.

Выделение объектов снизу вверх выполняется аналогично. Только нужно поместить фокус на нижний объект и нажимать Стрелку Вверх в сочетании с клавишей Shift.

Чтобы озвучить имена выделенных объектов, используйте клавиатурную команду Ins +Shift +стрелка вниз. JAWS сообщит количества выделенных объектов, а затем, будут перечислены их имена. Обратите внимание, что данная команда программы не визуального доступа работает только в Проводнике. В Total Commander изучить группу выделенных объектов можно перемещаясь по ней курсорными стрелками. JAWS будет произносить слово «выделено» перед именем выделенного объекта.

При выделении подряд стоящих объектов можно использовать клавиши Home и End в сочетании с клавишей Shift, чтобы выделить от текущего объекта до начала или конца списка соответственно.

В Проводнике можно выделять объекты не стоящие рядом. Если удерживать клавишу Ctrl и при этом перемещать фокус по списку, то объект, на котором раньше находился фокус останется выделенным,

а объект, на который переместился фокус, будет не выделен. При этом JAWS сообщит название объекта, на который переместился фокус и что он не выделен. Чтобы добавить текущий объект к множеству выделенных, добавьте к уже нажатой клавише Ctrl клавишу Пробел. Если возникнет необходимость отменить выделение данного объекта, то повторное нажатие сочетания Ctrl +пробел исключает его из множества выделенных. После отпускания клавиши Ctrl процесс выделения группы объектов завершается и добавить еще что-то к группе выделенных объектов уже нельзя.

Например, если необходимо выделить в списке из десяти объектов второй, пятый и восьмой, то можно воспользоваться следующим алгоритмом:

1. Запустите Проводник.
2. Установите фокус на второй объект и нажмите клавишу CTRL (отпускать CTRL нельзя!).
3. Переместите фокус стрелкой вниз до пятого объекта (CTRL остаётся нажатым!).
4. На пятом объекте нажмите и отпустите клавишу Пробел, после чего пятый объект выделяется.
5. Продолжая удерживать CTRL, переместите курсорной стрелкой вниз фокус на восьмой объект.
6. На восьмом объекте нажмите клавишу Пробел и отпустите обе клавиши Ctrl и Пробел.

После выполнения алгоритма эти три выделенных объекта образуют единую группу, с которой можно выполнить любую операцию как с единым объектом (копировать, переместить или удалить).

Таким же способом можно снять выделение с объектов, идущих не подряд, если предварительно выделить все объекты. Разница только в том, что теперь нажатие клавиши Пробел будет приводить к снятию выделения. В результате также получится единая группа объектов, с которой можно выполнять файловые операции как с единым целым.

При ошибочном выполнении какой-либо операции ее можно отменить. Программа Проводник, как и большинство приложений Windows, дает возможность отменить последнее действие и вернуть

то состояние, которое было до выполнения ошибочной операции. Для этого следует ввести клавиатурную команду CTRL +Z. После этого перемещенные или удаленные объекты вернутся в ту папку, откуда были перемещены или удалены, скопированные файлы исчезнут из папки назначения и т.д.

Обратите внимание, что клавиатурная комбинация CTRL +Z в файловом менеджере Total Commander служит для выполнения другой команды (вспомните какой). Отменить файловую операцию подобным образом в Total Commander нельзя.

Для выделения группы объектов в Total Commander используется клавиша Пробел. Перемещаясь стрелками вертикального управления курсором по списку папок и файлов, нажимайте Пробел на тех объектах, которые нужно выделить. При перемещении фокуса выделение не снимается, т.е. таким образом можно выделять объекты как идущие подряд, так и в разброс. Для выделения всех объектов в папке можно также, как в Проводнике использовать комбинацию клавиш CTRL +A. После выделения группы объектов как и в Проводнике Windows, с ней можно проделать любую файловую операцию кроме переименования.

Контрольные вопросы

1. Приведите пример ситуации, в которой необходимо произвести операцию над группой файлов или папок.
2. Как можно выделить все объекты в папке?
3. Как снять выделение с объектов в Проводнике?
4. Как можно выделить группу идущих подряд объектов?
5. Какую файловую операцию нельзя производить над группой объектов? Как вы думаете почему?
6. Что делать, если был случайно выделен лишний объект?
7. Как в Проводнике озвучить имена выделенных объектов?
8. Как в Проводнике выделить группу объектов не стоящих в списке рядом?
9. Как в Total Commander выделить группу объектов не стоящих в списке рядом?

10. Как в Проводнике отменить последнее действие? Что эта клавиатурная команда выполняет в Total Commander?

11. Как в Total Commander выделить группу объектов, не стоящих в списке рядом?

12. Как в Total Commander снять выделение с объекта?

§6.3. Шаблоны имен

Часто для объединения файлов в группу бывает удобно (а иногда даже необходимо!) использовать только часть их имени или расширения. Например, может возникнуть необходимость выбрать из папки с очень большим количеством файлов те, в именах которых присутствует слово «программирование». Для решения подобных задач используются шаблоны имён файлов.

Шаблон представляет собой имя файла, в котором используются специальные символы «?» (вопросительный знак) и «*» (звёздочка). Каждый из этих специальных символов заменяет один или несколько символов в имени файла:

- «?» (вопросительный знак) — заменяет любой одиночный символ в имени файла (причём символ должен обязательно присутствовать);
- «*» (звёздочка) — заменяет любой (в том числе пустой) набор символов в имени файла.

Другими словами, вопросительный знак указывает на то, что на его месте в имени файла обязательно должен стоять какой-либо единственный допустимый символ (буква, цифра, знак подчёркивания и т.д.). Звёздочка указывает на то, что на её месте может стоять любое количество допустимых символов, а также вообще может не быть никаких символов.

Заметим, что описанные специальные символы могут применяться как в имени файла, так и в расширении.

Шаблоны имен часто используются в качестве параметра для задания группы файлов во многих командах операционной системы Windows. При использовании шаблона операционная система будет просматривать все файлы в данной папке (а иногда и всю файловую систему), после чего имена, соответствующие шаблону, включаются в группу, с которой будет производиться некоторая операция.

Поясним принцип работы шаблона на примерах:

«*.*» – этот шаблон соответствует всем (любым) именам файлов (например, «лето.txt», «winword.exe», «стихотворение.docx» и т.д.);

«*.txt» – этот шаблон соответствует файлам с любым именем и расширением txt (например, «лето.txt», «зима.txt», «Саратов.txt» и т.д.);

«*s*.docx» – этот шаблон соответствует всем файлам с расширением docx, в имени которых присутствует буква «s», причём имя может состоять только из одной буквы «s» (например, «s.docx», «dools.docx», «fresh.docx» и т.д.);

«*.*??.*» – такой шаблон соответствует файлам с произвольными именами, но имеющими три любых символа в расширении (например, этот шаблон соответствует файлам с расширением txt, но не соответствует файлам с расширением docx);

«ко?.txt» – такой шаблон объединит файлы с расширением txt и именами из трёх символов, последний из которых может быть произвольным, а первые два «ко» (например, «кот.txt», «ком.txt», «код.txt» и т.д.).

Файловый менеджер Total Commander имеет удобный механизм работы с шаблонами имён. Клавиша Плюс на дополнительной клавиатуре вызывает диалоговое окно «Добавить выделение», в котором можно ввести необходимый шаблон и в соответствии с ним выделить файлы в текущей папке. Обратите внимание, что клавиша Плюс на дополнительной клавиатуре включает режим РС-курсора программы JAWS. Поэтому при включённом JAWS перед вызовом диалога выделения файлов следует ввести команду пропуска клавиши для того, чтобы JAWS пропустил эту команду, а выполнил её Total Commander.

Приведём подробный алгоритм выделения группы файлов в соответствии с шаблоном с помощью Total Commander:

1. Запустите Total Commander.
2. Перейдите в папку, в которой находятся искомые файлы.
3. Введите команду Ins +3 для того, чтобы JAWS пропустил следующую команду, JAWS произнесет «введите клавишу для пропуска». Это необходимо, поскольку следующая команда совпадает с командой включения РС-курсора в JAWS.

4. Нажмите клавишу Плюс на дополнительной клавиатуре для вызова диалога «Добавить выделение». Курсор окажется в поле редактирования, где уже будет написан шаблон «*.*».

5. Отредактируйте имеющийся по умолчанию шаблон «*.*» в соответствии со своей задачей и нажмите Enter.

После выполнения этого алгоритма в открытой в данный момент папке будут выделены все соответствующие шаблону файлы. Выделенные таким образом файлы образуют группу, о работе с которой было рассказано в предыдущем параграфе.

В файловом менеджере Total Commander существует также возможность снятия выделения по шаблону. Диалог «Снять выделение» вызывается командой Минус на дополнительной клавиатуре. Поскольку клавиша Минус на дополнительной клавиатуре включает режим JAWS-курсора, то при работе с JAWS перед вызовом этого диалога также следует ввести команду Ins +3 пропуска следующей клавиши.

Заметим, что в некоторых случаях для выделения необходимой группы файлов бывает гораздо удобнее сперва выделить все файлы командой CTRL +A, а затем нажав клавишу Минус на дополнительной клавиатуре и введя соответствующий шаблон снять выделение с лишних. Например, если необходимо оставить в данной папке только текстовые файлы, то гораздо проще выделить сначала все, а затем используя шаблон «*.txt» снять выделение с текстовых файлов. При этом останутся выделенными все остальные файлы и их можно удалить нажав F8 или Delete.

Если в диалоге «Снять выделение» не менять шаблон по умолчанию «*.*», то можно будет снять выделение со всех файлов в открытой папке.

Шаблоны часто используются вместо полных имен файлов в параметрах служебных программ, а также при просмотре, поиске и сортировке файлов, в случае их очень большого количества, когда просмотреть весь список файлов вручную слишком долго. Также следует учесть, что при просмотре списка файлов вручную можно допустить ошибку, пропустить какой-либо файл, а использование

шаблонов обеспечит точность отбора необходимых файлов из сколь угодно большого списка.

Правильно задавать шаблон – это своего рода искусство, с помощью которого можно быстро и безошибочно выбрать файлы с нужными именами. Навык работы с шаблонами приходит с опытом их использования. Если возникает необходимость обработки большого количества файлов, старайтесь использовать шаблоны.

Контрольные вопросы

1. Что такое шаблон имени файла?
2. В каких случаях применяются шаблоны имён файлов? Приведите примеры.
3. Какие символы применяются в шаблонах имён? Расскажите, какую функцию выполняет каждый из них.
4. Какие символы являются допустимыми для имени файла?
5. Почему имя файла не может содержать звездочку и вопросительный знак?
6. Расскажите, как в Total Commander можно выделить группу файлов по шаблону.
7. Зачем в Total Commander нужен диалог «Снять выделение»?
8. Как снять выделение со всех файлов в Total Commander? А в Проводнике?

§6.4. Свойства файлов

Каждый объект файловой системы (папка или файл) имеет набор индивидуальных свойств (параметров). Файлы различных типов имеют различные наборы свойств. Так, например, набор свойств текстового документа отличается от набора свойств аудио-файла и т.д. Набор свойств папки также будет отличаться от набора свойств файла.

Свойства объекта просматривают с помощью файлового менеджера (Проводника или Total Commander). Для этого следует найти необходимый объект, установить на него фокус и либо вызвав контекстное меню выбрать команду «Свойства», либо ввести команду ALT +Enter. Раскрывшееся окно будет стандартным многостранич-

ным диалогом. На вкладках этого диалога содержится информация о дате создания объекта, о его типе, о возможностях доступа к объекту и др. Закрыть этот диалог можно клавишей Esc или нажав на кнопку «ОК» на любой вкладке.

Рассмотрим алгоритм работы с диалоговым окном свойств объекта:

1. Запустите файловый менеджер (Проводник или Total Commander).

2. Найдите объект (файл или папку), свойства которого необходимо изучить.

3. Введите команду ALT+Enter для открытия диалога со свойствами.

4. В открывшемся диалоговом окне будет четыре вкладки, переключаться между которыми следует командой CTRL+TAB. Для перемещения между полями внутри выбранной вкладки нужно клавишей TAB в прямом направлении и SHIFT+TAB в обратном направлении.

5. После ознакомления со свойствами объекта закрыть диалоговое окно можно клавишей Esc или нажав Enter на кнопке «Ок» в любой вкладке.

Применив этот алгоритм к любому файлу или папке легко видеть, что свойств у них достаточно много. Рассмотрим более подробно некоторые полезные на практике свойства.

Запустите файловый менеджер, выберите какой-либо файл и раскройте окно его свойств командой ALT+Enter. Фокус окажется на вкладке «Общие». Нажимая клавишу TAB изучите поля этой вкладки.

После имени вкладки «Общие» идёт поле редактирования с именем файла. В этом поле редактирования можно изменить имя файла. Обратите внимание, что для сохранения внесённых изменений необходимо нажать клавишу Enter (клавиша Esc отменит внесённые изменения и закроет диалог).

Следующим будет кнопка «Изменить...», раскрывающая диалог, в котором можно сопоставить тип данного файла и программу для его открытия. Например, файлы с расширением «txt» открывает редактор Блокнот, файлы с расширением «docx» открывает редактор

Word и т.д. В данном диалоге подобное соответствие можно изменить.

Далее идёт флажок «Только чтение». Если этот флажок установлен, то редактировать данный файл будет невозможно, он будет доступен только для чтения. Напомним, что устанавливать или снимать флажок следует клавишей Пробел. После установки или снятия флажка не забудьте нажать клавишу Enter, иначе внесённые изменения не сохранятся.

Следующим будет флажок «Скрытый». Его установка приведёт к тому, что в режиме по умолчанию файловые менеджеры не будут отображать этот файл. Для отображения скрытых файлов следует снять данный флажок или изменить настройки файлового менеджера. Заметим, что поскольку скрытые файлы не отображаются, с ними невозможно выполнить никакую файловую операцию.

Ещё одно нажатие клавиши TAB установит фокус на кнопку «Другие...», раскрывающую диалог с некоторыми редко используемыми дополнительными атрибутами файла. Здесь они рассматриваться не будут.

Далее фокус попадёт на кнопку «Ок». Именно её срабатывание происходит при нажатии клавиши Enter.

Часто бывает необходимо работать со свойствами файла, отображёнными на вкладке «Подробно». На этой вкладке можно найти информацию о том, кто является автором текстового документа, когда он выводился на печать, когда последний раз редактировался и т.д. Напомним, что различные типы файлов имеют отличающиеся друг от друга свойства, что особенно заметно в этой вкладке.

Наиболее интересны свойства файла, содержащего текстовый документ. Запустите файловый менеджер, установите фокус на файл с расширением «docx» и вызовите его свойства. Используя клавиатурную команду CTRL + TAB перейдите на вкладку «Подробно». Далее клавишей TAB перейдите на список свойств, по которому следует перемещаться стрелками вертикального управления курсором.

Внимательно изучив список свойств текстового документа легко видеть, что некоторые из них могут нести информацию о вас и

о вашем компьютере. В некоторых случаях эта информация должна быть изменена или полностью удалена. Например, фамилия автора документа вносится текстовым редактором Word автоматически. Поэтому, если вы создали файл не на своём компьютере, в поле имя автора будет стоять не ваше имя.

Приведём алгоритм изменения имени автора текстового документа Word через редактирование свойств файла:

1. Запустите файловый менеджер (Проводник или Total Commander).

2. Найдите файл с текстовым документом и установите на него фокус.

3. Откройте окно свойств файла введя команду ALT +Enter. JAWS произнесёт: «свойства» и добавит имя файла, свойства которого вы просматриваете.

4. Используя клавиатурную команду CTRL +TAB перейдите на вкладку «Подробно». После ввода этой команды JAWS будет называть активную вкладку (возможно, JAWS использует слово «страница» вместо слова «вкладка»).

5. Фокус будет стоять на поле «Просмотр списка». Если это не так, с помощью клавиши TAB перейдите на это поле.

6. С помощью стрелок вертикального управления курсором найдите свойство «Авторы».

7. Удалите введённое ранее имя автора и введите свое. Если сразу начать вводить своё имя, то оно будет добавлено к имеющемуся, количество авторов увеличится.

8. Введя имя нажмите клавишу Enter. Окно свойств не закроется, фокус останется на свойстве «Авторы». Для закрытия окна свойств с сохранением внесённых изменений нажмите Enter ещё раз или перейдя на кнопку «Ок» активируйте её Пробелом.

Заметим, что этот алгоритм актуален и для редактирования имени автора любого файла, созданного приложением Microsoft Office.

Есть способ удаления всех свойств, которые вообще можно удалить. Для этого следует воспользоваться ссылкой «Удаление свойств и личной информации», находящейся на той же вкладке «Подробно» после списка свойств. Для удаления свойств нажмите клавишу

Enter на этой ссылке для открытия диалога «Удаление свойств». После открытия диалога активной станет ссылка, перейдя по которой можно получить справку о том, какую личную информацию могут содержать свойства файла. Нажмите ещё раз TAB и фокус окажется на радиокнопках. Первая радиокнопка «Создать копию, удалив все возможные свойства» позволяет создать ещё одну копию файла, в которой все возможные свойства будут удалены. При создании копии к имени файла в конце будет добавлено слово «копия». Вторая радиокнопка «Удалить следующие свойства для этого файла:» Позволяет в списке выбрать те свойства, которые необходимо удалить. Если выбрать все свойства, то все они будут удалены непосредственно в данном файле, копия создана не будет.

При отправке файлов по электронной почте или при передаче их другим лицам, учитывайте наличие личной информации в свойствах! При необходимости её легко удалить.

Контрольные вопросы

1. С помощью какой программы просматривают свойства папки или файла?
2. Какая клавиатурная команда вызывает окно свойств объекта?
3. По какому алгоритму следует действовать, чтобы просмотреть свойства данного файла или папки?
4. Какие вкладки окна свойств содержат наиболее часто используемую информацию?
5. Какие способы переименования файла вы знаете?
6. Расскажите, как можно сопоставить данному типу файлов программу, которая с этими файлами будет работать.
7. Что произойдёт с файлом, если установить флажок «Только чтение»?
8. Что произойдёт с файлом, если установить флажок «Скрытый»?
9. Что означают три точки после имени какой-либо кнопки?
10. В какой вкладке окна свойств отображаются личные данные текстового документа?
11. Расскажите, как можно изменить имя автора текстового документа.

12. Расскажите, как можно удалить личную информацию из свойств текстового документа.

13. Какие ещё файлы могут содержать имя автора и другую личную информацию?

Глава 7

Текстовый процессор Word и программы невизуального доступа к информации

Материал этой главы предполагает наличие у обучающихся начальных сведений и практических навыков по работе в текстовом процессоре Word. Ниже приведены основные команды брайлевского дисплея для работы с текстовым документом, которые нужно повторить, а также ответить на контрольные вопросы. При необходимости следует вернуться к материалу по работе с редактором Word за предыдущие классы.

Команды брайлевского дисплея для работы с текстовым документом:

- Озвучить текущий символ синтезатором речи – пробел +точки 36;
- Перейти на предыдущий символ и озвучить его – пробел +точка 3;
- Перейти на следующий символ и озвучить его – пробел +точка 6;
- Озвучить текущее слово синтезатором речи – пробел +точки 25;
- Перейти на предыдущее слово и озвучить его – пробел +точка 2;
- Перейти на следующее слово и озвучить его – пробел +точка 5;
- Озвучить текущую строку синтезатором речи – пробел +точки 14;
- Перейти на предыдущую строку и озвучить ее– пробел +Точка 1;
- Перейти на следующую строку и озвучить ее– пробел +точка 4;
- Озвучить текущий абзац - левый Shift +правый Shift +точки 235678;
- Озвучить фрагмент текста от начала строки до курсора – Правый Shift +точки 37;
- Озвучить фрагмент текста от курсора до конца строки – Правый Shift +точки 68;
- Озвучить весь текст – пробел +точки 12456;
- Перейти в начало строки – пробел +точки 13;
- Перейти в конец строки – пробел +точки 46;

- Остановить речь – левый или правый Shift;
- Выделить предыдущий символ – пробел +точки 37;
- Выделить следующий символ – пробел +точки 67;
- Выделить предыдущее слово – пробел +точки 27;
- Выделить текущее слово – пробел +точки 57;
- Выделить предыдущую строку – пробел +точки 17;
- Выделить следующую строку – пробел +точки 47;
- Выделить от начала строки до курсора – пробел +точки 137;
- Выделить от курсора до конца строки – пробел +точки 467;
- Выделить от начала текста до курсора – пробел +точки 1237;
- Выделить от курсора до конца текста – пробел +точки 4567;
- Скопировать в буфер обмена – левый Shift +точки 14;
- Вырезать в буфер обмена – левый Shift +точки 1346;
- Вставить из буфера обмена – левый Shift +точки 1236;
- Отмена последнего действия – левый Shift +точки 1356;
- Удалить – левый Shift +точки 145;
- Win (вызов главного меню) – правый Shift +точка 4;
- Escape – пробел +точки 1356;
- Alt (для входа в меню приложения) – правый Shift +точка 2;
- Вызов контекстного меню (Application) – пробел +правый Shift +точка 2;
- Tab – пробел +точки 45;
- Shift +Tab – пробел +точки 12.

Двухтактные команды брайлевского дисплея для выравнивания абзацев:

- Переключение между выравниванием абзаца по левому краю и по ширине – пробел +точки 38, затем точки 1238;
- Переключение между выравниванием абзаца по центру и по левому краю – пробел +точки 38, затем точки 158;
- Переключение между выравниванием абзаца по правому краю и по левому краю – пробел + точки 38, затем точки 12358;
- Переключение между выравниванием абзаца по ширине и по левому краю – пробел +точки 38, затем точки 2458.

Двухтактные команды брайлевского дисплея для изменения междустрочного интервала:

- Одинарный междустрочный интервал – пробел +точки 38, затем точка 2;
- Полutorный междустрочный интервал – пробел +точки 38, затем точки 26;
- Двойной междустрочный интервал – пробел +точки 38, затем точки 23.

Двухтактные команды брайлевского дисплея для изменения начертания символов:

- Полужирный – пробел +точки 38, затем точки 128;
- Курсивный – пробел +точки 38, затем точки 248;
- Подчеркнутый – пробел +точки 38, затем точки 1368.

Двухтактные команды брайлевского дисплея для увеличения и уменьшения размера шрифта:

- Уменьшение размера шрифта на один пункт – пробел +точки 38, затем точки 1235678;
- Увеличение размера шрифта на один пункт – пробел +точки 38, затем точки 2345678.

Информация об имеющемся форматировании абзаца: Ins +F или пробел +точка 5, затем точки 1248.

Контрольные вопросы

1. Какие способы запуска текстового процессора Word вы знаете?
2. Какие способы работы с ленточным меню вы знаете?
3. Какие режимы озвучивания вводимого текста вы знаете?
4. В чем разница между восьмиточечным и шеститочечным брайлем?
5. Как вводить текст с помощью брайлевского дисплея?
6. Как вводить цифры с помощью брайлевского дисплея?
7. Как вводить знаки препинания с помощью брайлевского дисплея?
8. Перечислите основные структурные элементы текста.
9. Как можно выделить:
 - А) слово;
 - Б) строку;

В) предложение;

Г) абзац;

Д) весь текст?

10. Расскажите как работать с фрагментами текста с помощью брайлевского дисплея.

11. Зачем нужно выделять фрагменты текста?

12. Как можно изменить параметры форматирования символов?

13. Как можно вызвать диалог «Шрифт»?

14. Какие клавиатурные команды для изменения начертания символов вы знаете?

15. Какие параметры форматирования абзаца вы знаете?

16. Как можно вызвать диалог «Абзац»?

17. Как озвучить параметры форматирования текущего фрагмента текста?

18. Какие клавиатурные команды для выравнивания текста вы знаете? Как ввести эти команды с помощью брайлевского дисплея?

§7.1. Стили

Для создания заголовков различных уровней, нумерованных и маркированных списков, а также установления других элементов форматирования в текстовом процессоре Word удобно пользоваться стилями.

Стиль – это поименованный и сохраненный набор значений параметров форматирования элементов текста (абзацев и символов), который можно использовать многократно.

Стили бывают стандартные и пользовательские. Стандартные стили имеются в коллекции текстового процессора Word. Пользовательские стили создаются пользователем путем модификации стандартных стилей или создания новых.

Стили можно применять (присваивать) к отдельному элементу документа, например абзацу, определяя этот абзац, как обычный текст или как список, или как заголовок и т.п.

Каждый стиль имеет имя. Имена некоторых стилей определяют их назначение. Например, абзацы, отформатированные стилями «За-

головок 1», «Заголовок 2» и т.д., автоматически включаются в оглавление, стиль «Обычный» определяет формат обычного текста.

Для вызова диалога в текстовом процессоре Word, содержащего список стилей, можно использовать команду стандартной клавиатуры Ctrl +Shift +S. После открытия диалога фокус окажется на списке стилей, перемещение по которому осуществляется стрелками вертикального управления курсором, а выбор желаемого стиля клавишей Enter.

Опишем алгоритм присваивания некоторому абзацу статуса заголовка первого уровня:

1. Запустите текстовый процессор Word.
2. Введите или найдите в тексте документа абзац, который будет заголовком. Это может быть одно или несколько слов. Знак препинания в конце необязателен.
3. Установите курсор в пределах данного абзаца (на любой его символ включая пробел).
4. Вызовите диалог выбора стиля клавиатурной командой Ctrl +Shift +S.
5. Двигаясь стрелкой вниз, найдите в коллекции стилей стиль «Заголовок 1» и нажмите клавишу Enter.

Теперь введенный вами текст приобрел статус заголовка первого уровня. При создании оглавлений он будет автоматически учитываться процессором Word.

Изменить стандартный стиль можно двумя способами: обновить стиль в соответствии с параметрами отформатированного фрагмента текста или изменить стиль вручную в диалоговом окне «Изменение стиля».

Для изменения стиля первым способом необходимо проделать следующее:

1. Запустите текстовый редактор Word и загрузите в него какой-либо документ.
2. Найдите в тексте фрагмент, к которому применен подлежащий изменению стиль (например, «Заголовок 1»).

3. Отформатируйте данный фрагмент с использованием желаемых параметров (например, можно изменить размер шрифта для стиля «Заголовок 1» с 16 до 14 пунктов).

4. Выделите отформатированный фрагмент текста.

5. Введя команду Alt + Я активируйте вкладку «Главная» (сделать это можно нажав Alt и стрелками горизонтального управления курсором найдя нужную вкладку).

6. На вкладке «Главное» найдите группу «Стили» и в ней найдите кнопку соответствующего стиля (сделать это можно используя клавишу Tab или введя русскую букву «К»). Обратите внимание, что имя стиля, которым отформатирован отмеченный фрагмент текста, будет выделено.

7. Вызовите на выделенном имени стиля контекстное меню.

8. В контекстном меню выберите команду «Обновить (название стиля) в соответствии с выделенным фрагментом».

После этого все параметры, в соответствии с которыми был отформатирован выделенный фрагмент, будут внесены в данный стиль. Все фрагменты текста, к которым применен этот стиль, автоматически переформатируются.

Можно изменить стиль в коллекции, не форматировая текст в документе. Для этого следует воспользоваться следующим алгоритмом:

1. Запустите текстовый процессор Word.

2. Вызовите клавиатурной командой Ctrl + Shift + S диалог со списком стилей.

3. Выберите в списке стиль, подлежащий изменению.

4. Клавишей Tab перейдите на кнопку «Изменить...» и нажмите ее.

5. В открывшемся диалоге изменения стиля установите желаемые параметры форматирования.

6. Выберите радиокнопку «Только в этом документе» или «Во всех новых документах» (смысл этих радиокнопок очевиден) и нажмите кнопку «ОК».

После этого стиль будет изменен и, как и в предыдущем случае, все фрагменты текста, отформатированные данным стилем, автоматически переформатируются.

Можно добавить в коллекцию новый стиль, созданный на основе отформатированного заранее фрагмента текста. Для этого следует поступать по алгоритму:

1. Запустите текстовый процессор Word и загрузите в него какой-либо документ.

2. Выберите в документе фрагмент текста и отформатируйте его желаемым образом, используя диалоговые окна «Абзац» и «Шрифт» или соответствующие клавиатурные команды.

3. Выделите подготовленный таким образом фрагмент текста.

4. Активируйте вкладку «Главное» введя команду Alt + Я.

5. Перейдите в группу «Стили» введя русскую букву «К» или используя клавиатурную команду Ctrl + стрелка вправо.

6. Раскройте диалог «Создание стиля» введя русскую букву «З» или выбрав в списке соответствующую команду.

7. В диалоговом окне «Создание стиля» в соответствующее поле редактирования введите имя создаваемого стиля и нажмите кнопку «ОК».

После выполнения указанных операций новый стиль появится в коллекции «Стили» и его можно будет использовать как и встроенные.

При копировании или перемещении фрагмента текста, содержащего определенное форматирование, отличное от форматирования целевого документа (т.е. документа, в который будет вставлен данный фрагмент) могут возникнуть различные ситуации.

Для вставляемого фрагмента текста может быть сохранено его исходное форматирование, отличающееся от форматирования текста в целевом документе. Например, вставляемый текст был набран шрифтом Times New Roman размера 10 пунктов и имел полужирное начертание, а целевой текст Набран шрифтом Calibri размера 11 пунктов обычного начертания. При этом вставленный фрагмент должен быть набран шрифтом Times New Roman, а не Calibri.

В этом случае поступать можно по следующему алгоритму:

1. Выделите текст, который необходимо переместить или скопировать.

2. Поместите его в буфер обмена введя команду вырезания или копирования.

3. Переместите курсор в желаемое место вставки фрагмента.

4. Вызовите контекстное меню.

5. В контекстном меню в группе «Параметры вставки» выберите кнопку «Сохранить исходное форматирование».

Обратите внимание, что между пунктами контекстного меню перемещаться следует стрелками вертикального управления курсором, а в группе команд «Параметры вставки» между кнопками удобнее перемещаться клавишей Tab.

Если необходимо удалить все исходное форматирование вставляемого текста, то в пятом пункте алгоритма в контекстном меню выберите кнопку «Сохранить только текст». Обратите внимание, что если выделенный фрагмент включал содержимое, не являющееся текстом, команда «Сохранить только текст» удалит его или преобразует в текст. Например, если команда «Сохранить только текст» используется при вставке фрагмента с рисунками и таблицами, то во вставленном тексте рисунков не будет, а таблицы преобразуются в ряд абзацев.

Если при работе в текстовом редакторе Word чаще других используется один из параметров вставки, можно настроить Word для его автоматического применения. Тогда станет ненужным выбирать параметр при каждой вставке текста и можно будет пользоваться стандартной клавиатурной командой Ctrl + V.

Для настройки параметров вставки по умолчанию можно воспользоваться следующим алгоритмом:

1. Запустите текстовый процессор Word.

2. Откройте вкладку «Файл» введя команду Alt + Ф (буква «Ф» русская) или выбрав эту вкладку на ленте.

3. Выберите команду «Параметры».

4. В раскрывшемся списке найдите пункт «Дополнительно». Обратите внимание, что Enter на этом пункте нажимать не следует.

5. Табулируя по полям диалога «Дополнительно» найдите область с несколькими комбинированными списками «Вырезание, копиро-

вание и вставка». Первым комбинированным списком в этой области будет «Вставка в пределах одного документа».

6. Выберите в списке желаемый режим вставки и нажмите Enter.

Обратите внимание, что параметры вставки можно настроить отдельно для случаев копирования в пределах одного документа, разных документов и для копирования из других программ.

Контрольные вопросы

1. Что такое стиль в текстовом процессоре Word?
2. Зачем нужно так много стилей?
3. Как применить стиль к фрагменту текста?
4. Как внести изменения в стиль?
5. Как можно создать новый стиль?
6. Что произойдет с параметрами форматирования фрагмента текста при его вставке в документ с другими параметрами форматирования?
7. Как изменить параметры вставки по умолчанию?

§7.2. Оглавление и сноски

В документе Word для именования глав, параграфов, частей и т.д. существует понятие «заголовок». Текстовый процессор Word поддерживает работу с заголовками девяти уровней – от первого (наивысшего) до девятого. Каждый заголовок – это абзац. Любому абзацу можно присвоить статус заголовка того или иного уровня. Только те абзацы, которые имеют уровень отличный от уровня основного текста, используются при автоматическом создании оглавления и при навигации по тексту с помощью программы невизуального доступа в режиме «Клавиш быстрой навигации». Устанавливается статус заголовка необходимого уровня с помощью стилей как это было показано в предыдущем параграфе.

Текстовый процессор Word может автоматически создать оглавление по имеющимся в тексте документа форматированным соответствующим стилем заголовкам. Чтобы автоматически создать оглавление в Word, необходимо, чтобы в тексте были заголовки, от-

форматированные при помощи соответствующего стиля «Заголовок 1», «Заголовок 2» и т.д., в зависимости от уровня заголовка.

Оглавление является неотъемлемой частью достаточно длинного текстового документа, например, курсовой работы, описания проекта и т.д. Оглавление существенно упрощает работу с электронной книгой, позволяя обратиться сразу к необходимому разделу текста.

Приведём алгоритм автоматического создания оглавления в текстовом редакторе Word:

1. Опираясь на логическую структуру документа создайте в тексте с помощью применения стилей «Заголовок 1», «Заголовок 2» и т.д. заголовки соответствующих уровней. Обратите внимание, что в списке стилей заголовки более низких уровней появляются не сразу. Чтобы появился стиль «Заголовок 3» предварительно необходимо создать в тексте заголовок уровня 1 или 2. Аналогично в списке будут появляться стили «Заголовок 4», «Заголовок 5» и т.д. Таким образом, создания заголовков следует начинать с заголовков более высокого уровня.

2. Установите курсор в то место документа, в котором предполагается создать оглавление. Если необходимо, чтобы под оглавление была выделена отдельная страница, то следует вставить разрыв страницы с помощью клавиатурной команды Ctrl +Enter перед и после будущего оглавления.

3. Раскройте на ленте вкладку «Ссылки» и нажмите Enter на команде «Оглавление».

4. В раскрывшемся диалоге выберите кнопку «Автособираемое оглавление 1». Фокус ввода сразу должен оказаться на этой кнопке. Также можно выбрать другую кнопку «Автособираемое оглавление 2». Автособираемое оглавление 1 и 2 в русской версии Word аналогичны.

5. После нажатия на выбранную кнопку оглавление будет автоматически создано.

В созданном таким образом оглавлении каждый заголовок будет снабжен отточием (так называемым заполнителем) и номером страницы, на которой он находится в тексте.

Если на каком-либо пункте оглавления ввести команду Ctrl +Enter, то курсор перейдёт на соответствующий заголовок в тексте документа.

По умолчанию, автособираемое оглавление в Word включает только заголовки первых трех уровней. Чтобы включить в оглавление заголовки, например, четвертого уровня, нужно выбрать кнопку «Настраиваемое оглавление». В раскрывшемся диалоге на вкладке Оглавление установите в комбинированном списке «Уровни» значение 4 и нажмите кнопку «ОК».

Оглавление в Word создается на основе стилей, поэтому форматировать его нужно тоже с помощью стилей. Если необходимо изменить шрифт, размер, начертание или другие атрибуты форматирования оглавления, выберите в списке нужный стиль и задайте форматирование для заголовков этого уровня. Сделайте так для всех стилей, которые используются для заголовков в тексте. Не рекомендуется выделять оглавление целиком и применять к нему единое форматирование, поскольку при обновлении оглавления это форматирование будет потеряно.

Если редактировать документ с уже созданным оглавлением, то в него могут добавиться новые заголовки, а номера страниц со старыми заголовками изменяться. В результате оглавление станет неактуальным. В этом случае его следует обновить. Для обновления оглавления можно воспользоваться следующим алгоритмом:

1. Установите курсор в любой точке оглавления.
2. Вызовите контекстное меню и в нем выберите команду «Обновить поле».
3. В раскрывшемся диалоге стрелками управления курсором установите радиокнопку «Обновить целиком» и нажмите Enter. Если выбрать радиокнопку «Обновить только номера страниц», то новые заголовки к оглавлению добавлены не будут, а номера страниц со старыми станут актуальными.

В тексте документов часто используются сноски. Сноски – это заметки, которые используются для объяснения или дополнения текста на странице. В документе Word можно использовать сноски

внизу текущей страницы или сноски в конце документа (концевые сноски).

Вставить сноску в документе Word можно с помощью следующего алгоритма:

1. Поставьте курсор после слова, к которому необходимо создать сноску.

2. Раскройте вкладку «Ссылки» и в группе «Сноски» выберите команду «Вставить сноску».

3. В открывшемся поле редактирования введите текст сноски и нажмите Enter.

Обратите внимание, что после выполнения данного алгоритма сноска добавится в нижней части страницы. Для добавления концевой сноски на втором шаге алгоритма нужно в группе «Сноски» выбрать кнопку «Вставить концевую сноску».

Для того, чтобы при чтении документа вывести в отдельное окно список всех сносок следует ввести команду **Ins + Shift + f**. А для озвучивания текста сноски следует установить курсор на её значок (номер сноски) и ввести команду **Alt + Shift + e**.

Для изменения параметров форматирования сносок, например, изменения формата номера или положения сносок в документе, можно воспользоваться следующим алгоритмом:

1. На вкладке «Ссылки» в группе «Сноски» нажмите кнопку «Диалоговое окно сносок».

2. В раскрывшемся диалоге найти комбинированный список «Формат номера» и выбрать в нём желаемый формат номера сноски, например, «1,2,3», «a, b, c» или «i, ii, iii». В этом диалоге можно настроить и другие параметры форматирования сносок.

3. Подтвердить внесённые изменения нажатием кнопки «Применить».

Для удаления сноски следует выделить в тексте документа её номер или маркер и нажать клавишу **Delete** или **BackSpace**. Если была удалена нумерованная сноска, то Word автоматически обновит нумерацию оставшихся сносок.

Контрольные вопросы

1. Что такое заголовок в Word?
2. Зачем нужны заголовки различных уровней?
3. Зачем нужно оглавление в документе Word?
4. Как создать оглавление в документе Word?
5. Чем отличается оглавление в обычной книге от оглавления в документе Word?
6. Как по оглавлению перейти сразу на искомый заголовок в тексте документа Word?
7. Заголовки каких уровней включает Word в оглавление по умолчанию?
8. Как расширить список уровней заголовков, включаемых в оглавление?
9. Зачем нужно обновлять оглавление?
10. Как можно автоматически обновить оглавление?
11. Что такое сноска?
12. Где в документе Word может располагаться текст сноски?
13. Как можно вставить сноску в документе Word?
14. Как удалить сноску?
15. Как изменить формат номера сноски?

§7.3. Поиск и замена

В текстовом процессоре Word есть удобные инструменты поиска и замены какого-либо контекста. Для поиска в тексте определённого набора символов (например, слова или его части) используется команда Ctrl + F, а для поиска с последующей заменой искомого набора символов на другой служит команда Ctrl + H.

Заметим, что Word позволяет искать в тексте не только определённые наборы букв, но и любые допустимые символы, например, @ (собака), \$ (доллар) или [] (квадратные скобки).

Здесь будет рассмотрен только простейший функционал поиска и замены. Алгоритмы выполнения более сложных операций вы сможете изучить самостоятельно.

Рассмотрим применение функции «Найти и заменить» на примере решения конкретной задачи.

Пример 1. Наберите в текстовом редакторе Word названия десяти городов, разделяя их запятыми. Затем каждую букву «а» заключите в квадратные скобки.

Решение. После запуска текстового редактора Word, введите названия городов, например, Москва, Санкт-Петербург, Новосибирск, Екатеринбург, Нижний Новгород, Казань, Челябинск, Омск, Самара, Ростов-на-Дону.

Для выполнения второй части задания используем команду «найти и заменить», для вызова которой введите клавиатурную команду Ctrl +H. В поле редактирования «Найти» введите букву «А» (буква вводится без кавычек). клавишей Tab перейдите в следующее поле редактирования «заменить на:», в которое введем букву, заключенную в квадратные скобки. После этого перейдем клавишей Tab к кнопке «заменить все» и активируем ее клавишей пробел. программа сообщит число выполненных замен. Нажимаем пробел для закрытия диалогового окна с этой информацией. После этого, закрываем диалоговое окно «найти и заменить», нажав клавишу Escape.

Задание выполнено.

Аналогичным образом осуществляется поиск в текстовом документе. После ввода команды Ctrl +F в текстовое поле открывшегося диалога «Поиск в документе» следует ввести искомый набор символов (слово или его часть) и нажать Enter. Word найдёт первое вхождение данного набора символов в текст и предложит продолжить поиск или перейти в текст к найденному фрагменту. Фокус будет стоять на кнопке «Продолжить поиск», поэтому для продолжения поиска достаточно нажать клавишу Пробел. Для прекращения поиска и перехода в текст документа к найденному фрагменту можно нажать клавишу Escape.

Контрольные вопросы

1. Зачем нужны функции поиска и замены?
2. Чем отличаются результаты выполнения команд Ctrl +F и Ctrl +H?
3. Опишите алгоритм замены определённого набора символов во всём тексте на другой.

4. Как можно узнать, сколько раз встречается в тексте конкретное слово?

§7.4. Таблицы в текстовом документе

Текстовый процессор Word позволяет оформлять данные в документах в виде таблиц. Таблица – это форма организации данных по столбцам и строкам, на пересечении которых находятся ячейки. В ячейках таблицы могут располагаться данные различного типа: текст, числа, рисунки, формулы и др.

В текстовом документе Word таблицу можно создать несколькими способами, например, воспользовавшись следующим алгоритмом:

1. Поместите курсор в то место документа, где будет создана таблица.

2. На вкладке «Вставка» нажмите Enter на подменю «Таблица». Здесь «Таблица» – это название группы, в которой находятся элементы управления различными параметрами таблицы.

3. Фокус окажется в поле «Вставка таблицы» на кнопке «1X1». При нажатии этой кнопки в текст будет вставлена таблица из одного столбца и одной строки. Передвигая фокус стрелкой вправо можно увеличивать количество столбцов во вставляемой таблице, а передвигая фокус стрелкой вниз можно увеличивать количество строк.

4. Перемещая фокус по этим кнопкам стрелками управления курсором выберите желаемый размер вставляемой таблицы, например, 2X3, т.е. 2 столбца и 3 строки.

После выполнения этих действий в документе появится таблица выбранного размера. Перемещаться по ячейкам созданной таблицы можно клавишей Tab или Shift + Tab в обратном направлении. Также можно пользоваться командой Ctrl + Стрелка вправо или Ctrl + Стрелка влево. Находясь в ячейке в нее можно вводить текст с клавиатуры или осуществлять вставку из буфера обмена.

Напомним, что в различных версиях текстового процессора Word и программы невизуального доступа к информации названия полей и сообщаемая пользователю информация могут быть различными. Однако, общий принцип вставки таблицы остаётся тем же.

Также можно преобразовать в таблицу ранее набранный текст. Для этого его необходимо разделить на столбцы и строки специальными символами. Поступать можно следующим образом:

1. Введите текст, разделяя БУДУЩИЕ ячейки символом табуляции (клавиша Tab стандартной клавиатуры или кнопки 45 +пробел на брайлевском дисплее). В конце строки нажмите Enter. Таким образом будет подготовлена первая строка будущей таблицы. При преобразовании в таблицу разделение на ячейки произойдет по символам табуляции, а на строки по символам абзаца (символ абзаца вводится при нажатии клавиши Enter). Таким же образом введите текст последующих строк таблицы.

2. Выделите подготовленный фрагмент текста.

3. На вкладке «Вставка» в группе «Таблица» нажмите кнопку «Преобразовать в таблицу...».

4. В открывшемся диалоговом окне «Преобразовать в таблицу» в первом поле можно изменить количество столбцов, которое Word определил автоматически. В следующем поле «Автоподбор ширины столбцов» доступны три радиокнопки, от которых зависит ширина столбцов. Если установить радиокнопку «Постоянная», то в следующем поле будет предоставлена возможность выбрать значение «Авто» или задать фиксированную ширину столбцов. Если установить значение «По содержимому», будут созданы узкие столбцы, расширяющиеся при добавлении содержимого в ячейки. Если же выбрать «По ширине окна», то ширина всей таблицы будет изменена в соответствии с размерами документа.

5. Выполнив все необходимые настройки, нажмите кнопку «ОК».

Теперь подготовленный заранее текст размещен в ячейках таблицы.

Обратите внимание, что свойства и возможности таблицы не зависят от способа ее создания.

Указанное при создании таблицы число столбцов и строк можно изменять, добавляя новые или удаляя существующие строки и столбцы.

Для добавления новой строки в конце таблицы нужно установить курсор в последней ячейке (JAWS сообщит, что курсор находится в последней ячейке таблицы) и нажать клавишу Tab. При этом кур-

сор окажется в первой ячейке добавленной строки. Добавить подобным образом столбец нельзя.

Более универсальный способ добавления столбцов и строк описывается следующим алгоритмом:

1. Поместите курсор в ячейку, рядом с которой необходимо добавить столбец или строку.

2. Раскройте контекстное меню.

3. В контекстном меню раскройте подменю «Вставить».

4. В открывшемся меню есть команды вставки столбца слева или справа и строки сверху или снизу по отношению к текущей ячейке. Т.Е. если выбрать команду «Вставить столбцы слева», то будет вставлен столбец слева от того, которому принадлежит активная ячейка.

Для удаления столбцов или строк следует поступать аналогичным образом:

1. Поместите курсор в ячейку, принадлежащую тому столбцу или той строке, которую необходимо удалить.

2. Раскройте контекстное меню.

3. В открывшемся меню выберите команду «Удалить ячейки...».

4. В открывшемся диалоговом окне курсорными стрелками выберите необходимую радиокнопку. Например, для удаления строки выберите радиокнопку «Удалить всю строку» и нажмите Enter.

Если выделить некоторую область в таблице (возможно всю таблицу) удерживая нажатой клавишу Shift и перемещая курсор, а затем нажать клавишу Delete, то будет удалено только содержимое выделенных ячеек, сама структура таблицы останется неизменной. Для удаления всей таблицы, ее необходимо выделить вместе с маркером абзаца, следующего за ней, и нажать клавишу Delete.

Обратите внимание, что по умолчанию маркеры абзацев не отображаются. В этом случае достаточно выделить пустую строку за таблицей. Если же необходимо отобразить маркеры абзацев, то следует воспользоваться командой Ctrl + Shift + 8.

Ячейка таблицы может содержать другую вложенную таблицу. Для создания вложенных таблиц надо поместить курсор в ячейку, в которую необходимо вложить таблицу, и выполнить те же действия, как при создании таблицы.

Когда при попадании курсора в некоторую таблицу JAWS сообщает «Однородная таблица» – это означает, что таблица имеет самую простую конструкцию. В ней нет вложенных таблиц, объединений ячеек и других особенностей структуры.

Текстовый процессор Word обеспечивает широкие возможности для редактирования и форматирования таблиц. Создав в документе таблицу, с ней можно совершать следующие операции:

- Вводить, копировать или вставлять текст и данные в ячейки;
- Перемещать текст и данные между ячейками в таблице и между различными таблицами;
- Вставлять или удалять ячейки, строки и столбцы;
- Объединять ячейки как по горизонтали, так и по вертикали;
- Окаймлять таблицу и отдельные ячейки границами;
- Изменять интервалы между ячейками;
- Помещать в ячейки рисунки;
- Выполнять в ячейках вычисления с помощью формул и функций (как в Excel);
- Сортировать данные в ячейках и т.п.

На большинстве этих возможностей мы останавливаться не будем. Приобретя определенный опыт в работе с программой Word любые его возможности можно освоить самостоятельно.

В таблицах Word реализованы некоторые функции электронных таблиц. Подобно Excel, Word позволяет выполнять вычисления с помощью формул и функций. Для проведения вычислений в таблице следует установить курсор в ячейке, в которой необходимо получить результат. Как и в табличном процессоре Excel, формула всегда вводится после знака «=» (равно).

Word представляет результаты вычисления в виде полей (подробнее о полях будет рассказано в следующем параграфе). При изменении исходных данных в ячейках таблиц Word в отличие от Excel не происходит автоматического пересчета результатов вычислений. Поэтому при изменении исходных данных или ссылок на ячейки таблицы результаты вычислений необходимо обновлять с помощью клавиши F9 или команды контекстного меню «Обновить поле».

При копировании формул ссылки на адреса ячеек в таблицах Word автоматически не изменяются как в Excel, т.е. ссылки на ячейки в таблицах Word всегда являются абсолютными, хотя и отображаются без знака доллара. Поэтому при копировании формул ссылки на адреса ячеек нужно редактировать вручную.

При работе с таблицами в текстовом процессоре Word могут быть полезны следующие клавиатурные команды программы невидимого доступа к информации JAWS for Windows:

- Читать первую ячейку в текущем столбце – Insert +Пробел, T, Alt +1;
- Читать первую ячейку в текущей строке – Insert +Пробел, T, Alt +5;
- Вывести список таблиц в документе – Ctrl +Insert +t;
- Озвучить заголовок столбца для текущей ячейки – Insert +Alt +Shift+ C;
- Озвучить заголовок строки для текущей ячейки – Insert +Alt +Shift +R;
- Читать текущий столбец таблицы – Alt +Win +. (точка);
- Читать текущую строку таблицы – Alt +Win +, (запятая);
- Перейти и прочитать предыдущий столбец таблицы Ю Win +Alt +Стрелка влево;
- Перейти и прочитать следующий столбец таблицы – Win +Alt +Стрелка вправо;
- Перейти и прочитать предыдущую строку таблицы – Win +Alt +Стрелка вверх;
- Перейти и прочитать следующую строку таблицы – Win +Alt +Стрелка вниз;
- Пометить место в текущем документе – Ctrl +Win +k;
- Вернуться к отмеченному месту в текущем документе – Alt +Win +k;
- Выделить текст от отмеченного места до текущей позиции курсора в текущем документе – Insert +Пробел, M.

Пользоваться такими командами, как первые две приведенного списка следует последовательно. Так, например, чтобы выполнить первую команду, сперва необходимо ввести Insert +Пробел (при этом

будет воспроизведен характерный звук), затем ввести латинскую букву «Т», а после этого команду Alt +1.

Контрольные вопросы

1. Что такое таблица?
2. Как можно создать таблицу в текстовом процессоре Word?
3. Как преобразовать текст в таблицу в текстовом процессоре Word?
4. Как добавить строку или столбец к существующей таблице?
5. Как удалить строку или столбец из таблицы?
6. Что такое вложенная таблица?
7. Что такое однородная таблица?
8. В чем разница между таблицами Word и Excel?
9. Как преобразовать таблицу в текст? (для ответа на этот вопрос вам может понадобиться ранее изученный материал этой главы).

§7.5. Поля и макросы

В этом параграфе будут коротко описаны такие важные элементы текстового документа Word, как поля и макросы. Здесь термин «поле» означает не пространство по краям листа при печати, а особый функционал текстового процессора Word. Многие поля помещаются в текстовый документ с помощью встроенных команд и функций Word. Например, поля используются при вставке номеров страниц и создании оглавления.

С помощью полей можно осуществлять математические вычисления, ссылаться на другие документы, помещать в документ текущее время и дату и т.д.

Рассмотрим в качестве примера, алгоритм вставки в текстовый документ Word поля, отображающего текущую дату:

1. Поместите курсор в то место текстового документа Word, куда необходимо вставить дату.
2. На вкладке «Вставка» нажмите кнопку «Дата и время...».
3. В раскрывшемся диалоге установите желаемые параметры отображения даты или времени. Если необходимо, чтобы дата и время обновлялись автоматически, в этом диалоге следует установить флажок «Обновлять автоматически».

4. Завершите настройку параметров отображения даты и времени нажав кнопку «ОК».

После выполнения этих действий в выбранном месте документа появится сегодняшняя дата и текущее время, которые будут автоматически обновляться.

Каждое поле имеет свой код, т.е. некую мини программу, написанную на специальном языке. Коды полей отображаются внутри фигурных скобок. Фигурные скобки кодов полей невозможно ввести с клавиатуры.

Для ввода кода поля вручную следует использовать клавиатурную команду Ctrl +F9. Появятся фигурные скобки, между которыми можно ввести необходимый код поля. Обратите внимание, что JAWS не озвучит сами скобки, однако курсор будет между ними.

Поля работают как формулы в Excel: код поля соответствует формуле, а значение поля – значению этой формулы. Чтобы переключать режимы отображения кодов полей и их значений, следует пользоваться клавиатурной командой Alt +F9.

Синтаксис кода поля выглядит следующим образом:

{ИМЯ Свойства ключи}.

Имя – это Имя, которое отображается в списке имен полей в диалоговом окне вставки поля. Например, FILENAME.

Свойства – это Любые инструкции или переменные, используемые в данном поле. Не все поля имеют свойства.

Ключи – это необязательные параметры, доступные для данного поля. Например, ключи форматирования.

Для того, чтобы изменить параметры работы какого-либо поля, можно поступать по следующему алгоритму:

1. Установите курсор на поле в текстовом документе Word, которое необходимо изменить.

2. Раскройте контекстное меню.

3. В контекстном меню выберете команду «Изменить поле...».

4. В открывшемся диалоге можно установить необходимые параметры работы поля. Например, если было выбрано поле Date, то в диалоге можно будет изменить формат представления даты.

5. Завершите работу с диалогом нажав кнопку «ОК».

Для изменения некоторых полей необходимо отобразить код поля. Чтобы отобразить коды полей во всем документе, введите клавиатурную команду Alt +F9.

Некоторые поля изменяются в собственных диалоговых окнах. Например, если контекстное меню вызвать находясь на поле гиперссылки и выбрать команду «Изменить гиперссылку...», откроется диалоговое окно Изменение гиперссылки.

Обратите внимание, что по умолчанию значения полей в документе Word не отличаются по виду от остального содержания документа, так что при визуальном чтении пользователь не замечает, что часть содержимого документа располагается в поле. Однако поля можно отображать и с затенением, чтобы выделить их в документе визуально.

Чтобы отформатировать значения полей, можно применить к нему те же приемы форматирования, что и к обыкновенному фрагменту текста. Например, если выделить значение поля и ввести команду Ctrl +B, то это значение примет полужирное начертание.

Однако, форматирование, примененное к значению поля, может быть потеряно при обновлении поля. Чтобы сохранить форматирование, следует добавить к коду поля ключ * MERGEFORMAT. При вставке полей с помощью диалогового окна «Поле» этот ключ добавляется по умолчанию.

Отформатировать значение поля можно также с помощью соответствующих ключей, добавив их к коду поля. Обратите внимание, что ключи формата сохраняют формат значений при обновлении полей.

Приведем два примера использования ключей с соответствующими параметрами.

Ключ формата * определяет способ отображения значений полей. Например, ключ * Caps преобразует первую букву каждого слова в прописную. Обратите внимание, что ключ * может иметь несколько значений. Так, например, помимо значения Caps, выше уже говорилось о значении MERGEFORMAT.

Ключ формата даты/времени \@ определяет способ отображения даты или времени. Например, если отредактировать код поля DATE следующим образом

{DATE \@ «dddd, d MMMM, уууу»}

То поле вернет значение в виде
«воскресенье, 30 января, 2022».

Для создания ключа формата даты/времени используется сочетание следующих значений:

день – d;

месяц – M;

год – у;

часы – h;

минуты – m.

В значение ключа можно также включать текст, знаки препинания и пробелы.

Обратите внимание, что в обозначении месяца буква «M» должна быть прописной в отличие от строчной буквы «m» в обозначении минут.

При формировании ключа необходимо учитывать следующие обозначения:

M – Число без начального нуля (для первых девяти месяцев). Например, июль отображается как 7.

MM – Число с начальным нулем (для первых девяти месяцев). Например, июль отображается как 07.

MMM – Сокращенное название месяца (из трех букв). Например, июль отображается как «Июл».

MMMM – Полное название месяца.

Буква «d» отображает число месяца или день недели. Буква может быть как строчной, так и прописной.

D – Число без начального нуля (для первых девяти дней). Например, шестой день месяца отображается как 6.

dd – Число с начальным нулем (для первых девяти дней). Например, шестой день месяца отображается как 06.

ddd – Сокращенное название дня недели. Например, вторник отображается как «Вт».

dddd – полное название дня недели.

Буква «у» используется для отображения года двумя или четырьмя цифрами. Буква может быть как строчной, так и прописной.

уу – Две цифры с начальным нулем (для лет с 01 по 09). Например, 1999 отображается как 99, а 2006 отображается как 06.

уууу – Год отображается четырьмя цифрами.

В текстовом процессоре Word можно автоматизировать часто выполняемые задачи с помощью макросов. Макрос – это набор команд и инструкций, группируемых вместе в виде единой команды для автоматического выполнения определенного действия. При частом использовании одной и той же последовательности команд ее удобно представить в виде макроса.

Существует два основных способа создания макроса:

1. Записать последовательность команд в макрос;
2. Написать макрос на языке программирования Visual Basic for Application (VBA).

Готовый макрос можно запускать с помощью клавиатурной команды, выбирать из списка макросов или с помощью кнопки на панели быстрого доступа. Способ запуска зависит от настроек макроса.

Приведем алгоритм создания макроса, который будет запускаться с помощью клавиатурной команды:

1. Запустите текстовый процессор word.
2. На вкладке «Вид» установите фокус на разделённую кнопку «Макросы».
3. Раскройте список команд разделенной кнопки «Макросы» введя команду Alt +Стрелка вниз.
4. Выберите команду «Запись макроса...».
5. В раскрывшемся диалоге в первом поле редактирования (курсор будет на нем) введите имя макроса. Желательно придумать такое имя, чтобы впоследствии можно было понять какую задачу решает данный макрос. В данном диалоге есть также поле редактирования «Описание» в которое можно ввести краткое описание создаваемого макроса.

6. Перейдите на кнопку «Клавишам» и нажмите ее. Затем в появившемся диалоге «Настройка клавиатуры» в поле редактирования

введите сочетание клавиш, с помощью которого будет запускаться создаваемый макрос и нажмите кнопку «Назначить». Перед тем, как вводить сочетание клавиш, необходимо убедиться, что оно не используется.

7. Чтобы создаваемый макрос можно было использовать во всех новых документах, убедитесь, что в комбинированном списке «Макрос доступен» установлено значение «Во всех документах (Normal.dotm)».

8. Нажмите кнопку «ОК» для начала записи макроса.

9. После выполнения указанных выше действий, Word перешел в режим записи команд. Теперь необходимо без ошибок (поскольку они также будут записаны в макрос) выполнить все команды, которые должны быть объединены в макрос.

10. Для завершения записи макроса необходимо активировать вкладку «Вид», перейти к разделенной кнопке «Макросы», раскрыть список ее команд и выбрать команду «Остановить запись».

Макрос создан и может быть вызван с помощью комбинации клавиш, указанной при выполнении пункта 6 приведенного алгоритма.

Перед тем, как начать запись макроса, необходимо четко представлять какие именно команды будут в него записаны. Также следует убедиться, что перед началом записи созданы именно те условия, которые будут иметь место при его выполнении. Например, в окне редактирования должен быть текст и курсор находится в определенном его месте, в зависимости от решаемой макросом задачи.

Запустить макрос на выполнение можно также выбрав его из списка. Для этого следует активировать вкладку «Вид», перейти к разделенной кнопке «Макросы» и в списке ее команд выбрать «Макросы». В раскрывшемся диалоге курсор будет сразу находится на списке макросов. Также в этом диалоге есть кнопки, позволяющие отладить, изменить и удалить выбранный макрос.

Второй способ создания макросов требует определенных знаний в области программирования. Этот способ дает возможность создавать достаточно сложные макросы путем написания программы на языке Visual Basic for Application (VBA). Чтобы воспользоваться этим способом следует активировать вкладку «Вид», перейти к разделен-

ной кнопке «Макросы», в списке ее команд выбрать «Макросы» и в открывшемся диалоге нажать кнопку «Создать». Откроется окно, в котором надо написать программу, реализующую желаемый макрос. Подробнее на этом способе здесь мы останавливаться не будем, поскольку он достаточно сложен и требует дополнительных знаний.

Контрольные вопросы

1. Какие значения термина «поле» вы знаете в связи с текстовым процессором Word?
2. Что такое код поля?
3. Из каких элементов состоит код поля?
4. Зачем нужны ключи в коде поля?
5. Как можно вставить поле в текстовый документ Word?
6. Что такое макрос в текстовом процессоре Word?
7. Зачем нужны макросы?
8. Какие способы создания макроса вам известны?
9. Как можно запустить созданный макрос?
10. Опишите алгоритм записи макроса.
11. На каком языке программирования можно написать макрос?

§7.6. Режим быстрой навигации JAWS for Windows

Режим быстрой навигации весьма удобен для изучения больших по объёму текстов, выполненных с учетом всех правил подготовки текстового документа Word. Быстрая навигация по текстовому документу осуществляется за счёт особого функционала программы невидимого доступа к информации, который поддерживают обе популярные в настоящее время программы этого класса JAWS и NVDA. Этот режим основан на использовании так называемых «Клавиш быстрой навигации». Подчеркнем, что этот режим будет удобен для работы с документом Word, выполненным по всем правилам подготовки, т.е. заголовки должны иметь соответствующий уровень, клавиша Enter использовалась только для завершения абзаца и т.д.

Обратите внимание, что при включении данного режима невозможно внести в текст какие-либо изменения, документ доступен

только для чтения. Т.е. текст можно только читать, для его редактирования или форматирования режим «Клавиш быстрой навигации» следует выключить.

Включается и выключается режим «Клавиш быстрой навигации» клавиатурной командой Ins +Z.

Для навигации по документу Word используются следующие клавиатурные команды режима «Клавиш быстрой навигации»:

- Перейти к следующему предложению с грамматической ошибкой – A;
- Перейти к предыдущему предложению с грамматической ошибкой – Shift +A;
- Перейти к следующей орфографической ошибке – M;
- Перейти к предыдущей орфографической ошибке – Shift +M;
- Перейти к следующему комментарию – N;
- Перейти к предыдущему комментарию – Shift +N;
- Перейти к следующему примечанию – D;
- Перейти к предыдущему примечанию – Shift +D;
- Перейти к следующей сноске – O;
- Перейти к предыдущей сноске – Shift +O;
- Перейти к следующему полю – F;
- Перейти к предыдущему полю – Shift +F;
- Перейти к следующему исправлению – R;
- Перейти к предыдущему исправлению – Shift +R;
- Перейти к следующему графическому элементу – G;
- Перейти к предыдущему графическому элементу – Shift +G;
- Перейти к следующему заголовку – H;
- Перейти к предыдущему заголовку – Shift +H;
- Перейти к следующему списку – L;
- Перейти к предыдущему списку – Shift +L;
- Перейти к следующей таблице – T;
- Перейти к предыдущей таблице – Shift +T;
- Перейти на следующую страницу – Пробел;
- Перейти на предыдущую страницу – BackSpace;
- Читать следующий абзац – P;
- Читать предыдущий абзац – Shift +P;

- Перейти к следующему разделу – S;
- Перейти к предыдущему разделу – Shift +S.

Для перехода по заголовкам различных уровней следует использовать цифры верхнего ряда, соответствующие уровню заголовка. Т.е. для перехода к следующему заголовку первого уровня используют цифру 1, для перехода к заголовку второго уровня цифру 2 и т.д. Для перехода к предыдущему заголовку к соответствующей цифре добавляют клавишу Shift.

При изучении большого по объёму документа удобно начинать с перехода по заголовкам с помощью клавиши H. Программа невидимого доступа будет озвучивать заголовок и называть его уровень. После знакомства со структурой документа можно переходить к его последовательному чтению по абзацам (клавиша P) или по строкам (стрелка вниз).

При практической работе могут быть полезны следующие клавиатурные команды программы JAWS for Windows:

- Ins +V – Задать быстрые настройки;
- Ctrl +Ins +5 (5 на дополнительной клавиатуре) – Озвучить текущее поле;
- Ins +Del – Озвучить строку и столбец в позиции курсора;
- Alt +Ctrl +I – Переключить режим ввода с вставки на замену;
- Ctrl +Ins +V – Озвучить номер используемой версии MS Word;
- Ctrl +Ins +Home – Перевести фокус в первое поле формы;
- Ins +F5 – Вывести список полей;
- Ctrl +Ins +t – Вывести список таблиц;
- Alt +Shift +L – Вывести список слов с ошибками;
- Ins +Shift +G – Вывести список грамматических ошибок;
- Ins +Shift +R – Вывести список исправлений документа;
- ALT +Ins +B – Вывести список закладок;
- Ctrl +Shift +O – Вывести список объектов, таких, как медиаклипы и текстовые блоки;
- Ctrl +Shift +апостроф – Вывести список комментариев корректоров;
- Ins +Shift +f – Вывести список сносок в документе;
- Ins +Shift +e – Вывести список примечаний в документе;

- Ins +F6 – Вывести список заголовков в документе;
- Ins +F7 – Вывести список гиперссылок в документе;
- Ins +W – Вывести список «горячих» клавиш Word;
- Ctrl +ALT +a – Озвучить язык ввода;
- Alt +Shift +апостроф – Озвучить комментарий, ссылка на который находится в позиции курсора;
- Alt +Shift +e – Озвучить сноску или примечание, ссылка на которые находится в позиции курсора;
- Ctrl +Ins +R – Озвучить исправление, ссылка на которое находится в позиции курсора;
- Win +точка с запятой – Показать комментарии, сноски, примечания или исправления в окне виртуального просмотра;
- Ctrl +Win +k – Пометить место в документе;
- Alt +Win +k – Вернуться к отмеченному месту в документе.

Контрольные вопросы

1. Для чего используется режим клавиш быстрой навигации?
2. В чём особенности режима клавиш быстрой навигации?
3. Как включить режим клавиш быстрой навигации?
4. Каким требованиям должен соответствовать текстовый документ, чтобы использование клавиш быстрой навигации было максимально эффективным?
5. Какие команды режима клавиш быстрой навигации используются для перехода к следующим структурным элементам текстового документа:
 - А) Заголовок;
 - Б) Абзац;
 - В) Таблица;
 - Г) Список;
 - Д) Строка?
6. Как в режиме клавиш быстрой навигации переходить по заголовкам заданного уровня?

Глава 8

Табличный процессор Excel и программы невизуального доступа к информации

Материал этой главы предполагает наличие у обучающихся начальных теоретических сведений и практических навыков работы в табличном процессоре Excel. Ниже приведены основные клавиатурные команды для работы в этой программе с помощью JAWS for Windows, а также комбинации точек для ввода некоторых символов с помощью брайлевского дисплея.

Перед изучением материала главы ответьте на вопросы, приведённые ниже списка команд. При необходимости следует вернуться к материалу по работе с табличным процессором Excel за предыдущие классы.

Клавиатурные команды для работы в табличном процессоре Excel:

- Озвучить список основных сочетаний клавиш Excel – Ins +W;
- Озвучить координаты текущей ячейки – Ins +C.
- Прочитать формулу ячейки – Ins +Ctrl +F2.
- вывести формулу ячейки в Окне виртуального просмотра – Ins +Ctrl +F2 дважды быстро;
- Озвучить информацию в выделенных ячейках – Shift +Ins +стрелка вниз;
- Прочитать заголовок строки – Ins +Alt +Shift +R;
- Прочитать заголовок столбца – Ins +Alt +Shift +C;
- Прочитать итоговое значение строки – Ins +Del;
- Прочитать итоговое значение столбца – Ins +NumPadEnter;
- Задать диапазон заголовков строк – Ins +Ctrl +Alt +r;
- Задать диапазон заголовков столбцов – Ins +Ctrl +Alt +c;
- задать до 10 контрольных ячеек для текущего листа – Ins +Shift +Цифры от 1 до 0 в цифровом ряду;
- Озвучить любую из десяти контрольных ячеек для данного листа – Alt +Shift +Цифры от 1 до 0 в цифровом ряду;
- Перейти в контрольную ячейку – Ctrl +Shift +m;

- вернуться в последнюю ячейку, которая имела фокус перед переходом в контрольную ячейку – Ctrl +Shift +` (обратный апостроф);
- вывести список ячеек с комментариями на текущем листе – Ctrl +Shift +' (апостроф);
- Прочитать комментарий ячейки, если он видим – Alt +Shift +' (апостроф);
- вывести список ячеек с формулами – Ins +Shift +f;
- вывести список ячеек с данными, видимых в активном Окне, – Ctrl +Shift +D;
- вывести список ячеек с данными в текущей строке – Ins +Shift +R;
- вывести список ячеек с данными в текущем столбце – Ins +Shift +C;
- Озвучить адрес ячейки, содержащей гиперссылку – Alt +Shift +H;
- описать границу активной ячейки – Alt +Shift +B;
- Озвучить состояние сетки в активном Окне – Alt +Shift +G;
- Показать содержимое объекта в текстовом поле – Ctrl +Shift +t;
- Установить маркер ячейки – Win +Ins +k;
- Вернуться к маркеру ячейки на текущем листе – Alt +Win +k;
- Вернуться к следующему доступному маркеру ячейки на следующем листе – Ctrl +Win +k;
- Вернуться к предыдущему доступному маркеру ячейки на предыдущем листе – Ctrl +Win +Shift +k;
- Вывести список маркеров ячеек – Ctrl +ALT +Win +k;
- Переход на новую строку внутри ячейки – ALT +Enter;
- Завершение ввода в текущей ячейке и переход к ячейке выше – Shift +Enter;
- Вывод диалогового окна «Удаление» для удаления выделенных ячеек – Ctrl +/- (минус);
- Повторение последнего действия, если это возможно – Ctrl +Y;
- Отмена последнего действия – Ctrl +Z.

Комбинации точек для ввода символов с помощью клавиатуры Перкинса брайлевского дисплея:

> (Больше) – точки 45;

- < (Меньше) – точки 56;
- = (Равно) – точки 123456;
- ~ (Тильда) – точки 12456;
- @ (Собака) – точки 3457;
- # (решётка) – точки 3456;
- \$ (Доллар) – точки 467;
- % (Процент) – точки 146;
- ^ (Крышка) – точки 4578;
- & (Амперсанд) – точки 1234678;
- * (Звездочка) – точки 357;
- / (Косая черта) – точки 34;
- \ (Обратная косая черта) – точки 3478.

Контрольные вопросы

1. Как можно запустить программу Excel?
2. Как следует работать с ленточным меню?
3. Что такое динамическая таблица Excel?
4. Как можно перемещаться по таблице Excel?
5. Что можно вводить в ячейки таблицы Excel?

§8.1. Форматирование таблицы

В ячейках рабочего листа Excel может содержаться самая различная информация: текст, числа, время, дата, формула, функция. Для правильного отображения данных необходимо устанавливать соответствующий формат ячеек. По умолчанию каждая ячейка имеет формат «Общий». Изменить формат ячейки можно с помощью многостраничного диалога «Формат ячеек», который вызывается через контекстное меню или клавиатурной командой Ctrl + 1.

Обратите внимание, что в некоторых случаях формат ячейки «Общий» может привести к неправильной работе таблицы. Старайтесь устанавливать формат ячеек, соответствующий тем данным, которые в ней будут находиться.

Пусть, например, необходимо установить у всех ячеек столбца А формат «Дата», для хранения в его ячейках календарных дат. Из-

менить формат ячейки на листе Excel можно с помощью следующего алгоритма:

1. Установите фокус в любую ячейку столбца А и введите команду Ctrl +пробел для его выделения. Помните, что после выделения какой-либо области перемещение фокуса по ячейкам приведёт к снятию выделения.

2. Введите команду Ctrl +1 для вызова диалога «Формат ячеек». Фокус окажется на вкладке «Число».

3. На этой вкладке используя клавишу Tab перейдите в поле «Числовые форматы».

4. Перемещаясь по списку стрелками вертикального управления курсором, установите фокус на формат «Дата».

5. Нажав клавишу Tab перейдите в поле «Тип» и стрелками вертикального управления курсором выберите из списка желаемый тип представления даты, например, с текстовым представлением названия месяца.

6. Завершите выбор желаемого формата нажатием клавиши Enter.

Теперь введите в любую ячейку столбца А какую-либо дату в виде 01.03.2022 и убедитесь, что она будет представлена в желаемом формате. Обратите внимание, что теперь при попытке ввода чисел в ячейках столбца А Excel будет пытаться интерпретировать их как даты.

В качестве еще одного примера использования диалога «Формат ячеек» рассмотрим алгоритм объединения ячеек и выравнивания в них текста. Пусть, например, необходимо объединить ячейки А1, В1 и С1 в одну, после чего выровнять текст в объединённых ячейках по центру по вертикали и по горизонтали. Приведем алгоритм выполнения этой операции в табличном процессоре Excel:

1. Выделите ячейки А1, В1 и С1 (о том, как это сделать было рассказано в предыдущих классах).

2. Вызовите диалог «Формат ячеек» командой Ctrl +1.

3. Стрелками горизонтального управления курсором или командой Ctrl +Tab перейдите на вкладку «Выравнивание».

4. Перемещаясь по этой вкладке клавишей Tab найдите комбинированный список «по горизонтали» и стрелками вертикального управления курсором выберите значение «по центру».

5. Далее клавишей Tab перейдите к комбинированному списку «по вертикали» и стрелками вертикального управления курсором выберите значение «по центру».

6. Продолжая перемещаться по элементам управления данной вкладки клавишей Tab найдите флажок «объединение ячеек» и установите его клавишей пробел.

7. Завершите выполнение операции вводом команды Enter, что эквивалентно нажатию кнопки «Ок».

Теперь три ячейки объединены в одну и информация, вводимая в эту большую ячейку, будет центрироваться по вертикали и по горизонтали.

Обратите внимание, что информация в ячейке может выравниваться не только по горизонтали (как в текстовом редакторе), но и по вертикали, т.е. между верхней и нижней границами ячейки.

Заметим, что реализовать приведённый выше алгоритм (как и все алгоритмы этой книги) можно и с помощью команд брайлевского дисплея.

Для форматирования шрифта используется вкладка «Шрифт» того же диалога «Формат ячеек». С помощью клавиатурной команды Ctrl + Shift + F можно открыть этот диалог так, чтобы фокус оказался сразу на вкладке «шрифт».

Приведем список команд стандартной клавиатуры, которые могут быть полезны при форматировании ячеек таблицы:

- Вызов диалогового окна «Формат ячеек» – Ctrl + 1;
- Форматирование шрифтов с помощью диалогового окна «Формат ячеек» – Ctrl + Shift + F или Ctrl + Shift + P;
- Переключение между выводом в листе значений ячеек и формул – Ctrl + ` (акцент);
- Применение или удаление полужирного начертания – Ctrl + B или Ctrl + 2;
- Применение или удаление курсивного начертания – Ctrl + I или Ctrl + 3;

- Подчеркивание текста или удаление подчеркивания – Ctrl +U или Ctrl +4;
- Применение или удаление зачеркивания – Ctrl +5;
- Скрытие выделенных строк – Ctrl +9;
- Скрытие выделенных столбцов – Ctrl +0;
- Применение общего числового формата – Ctrl +Shift +~ (тильда);
- Применение денежного формата с двумя десятичными знаками (отрицательные числа отображаются в круглых скобках) – Ctrl +Shift +\$ (доллар);
- Применение процентного формата без дробной части – Ctrl +Shift +% (процент);
- Применение экспоненциального числового формата с двумя десятичными знаками – Ctrl +Shift +^ (крышка);
- Применение формата даты с указанием дня, месяца и года – Ctrl +Shift +# (решётка);
- Применение формата времени с отображением часов и минут и индексами AM или PM – Ctrl +Shift +@ (собака);
- Применение числового формата с двумя десятичными знаками, разделителем разрядов и знаком минус (-) для отрицательных значений – Ctrl +Shift +! (восклицательный знак).

При работе с таблицами Excel без визуального контроля, следует учитывать, что каждая ячейка обладает определёнными фиксированными размерами. Таким образом, в её видимую часть большой текст может не поместиться, однако JAWS for Windows будет читать всё содержимое ячейки. При этом не поместившаяся часть данных может либо обрезаться и будет отсутствовать на экране, либо, если следующая ячейка на строке не занята, наложится на неё. Для визуального восприятия информации подобный эффект нежелателен.

Для получения информации о содержимом и внешнем виде ячейки следует использовать клавиатурную команду Ins +Tab повторенную быстро дважды. В отдельном окне будет выведена сводная информация по данной ячейке. В этом окне может присутствовать информация о визуальном расположении данных. Например, может присутствовать текст: «перекрывает справа возле C1». Это означает, что содержимое ячейки не помещается в её видимых границах, и

оно наложилась на последующую ячейку B1, так как она была пуста. Если бы в B1 присутствовали данные, то не поместившееся содержимое A1 было обрезано. В этом случае в данном диалоге был бы текст: «обрезано справа на B1».

Для исправления подобных ошибок в форматировании таблицы, следует использовать функции автоподбора высоты строки и ширины столбца. Таким образом можно выровнять сетку таблицы по её содержимому и устранить наложение и обрезание данных в ячейках. Подробно этот функционал здесь рассматриваться не будет. При необходимости вы сможете освоить его самостоятельно.

Контрольные вопросы

1. Зачем нужны различные типы данных в Excel?
2. Какие типы данных в Excel вы знаете?
3. Как можно изменить тип данных в ячейке листа Excel?
4. Какой тип данных имеют ячейки таблицы Excel по умолчанию?
5. Как объединить несколько ячеек в одну?
6. Как можно выравнивать информацию в ячейке? Чем отличается выравнивание текста в ячейке от выравнивания текста в текстовом редакторе?
7. Как изменить шрифтовые характеристики текста в ячейке?
8. Как можно узнать, помещаются данные в ячейку или нет?
9. Что произойдет, если данные не помещаются в границах ячейки?
10. Как можно обеспечить соответствие границ ячейки и видимого размера размещенных в ней данных?

§8.2. Автоматический ввод данных

Как вы уже знаете, для ввода любой информации в ячейку рабочего листа книги Excel в неё следует установить фокус и набирать нужные данные на стандартной клавиатуре или на брайлевском дисплее. Для перехода на новую строку внутри ячейки следует использовать команду Alt +Enter. Команда Shift +Enter в табличном процессоре Excel завершит ввод в данную ячейку и переведёт фокус в ячейку над ней.

Чтобы отредактировать уже введенные в ячейку данные, необходимо установить в неё фокус и ввести команду F2, откроется поле ре-

дактора с отображенным в нём содержимом активной ячейки. В этом поле можно пользоваться всеми приемами локального редактирования текста. Для завершения редактирования информации в ячейке следует нажать клавишу Enter.

Достаточно часто на практике встречаются ситуации, в которых требуется на рабочем листе заполнить большой диапазон ячеек одинаковыми или связанными между собой по определённым правилам данными. Табличный процессор Excel имеет в своем арсенале большое количество средств, облегчающих ввод данных. Например, если необходимо ввести в несколько ячеек один и тот же текст, эти ячейки следует выделить, затем не снимая выделения ввести с клавиатуры нужный текст и дать клавиатурную команду Ctrl +Enter. В каждой из выделенных ячеек появится введенный текст.

Приведём подробный алгоритм решения этой задачи:

1. Установите фокус в первую ячейку области на рабочем листе Excel, в которой должны располагаться одинаковые данные.

2. Удерживая клавишу Shift стрелками управления курсором выделите эту область.

3. Введите необходимые данные (например, цифру 1) в последнюю ячейку выделенной области (именно в ней будет находиться фокус). Обратите внимание, что перемещать фокус после выделения уже нельзя.

4. Введите команду Ctrl +Enter. JAWS for Windows произнесёт координаты выделенного диапазона и прочтёт введенные данные.

После выполнения описанного алгоритма необходимый диапазон будет заполнен введенными данными.

Также часто на практике возникает необходимость заполнить диапазон ячеек различными данными, но связанными друг с другом по определенному правилу. Например, получить в столбик даты всех понедельников, начиная с 28 февраля 2022 до конца года.

Для решения этой задачи можно воспользоваться следующим алгоритмом:

1. На рабочем листе Excel установите фокус в первую ячейку выделяемого диапазона (например, в A1).

2. Нажмите клавишу F5 и в поле комбинированного списка «Ссылка» Раскрывшегося диалога «Переход» введите координаты выделяемого диапазона (например, A1:A100). После нажатия клавиши Enter, JAWS произнесет координаты выделенного диапазона. Будьте внимательны, поскольку любое неправильное действие может привести к снятию выделения!

3. Вызовете диалог «Формат ячеек» введя команду Ctrl +1.

4. На вкладке «Число» в списке числовых форматов стрелками вертикального управления курсором выберите «Дата» и нажмите Enter. При необходимости можно изменить формат отображения даты.

5. В первой ячейке диапазона (именно в ней будет находиться фокус после выбора нужного типа данных) введите начальную дату 28.02.2022. и повторите выделение диапазона в соответствии с пунктом 2 данного алгоритма, поскольку при вводе даты выделение будет снято.

6. Активируйте вкладку «Главная» выбрав ее на ленте или введя Alt +Я.

7. На нижней ленте выберите группу «Редактирование» используя команду Ctrl +Стрелка вправо или введя последовательно русские буквы «З» и «А».

8. Стрелками вертикального управления курсором выберите команду «Прогрессия...». Для выбора этой команды можно также использовать русскую букву «Г».

9. В раскрывшемся диалоге «Прогрессия» в поле «Шаг» укажите значение 7, а в поле «Предельное значение» укажите 31.12.2022 и нажмите Enter.

Обратите внимание, что в этом алгоритме был использован другой способ выделения диапазона ячеек, который более удобен для выделения большого количества ячеек. Приведённые выше алгоритмы не зависят от способа выделения диапазона ячеек.

В диалоговом окне «Прогрессия» есть поле с четырьмя радиокнопками, используя которые можно заполнять выделенный диапазон числовыми значениями, образующими арифметическую или геометрическую прогрессию.

Приведём несколько команд стандартной клавиатуры, используемых при вводе информации в ячейку рабочего листа Excel:

- Завершение ввода и переход в ячейку ниже – Enter;
- Завершение ввода и переход в ячейку выше – Shift +Enter;
- Переход на новую строку внутри ячейки – Alt +Enter;
- Вставка текущего времени – Ctrl +Shift +: (ДВОЕТОЧИЕ);
- Вставка текущей даты – Ctrl +; (ТОЧКА С ЗАПЯТОЙ);
- Вставка пустых ячеек с помощью диалогового окна «Вставка» – Ctrl +Shift ++ (плюс).

Контрольные вопросы

1. Как вводить и редактировать данные в ячейке рабочего листа Excel?
2. Какие способы выделения диапазона ячеек на рабочем листе Excel вы знаете?
3. Как можно выделить всю строку рабочего листа Excel? Весь столбец?
4. Опишите алгоритм заполнения заданного диапазона на рабочем листе Excel одинаковыми данными.
5. Опишите алгоритм заполнения заданного диапазона на рабочем листе Excel связанными данными.
6. Как можно ввести в ячейку рабочего листа Excel текущую дату? Время?

§8.3. Формулы и функции

В отличие от статических таблиц, создаваемых текстовым процессором Microsoft Word, таблицы Excel являются динамическими. По мере ввода информации, автоматически производятся различные вычисления и обработка данных. Excel это не просто таблица для ввода чисел, в этой программе можно считать суммы в строках и столбцах, вычислять платежи по ипотеке, решать математические, инженерные и статистические задачи, а также находить наиболее благоприятные варианты, зависящие от заданных переменных значений.

В Excel все эти возможности реализованы с помощью формул, которые можно помещать в ячейках. По этим формулам выполняются

вычисления и другие действия с данными на листе. Формула всегда начинается со знака равенства (=), после которого можно вводить числа, математические операторы (например, знаки + и - для сложения и вычитания) и встроенные функции Excel, значительно расширяющие возможности формул.

Формула Excel может содержать следующие элементы:

1. Функции. Например, функция сегодня () – возвращает сегодняшнюю дату.
2. Ссылки. Например, A2 возвращает значение ячейки A2.
3. Константы. Числа или текстовые значения, введенные непосредственно в формулу, например число 2.
4. Операторы. Например, оператор ^ (крышка) применяется для возведения числа в степень, а * (звездочка) — для умножения.

Константа представляет собой готовое (не вычисляемое) значение, которое всегда остается неизменным. Например, дата 09.10.2008, число 210 и текст «Прибыль за квартал» являются константами. Выражение или его значение константами не являются. Если формула в ячейке содержит константы, а не ссылки на другие ячейки (например, имеет вид =30 +70 +110), значение в такой ячейке изменяется только после редактирования формулы. Лучше помещать такие константы в отдельные ячейки, где их можно будет легко изменить при необходимости, а в формулах использовать ссылки на эти ячейки.

Операторы задают обычные арифметические действия, производимые над элементами формулы. В Excel соблюдается обычный порядок вычислений: скобки, возведение в степень, умножение и деление и, наконец, сложение и вычитание.

Арифметические операторы служат для выполнения базовых арифметических операций, таких как сложение, вычитание, умножение или деление чисел. Результатом операций являются числа. Приведем список арифметических операторов Excel:

- + (плюс) – Сложение;
- - (минус) – Вычитание;
- * (звездочка) – Умножение;
- / (косая черта) – Деление;
- % (знак процента) – Процент;

- ^ (крышка) – Возведение в степень.

Кроме арифметических операторов в Excel существуют еще операторы сравнения, операторы объединения текста и ссылки.

Операторы сравнения используются для сравнения двух значений. Результатом применения этого оператора является логическое значение: ИСТИНА либо ЛОЖЬ. Операторов сравнения всего шесть: = (равно), > (больше), < (меньше), >= (больше или равно), <= (меньше или равно), <> (не равно).

Оператор объединения текста & (амперсанд или «энд») используется для объединения (конкатенации) нескольких текстовых строк в одну. Например, результатом формулы =»Ветер»&»ок» будет «Ветерок». Если в ячейку A1 записать «Фамилия», а в B1 «Имя», формула =A1&», «&B1 вернет «Фамилия, Имя».

Многие функции Excel могут обрабатывать данные из диапазона ячеек. Для определения ссылок на диапазоны ячеек можно использовать операторы:

- : (двоеточие) - Оператор диапазона, который образует одну ссылку на все ячейки, находящиеся между первой и последней ячейками диапазона, включая эти ячейки. Например, B5:B15 укажет на все ячейки с b5 по b15 включительно;

- ; (точка с запятой) – Объединяет несколько ссылок в одну. Например, =СУММ(B5:B15;D5:D15) вычислить сумму всех 22-ух ячеек с b5 по b15 и с d5 по d15.

Функции Excel представляют собой заранее определенные формулы, выполняющие вычисления над заданными величинами – аргументами. Чтобы просмотреть весь список доступных функций Введите клавиатурную команду Shift +F3. Откроется диалоговое окно «Вставка функции».

Диалоговое окно «Вставка функции» облегчает ввод формул, содержащих функцию. При выборе функции в диалоге «Вставка функции» Excel запускает мастер функций. В нем перечислены имя функции, все ее аргументы, описание функции и каждого аргумента.

Функции в ячейки рабочего листа можно вводить и с клавиатуры не пользуясь мастером. При вводе функций вручную следует придерживаться следующего порядка ввода:

- Знак равенства;
- Имя функции (русскими буквами);
- открывающая круглая скобка;
- Список аргументов, разделенных точками с запятой (или диапазоном адресов);
- Закрывающая круглая скобка.

Используемый аргумент должен возвращать допустимое для него значение. В качестве аргументов могут выступать ссылки на ячейки, константы, формулы и другие функции.

Имя функции, написанное с ошибкой, например =СУММА(A1:A10) вместо =СУММ(A1:A10), вернет ошибку «#ИМЯ?».

В некоторых случаях может потребоваться использовать функцию в качестве одного из аргументов другой функции. Вложенная функция, используемая в качестве аргумента, должна возвращать соответствующий этому аргументу тип данных. Например, если аргумент должен быть логическим, т. е. иметь значение ИСТИНА либо ЛОЖЬ, вложенная функция должна возвращать логическое значение. В противном случае Excel выдаст ошибку «#ЗНАЧ!». В формулах можно использовать до семи уровней вложенных функций.

Ниже перечислены некоторые из наиболее распространенных ошибок, допускаемых при вводе формул, а также способы избежать их совершения.

1. Убедитесь, что Все открывающие и закрывающие скобки в формуле согласованы, т.е. что каждой открывающей скобке соответствует закрывающая.

2. Для указания диапазонов используется двоеточие.

3. Указаны обязательные аргументы. У функции могут быть обязательные и необязательные аргументы (последние при описании синтаксиса заключаются в квадратные скобки).

4. Имена книг и листов заключены в одинарные кавычки. Если имена листов или книг, на ячейки в которых ссылается формула, содержат небуквенные символы, их нужно заключить в одинарные кавычки (апострофы).

5. Указан путь к внешним книгам. Внешние ссылки должны содержать имя книги и путь к ней.

6. Числа введены без форматирования. При вводе чисел в формулу нельзя указывать знак доллара, так как он используется для обозначения абсолютных ссылок.

Контрольные вопросы

1. Чем отличаются статические таблицы от динамических?
2. Какие задачи можно решать в табличном процессоре Excel?
3. С какого знака должна начинаться формула в Excel?
4. Какие элементы может содержать формула Excel?
5. Что такое константа?
6. Какие арифметические операторы вы знаете?
7. Что такое операторы сравнения?
8. Как работает оператор объединения?
9. Как создать ссылку на диапазон ячеек?
10. Что такое функции в Excel? Приведите примеры.
11. Как просмотреть список всех функций Excel?
12. Как следует вводить функцию в ячейку рабочего листа Excel?

§8.4. Абсолютные и относительные ссылки

Ссылка указывает на ячейку или диапазон ячеек рабочего листа и сообщает Excel, где находятся необходимые формуле значения или данные. С помощью ссылок можно использовать в одной формуле данные, находящиеся в разных частях листа, а также использовать значение одной ячейки в нескольких формулах. Можно задавать ссылки на ячейки разных листов одной книги либо на ячейки из других книг. Ссылки на ячейки других книг называются связями или внешними ссылками.

Как вы уже знаете, по умолчанию Excel использует стиль ссылок, в котором столбцы обозначаются латинскими буквами, а строки — цифрами.

Относительная ссылка в формуле, например A1, основана на относительной позиции ячейки, содержащей формулу, и ячейки, на которую указывает ссылка. При изменении позиции ячейки, содержащей формулу, изменяется и ссылка. При копировании формулы вдоль строк и вдоль столбцов ссылка автоматически корректирует-

ся. По умолчанию в новых формулах используются относительные ссылки. Например, при копировании относительной ссылки из ячейки В2 в ячейку В3 она автоматически изменяется с =А1 на =А2.

Рассмотрим пример таблицы, обрабатывающей информацию о количестве деталей, выпущенных тремя рабочими в течении недели.

Пример 1. Создать таблицу из пяти столбцов и семи строк. Все пять ячеек первой строки объединяются и в получившейся большой ячейке по её центру (по вертикали и по горизонтали) записывается название таблицы «Учет произведённой продукции за неделю». В ячейках второй строки по центру (по вертикали и по горизонтали) записываются заголовки каждого столбца: в ячейке А2 – «Дата», в В2 – «Первый рабочий», в С2 – «Второй рабочий», в D2 – «Третий рабочий», а в Е2- «Деталей в день». В пяти оставшихся строках в столбце А записываются календарные даты пяти рабочих дней текущей недели. В столбце В записывается количество деталей, произведённых первым рабочим. В столбце С – количество деталей, произведённых вторым рабочим, а в столбце D – количество деталей, произведённых третьим рабочим. В столбце Е должна вычисляться сумма деталей, произведённых всеми тремя рабочими в данный день.

Приведём алгоритм решения этой задачи:

1. Объедините пять ячеек первой строки с А1 по Е1, задайте выравнивание по центру по горизонтали и по вертикали, а затем введите в объединённую ячейку заголовков таблицы «Учет произведённой продукции за неделю» (о том как это сделать было подробно рассказано в параграфе 8.1.).

2. В ячейки с А2 по Е2 по центру введите соответствующие названия столбцов таблицы.

3. Выделите ячейки с А3 по А7 и используя диалог «Формат ячеек» установите в выделенных ячейках формат «Дата» и тип с текстовым названием месяца, например, «март 2022».

4. Заполните пять ячеек первого столбца датами рабочих дней текущей недели используя прогрессию (как это сделать было подробно рассказано в параграфе 8.2.).

5. Установите фокус в ячейку E3 (это верхняя ячейка столбца с вычисляемым количеством изготовленных в день деталей) и введите в неё формулу =СУММ(B3:D3). Завершите ввод данных нажатием клавиши Enter.

6. Скопируйте содержимое ячейки E3 в буфер обмена. Обратите внимание, что ячейка на которой находится фокус всегда является выделенной, поэтому достаточно только переместить фокус в ячейку E3 и ввести команду Ctrl +C.

7. Вставьте скопированную формулу последовательно в ячейки E4, E5, E6 и E7 с помощью команды Ctrl +V.

8. Заполните ячейки диапазона B3:D7 произвольными данными. В ячейках этого диапазона должны быть значения выработки отдельного рабочего за каждый рабочий день недели.

Изучите значения дневной выработки трёх рабочих за каждый день. Используя команду F2 изучите в каком виде скопировалась формула суммирования в ячейки с E4 по E7. Обратите внимание, что адреса суммируемых ячеек изменялись в соответствии со смещением формулы вниз по столбцу.

Наряду с изменяемыми относительными ссылками в Excel есть возможность создавать абсолютные неизменяемые ссылки. Абсолютная ссылка на ячейку в формуле, например \$A\$1, всегда ссылается на ячейку, расположенную в определенном месте. При изменении позиции ячейки, содержащей формулу, абсолютная ссылка не изменяется. При копировании формулы по строкам и столбцам абсолютная ссылка не корректируется. Для создания абсолютных ссылок надо использовать знак доллара. Например, при копировании абсолютной ссылки из ячейки B2 в ячейку B3 она остается прежней в обеих ячейках: =\$A\$1.

Смешанная ссылка содержит либо абсолютный столбец и относительную строку, либо абсолютную строку и относительный столбец. Абсолютная ссылка столбцов имеет вид \$A1, \$B1 и т.д. Абсолютная ссылка строки имеет вид A\$1, B\$1 и т.д. При изменении позиции ячейки, содержащей формулу, относительная ссылка изменяется, а абсолютная нет.

Пример 2. Создать таблицу Excel, реализующую стандартную таблицу умножения.

Приведём алгоритм решения этой задачи:

1. В первой строке рабочего листа в ячейках с B1 по K1 введите заголовки столбцов таблицы умножения – числа от 1 до 10.

2. В первом столбце в ячейках с A2 по A11 введите заголовки строк – числа от 1 до 10.

3. В ячейку B2 введите формулу $=\$A2*B\1 .

4. Скопируйте формулу из ячейки B2 в ячейки этой же строки с C2 по K2.

5. Скопируйте получившуюся таким образом строку из десяти формул (ячейки с B2 по K2) девять раз строго вниз на строки с третьей по одиннадцатую, т.е. сперва в ячейки с B3 по K3, затем в ячейки с B4 по K4 и т.д. до ячеек с B11 по K11.

Проверьте значения в ячейках получившейся таблицы умножения. Используя команду F2 изучите в каком виде скопировалась формула в различные ячейки. Обратите внимание, что адреса перемножаемых ячеек изменялись при копировании только частично.

В табличном процессоре Excel существует еще один вид ссылок – трёхмерные ссылки. Трёхмерные ссылки используются для анализа данных из одной и той же ячейки или диапазона ячеек на нескольких листах одной книги. Трёхмерная ссылка содержит ссылку на ячейку или диапазон, перед которой указываются имена листов, из которых необходимо брать данные, причём после указания листов перед адресом ячейки ставится восклицательный знак. Например, формула $=\text{СУММ}(\text{Лист2}:\text{Лист13}!\text{B5})$ суммирует все значения, содержащиеся в ячейке B5 на всех листах в диапазоне от Лист2 до Лист13 включительно.

Контрольные вопросы

1. Что такое ссылка в Excel?
2. Какие виды ссылок в Excel вы знаете?
3. Как выглядит обычная ссылка в Excel?
4. Как изменяется при копировании формулы относительная ссылка?

5. Как создаётся абсолютная ссылка?
6. Что такое смешанная ссылка?
7. Для чего служат трёхмерные ссылки?

§8.5. Управление листами

При создании книги Excel в ней присутствует только один лист. В ранних версиях табличного процессора в созданной книге сразу было три листа. Для добавления листов в книгу или переименования имеющихся удобно использовать команду **Ins +Shift +S**. Она раскроет вертикальное меню управления листами. Перемещаться по нему следует стрелками вертикального управления курсором, а выбирать нужную команду клавишей **Enter**. С помощью команд этого меню можно создавать, удалять, переименовывать, перемещать листы и выполнять некоторые другие операции с ними. Подчеркнём, что это меню управления листами формируется программой **JAWS for Windows**, а не программой **Excel**.

Активным в каждый момент является один лист загруженной книги. Для перехода к следующему листу можно использовать команду **Ctrl +PgDn**, а к предыдущему – **Ctrl +PgUp**. В списке листов, в соответствии с которым происходит переход к следующему или к предыдущему с помощью данных команд, листы расположены в том порядке, в котором были созданы. В этом списке порядок следования листов можно изменять, используя соответствующую команду меню управления листами.

Такие часто используемые команды меню управления листами, как «Вставить...» и «Переместить/скопировать...» вызывают соответствующие диалоговые окна. С помощью диалога «Вставить...» кроме листа можно вставить и некоторые другие объекты, но работа с ними выходит за рамки этой книги.

Обратите внимание, что при вставке нового листа имя ему присваивается автоматически (лист1, лист2 и т.д.). Поэтому после вставки нового листа его следует переименовать. Также при вставке нового листа полезно следить за тем, какой лист активен, учитывая, что вставляемый лист в списке окажется перед активным.

Переместить лист по списку можно с помощью диалога «Переместить/скопировать...». Для этого можно воспользоваться следующим алгоритмом:

1. С помощью команд Ctrl +PgUp и Ctrl +PgDn сделайте активным перемещаемый лист Excel.
2. Введите команду Ins +Shift +S для вызова меню управления листами.
3. В открывшемся меню выберите команду «Переместить/скопировать...».
4. В открывшемся диалоге фокус окажется в списке «Перед листом», в котором стрелками вертикального управления курсором следует выбрать лист, в позицию перед которым будет перемещен активный лист.
5. Завершите перемещение листа нажатием клавиши Enter.

Контрольные вопросы

1. Что такое лист Excel?
2. Сколько листов содержит при создании книга Excel?
3. Как осуществляется переход между листами?
4. Какой функционал предоставляет JAWS for Windows для управления листами?
5. Какой командой вызывается меню управления листами?
6. В какую позицию списка помещается создаваемый лист?
7. Опишите алгоритм перемещения листа.

Глава 9

Реляционные базы данных и программы невизуального доступа к информации

§9.1. Основные понятия баз данных

В учебниках информатики предлагается несколько различных определений понятия «база данных», при этом общепризнанная единая формулировка отсутствует. Приведём несколько близких по сути, но различающихся по формулировке определений:

- База данных – это представленная в объективной форме совокупность самостоятельных материалов (статей, расчётов, нормативных актов, судебных решений и иных подобных материалов), систематизированных таким образом, чтобы эти материалы могли быть найдены и обработаны с помощью электронной вычислительной машины (ЭВМ);

- База данных – это организованная в соответствии с определёнными правилами и поддерживаемая в памяти компьютера совокупность данных, характеризующая актуальное состояние некоторой предметной области и используемая для удовлетворения информационных потребностей пользователей;

- База данных – это совместно используемый набор логически связанных данных (и описание этих данных), предназначенный для удовлетворения информационных потребностей организации;

- База данных (БД) – это организованная совокупность данных и связей между ними, предназначенная для длительного хранения во внешней памяти ЭВМ, постоянного обновления и использования.

В этих определениях в той или иной форме присутствуют следующие отличительные признаки базы данных:

- База данных хранится и обрабатывается в вычислительной системе (т.е. любые внекомпьютерные хранилища информации такие, как архивы, библиотеки, картотеки и тому подобное базами данных не являются);

- Данные в базе данных логически структурированы (систематизированы) с целью обеспечения возможности их эффективного поиска и обработки в вычислительной системе (структурированность подразумевает явное выделение составных частей, связей между ними, а также типизацию элементов и связей, при которой с типом элемента соотносятся определённые допустимые операции);

- База данных включает схему, описывающую её логическую структуру в формальном виде.

Обратите внимание, что в соответствии с общей практикой, например, не называют базами данных файловые архивы или Интернет-порталы, хотя они в некоторой степени обладают признаками БД.

История баз данных, рассматривающая их в традиционном (современном) понимании, начинается с 1955 года, когда появилось программируемое оборудование обработки записей. Программное обеспечение того времени поддерживало модель обработки записей на основе файлов.

Сетевые базы данных появились в 60-е годы прошлого века. Операции над ними выполнялись в интерактивном режиме с помощью терминалов. Простые индексно-последовательные организации записей быстро развились к более мощной модели записей.

Следующий важный этап связан с появлением в начале 70-х годов реляционной модели данных, благодаря работам Эдгара Кодда. Работы Кодда открыли путь к тесной связи прикладной технологии баз данных с математикой и логикой. За свой вклад в теорию и практику Эдгар Ф. Кодд получил премию Тьюринга.

Сам термин «база данных» (англ. database) появился в начале 60-х годов. В широкое употребление в современном понимании термин вошёл лишь в 70-е годы XX века.

Базы данных предназначены для хранения и обработки всевозможной информации и в условиях современного информационного общества используются во всех областях деятельности человека. Вам уже знакомы такие примеры базы данных, как библиотечный каталог, электронный журнал успеваемости, списки обучающихся, описание товаров на каком-либо складе, телефонный справочник и

т.д. База данных, созданная на персональном компьютере, сохраняется на диске и обрабатывается системой управления базами данных (СУБД).

Система управления базами данных (СУБД)— это комплекс программ и языковых средств для формирования запросов, предназначенный для создания, ведения и использования баз данных. Системы управления базами данных на персональных компьютерах поддерживают в основном реляционную модель данных.

Реляционная модель данных – это набор таблиц, реализующих связи или отношения (relation – отношение), к которым применимы некоторые специальные операции. Это такие операции, как, например, операции объединения, пересечения, сложения, вычитания и другие. В рамках этой книги работа с подобными операциями изучаться не будет.

Реляционной называется такая база данных, в которой реализована реляционная (табличная) модель данных. Применение упомянутых выше операций в СУБД позволяет создавать базы данных из нескольких взаимосвязанных таблиц, выполнять различные операции над ними, организовывать связи между таблицами. Реляционные базы данных являются наиболее распространёнными на практике. Данные в них представляются в табличном виде.

Основными элементами структуры реляционной базы данных являются записи и поля. Коротко говоря, запись – это строка таблицы, а поле – это её столбец.

Каждая запись реляционной базы данных (строка таблицы) содержит информацию об отдельном объекте. Например, если база данных содержит информацию о библиотечном каталоге, то в записи (в строке таблицы) будет храниться вся информация об одной конкретной книге.

Каждое поле (столбец таблицы) – это определенная характеристика (свойство, атрибут) объекта. Используя тот же пример с библиотечным каталогом, полями будут: название книги, её автор, год издания, количество страниц, номер стеллажа, на котором она хранится и т.д.

Каждое поле имеет свой тип. Тип поля определяет какие именно значения могут быть в него внесены. С типами переменных вы сталкивались в программировании, а с типами ячеек таблицы вы работали в предыдущей главе при изучении табличного процессора Excel. В базе данных тип поля имеет тот же смысл.

Поскольку реляционные базы данных как правило реализуются в табличном процессоре, то основные типы полей будут теми же, что и в Excel:

- Текстовый;
- Числовой;
- Дата/Время;
- Логический;
- Денежный.

В следующих параграфах будут рассмотрены некоторые практические приёмы работы с реляционными базами данных в знакомом вам табличном процессоре Excel.

Контрольные вопросы

1. Что такое база данных? Существует ли строгое определение?
2. Какие отличительные признаки базы данных вы знаете?
3. Где хранится база данных?
4. Когда появились первые базы данных в современном смысле?
5. Приведите примеры базы данных.
6. Что такое СУБД?
7. Что такое реляционная база данных?
8. Что такое запись в реляционной базе данных? Приведите примеры.
9. Что такое поле в реляционной базе данных? Приведите примеры.
10. Зачем нужны различные типы полей?
11. Какие типы полей вы знаете?

§9.2. Табличное представление реляционной базы данных

Для решения практических задач табличный процессор Excel предоставляет удобное и эффективное средство работы с базами данных. Подготовленный пользователь может создать в Excel реляцион-

ную базу данных, отвечающую его потребностям, а также условиям комфортной работы с помощью программ невизуального доступа к информации. Таким образом, пользователь с глубоким нарушением зрения обладая определённой квалификацией способен создавать в Excel базы данных, отвечающие его бытовым потребностям и учебным задачам.

Реляционная база данных в Excel – это набор строк (список) в одной таблице, представляющий собой непрерывный диапазон ячеек (строк и столбцов). Т.е., база данных в Excel представляет собой некоторую таблицу на рабочем листе, состоящую из строк данных или записей. Строка такой таблицы является записью базы данных, включающей совокупность полей.

Каждая запись содержит информацию об отдельном объекте базы данных. Запись (строка) делится на поля – ячейки строки, соответствующие столбцам таблицы (полям базы данных). Одни и те же поля для различных записей предназначаются для данных одного типа.

Поле – это столбец таблицы, ячейками которого являются однотипные данные. Таким образом, каждый столбец таблицы является полем базы данных. Столбцам таблицы присваиваются уникальные имена полей базы данных, которые заносятся в первую строку. Эта строка называется строкой заголовков. Имя поля базы данных – это уникальный заголовок столбца таблицы. Каждое имя поля должно помещаться в отдельной ячейке. Все имена полей должны находиться в ячейках одной и той же строки над списком строк.

Перед практическим созданием собственной базы данных следует разработать её проект: детально продумать структуру, выписать её предполагаемые размеры, количество и название полей, учесть их последовательность, установить тип каждого поля, выбрать шрифты и способы выравнивания данных в ячейках, разработать название таблицы и т.д. Учитывайте, что для правильной работы фильтра полям необходимо давать уникальные имена, т.е. среди имён полей не должно быть двух одинаковых. В качестве имен полей нельзя использовать даты, формулы или пустые ячейки. Все результаты пред-

варительной подготовки должны быть записаны в тетради или в текстовом документе.

При создании структуры базы данных на рабочем листе Excel необходимо учитывать следующие правила:

1. На отдельном рабочем листе можно создать только одну базу данных. Не рекомендуется располагать на нём какую-либо другую информацию.

2. Не следует вставлять в базу данных пустые строки. Если пустая строка вставлена между строкой заголовков и данными таблицы, то Excel не определит имена полей. Пустые строки между записями воспринимаются табличным процессором как конец базы данных.

3. При вводе данных числового типа не следует писать наименование величин, например, руб., см, кг и др. Если наличие наименований необходимо, следует задавать текстовый тип данного поля, учитывая, что при этом математические функции с данными этого поля работать не будут.

4. Над таблицей базы данных следует располагать строку заголовков с уникальным именем для каждого поля. При этом все заголовки полей должны быть в одной строке таблицы. Если заголовок необходимо записывать в несколько строк, то все строки заголовка должны находиться в одной ячейке. При вводе таких заголовков для перехода на новую строку внутри ячейки используйте команду Alt +Enter.

5. Иногда бывает необходимо дать названия каждой записи (строке таблицы). В этом случае заголовки размещаются в первом столбце. Таких строгих требований как к заголовкам полей здесь нет, поскольку фильтрация по ним не производится.

6. Для визуального восприятия таблицы имена полей должны отличаться от других данных форматом символов, размером или начертанием (полужирным, курсивом и т.д.).

7. Ширина ячейки для каждого поля должна быть такой, чтобы внесённые в неё данные не выходили за границу ячейки.

8. Имена полей должны быть точными, короткими и понятными. Не используйте непонятные или не принятые сокращения слов.

Пример 1. В конце предыдущего параграфа при выполнении упражнения 9.1.1. вы подготовили материал для базы данных по автомобилям. Приведём алгоритм создания этой БД:

1. Сначала необходимо описать создаваемую базу данных:
 - Предположительные размеры БД – 500 записей;
 - Название БД – «Автомобильный салон»;
 - Поля в порядке их следования – «Номер», «Фирма-производитель», «Модель», «Двигатель», «Мощность», «КПП» (коробка переключения передач);
 - Форматы полей – первое поле числовое, остальные текстовые;
 - Шрифт для данных – Arial, размер 12, начертание обычное;
 - Шрифт для заголовков – Courier New, размер 14, начертание полужирное;
 - Выравнивание для данных – по левому краю и по центру по вертикали;
 - Выравнивание для заголовков – по центру ячейки по вертикали и по горизонтали.
2. Запустите программу Excel и сохраните пустой файл. Имя файла может не совпадать с именем базы данных.
3. В первой строке рабочего листа объедините 6 ячеек с A1 по F1. Введите в объединённую ячейку название таблицы и отформатируйте её по указанным в первом пункте параметрам.
4. Во второй строке в каждую ячейку с A1 по F1 введите соответствующие заголовки полей и отформатируйте их по указанным в первом пункте параметрам.
5. В соответствии с предполагаемым количеством записей в базе выделите 500 ячеек в столбце A с A3 по A502 и установите для них указанные в первом пункте параметры форматирования для первого поля. Для выделения такого большого диапазона используйте клавишу F5.
6. Выделите также с помощью клавиши F5 диапазон ячеек с B3 по F502 и установите для них указанные в первом пункте параметры форматирования для данных.
7. Заполните 10 записей созданной базы данных и сохраните файл.

Для работы с реляционной базой данных в табличном процессоре Excel удобно пользоваться следующими двумя командами программы JAWS:

- Insert +Alt +Ctrl +C – назначение диапазона заголовков столбцов;
- Insert +Alt +Ctrl +R – назначение диапазона заголовков строк.

Применять их можно по следующему алгоритму:

1. Выделить в строке все заголовки полей базы данных.
2. Ввести команду Insert +Alt +Ctrl +C.
3. Теперь при перемещении по полям базы данных в любой строке JAWS будет сообщать название поля, на которое переходит фокус.

Аналогичным образом следует поступать и для работы с заголовками строк (записей). Эти команды удобно использовать не только для чтения данных, но и при заполнении таблицы.

Контрольные вопросы

1. В какой программе рекомендуется создавать реляционные базы данных с возможностью комфортной работы без визуального контроля?
2. Что представляет собой реляционная база данных в Excel?
3. Что является записью базы данных в Excel?
4. Что является полем базы данных в Excel?
5. Сколько баз данных можно создать на одном листе? Как вы думаете почему?
6. Почему в базе данных не должно быть пустых строк?
7. Где на рабочем листе Excel должны располагаться заголовки полей базы данных?
8. Каким требованиям должны отвечать заголовки полей базы данных?
9. Почему в полях числового типа после данных нельзя писать наименования величин?
10. С чего следует начинать разработку базы данных?
11. Расскажите, что следует учесть при разработке проекта базы данных.
12. Как следует выделять в Excel большие диапазоны ячеек?

13. Как вы думаете для чего следует определять для каждого поля свой формат?

14. С помощью каких команд JAWS можно работать с заголовками полей и записей?

§9.3. Сортировка и поиск данных

Табличный процессор Excel наряду с вычислениями позволяет выполнять и другую автоматическую обработку данных – это сортировка и фильтрация. Смысл этих операций удобно рассмотреть на примере. Пусть, например, в некотором магазине имеется база данных товаров с указанием их цен (прайс-лист). Тогда сортировкой будет упорядочивание списка товаров в алфавитном порядке или в порядке возрастания или убывания их цен. Фильтром будет выбор всех товаров, цена которых не превосходит некоторого значения. А вычисление позволит рассчитать совокупную цену некоторого количества отобранных товаров.

Достаточно часто при работе с большими по объёму таблицами требуется отсортировать содержимое базы данных по определённому критерию (по алфавиту, по дате, по возрастанию или убыванию цены и т.д.), для этого можно воспользоваться специальной функцией Excel.

Пример 1. Выполнив упражнение 9.2.1. вы создали базу данных автомобилей. Приведём алгоритм сортировки данных на примере созданной таблицы:

1. Запустите табличный процессор Excel и загрузите в него книгу с созданной ранее базой данных.

2. Установите фокус в любую не пустую ячейку таблицы, например, в ячейку A3.

3. На ленте меню Excel во вкладке «Данные» найдите кнопку «сортировка...» и нажав на неё раскройте соответствующий диалог. Для этого следует перейти на ленту меню клавишей Alt, развернуть стрелкой вниз ленту «Данные», перемещаясь по ней стрелкой влево найти кнопку «сортировка...» и нажать на ней клавишу Пробел. Обратите внимание, что в различных версиях табличного процессора

Excel ленты меню могут отличаться некоторыми подробностями доступа к необходимой кнопке.

4. В раскрывшемся диалоге фокус окажется на кнопке «Ок». Перемещаясь по элементам управления клавишей Tab найдите флажок «мои данные содержат заголовки». Если этот флажок не установлен, установите его. Если этого не сделать, то заголовки столбцов таблицы будут участвовать в сортировке. Если же флажок установлен, то заголовки останутся на месте, а сортироваться будут только данные.

5. Перемещаясь далее по диалогу в комбинированном списке «Сортировать по» выберите заголовок столбца (имя поля), по которому необходимо отсортировать таблицу, например, заголовок «Модель». Напомним, что для раскрытия комбинированного списка следует ввести команду Alt +Enter.

6. Далее в этом же диалоге в комбинированном списке «Порядок» установите значение «По возрастанию». В этом случае записи базы данных будут отсортированы так, что названия моделей упорядочатся по алфавиту.

7. Перейдите далее на кнопку «Ок» и нажмите её для выполнения сортировки по настроенным параметрам.

В диалоге «Сортировка» есть кнопка «Добавить уровень», с помощью которой можно добавить в условия сортировки ещё один столбец. Это может быть необходимым, например, в случае наличия в базе данных большого количества одинаковых по имени моделей автомобилей, но отличающихся каким-либо другим параметром (мощность двигателя, вид КПП и др.). Второй уровень сортировки позволит упорядочить такие записи удобным для решения конкретной задачи способом, например, модели с одинаковым названием будут упорядочены по мощности двигателя.

В данном диалоговом окне есть также кнопка «Параметры...», при нажатии на которую, раскроется диалог с дополнительными возможностями настройки сортировки. В нём можно задать сортировку, например, с учётом регистра (большие и малые буквы) или указать, что сортировать следует не по строкам (установлено по умолчанию),

а по столбцам, а также выполнить некоторые другие установки режима сортировки.

Такая обработка данных, как фильтрация, предназначена для поиска записей, отвечающих определённым условиям. Например, для поиска в рассмотренной выше базе данных по автомобилям всех моделей с мощностью двигателя 150 л/с. Другими словами, поиск необходимых записей в базе данных Excel осуществляется наложением фильтров, т.е. наборов определённых критериев поиска. Обратите внимание, что для работы с фильтрами в Excel требуется наличие строки заголовков полей.

Пример 2. Предположим, что в базе данных автомобилей необходимо найти все модели японского производителя автомобилей Nissan Motor Company. Приведём алгоритм сортировки данных на примере созданной таблицы:

1. Запустите табличный процессор Excel и загрузите в него книгу с созданной ранее базой данных.

2. Установите фокус в любую ячейку созданной таблицы, например, в ячейку A2.

3. На ленте меню Excel во вкладке «Данные» найдите кнопку «Фильтр» и нажмите на ней клавишу Пробел. Обратите внимание, что при попадании фокуса на эту кнопку JAWS произнесёт: «фильтр кнопка не отмечено». Хотя программа невизуального доступа произносит «кнопка», данный элемент управления ведёт себя как флажок, его можно только отметить или снять отметку. После нажатия на «Фильтр» никаких дополнительных сообщений JAWS не выдаёт, отметка на элементе управления устанавливается и меню закрывается. Фильтр после этого будет включён.

4. Установите фокус на заголовок поля, по значению которого осуществляется фильтрация (поиск). В данном примере это будет заголовок «Фирма-производитель» в ячейке B2. После включения функции фильтра визуально в каждой ячейке строки заголовков появится значок в виде небольшой стрелки вниз, сообщающий о том, что в этой ячейке можно открыть меню фильтра. Когда фокус переходит в ячейку с заголовком поля при включённой функции фильтра, JAWS

сообщает: «Раскрывающаяся кнопка без применения фильтра. Выпадающее меню автофильтра».

5. Чтобы раскрыть меню автофильтра в ячейке заголовка поля (столбца) следует ввести команду Alt +стрелка вниз. Закрывается меню без выбора команды клавишей Escape.

6. Перемещаясь по меню стрелками вертикального управления курсором найдите пункт «Введите имя поля для поиска» и введите в этот редактор искомое значение поля – «Nissan Motor Company» (значение поля вводится без кавычек). В созданной вами базе данных такого значения может не быть. В этом случае введите любое из имеющихся значений. Обратите внимание, что при перемещении по заголовкам полей в ячейках с включённым автофильтром JAWS будет сообщать: «Раскрывающаяся кнопка применённого фильтра. Выпадающее меню автофильтра».

7. После применения автофильтра по значению данного поля, в таблице будут отображаться только те записи (строки), которые содержат искомое значение поля. Остальные строки будут скрыты.

8. Для отмены фильтрации по данному полю следует в меню автофильтра в заголовке поля выбрать команду «Снять фильтрацию». После этого скрытые строки отобразятся и таблица примет исходный вид.

Если применение фильтра к базе данных больше не требуется, то на ленте меню Excel во вкладке «Данные» следует снять отметку с кнопки «Фильтр». После этого меню автофильтра из заголовков полей в таблице исчезнут.

Изучив вкладку «данные» и меню автофильтра легко видеть, что Табличный процессор Excel предоставляет большое количество разнообразных возможностей по фильтрации и сортировке данных в таблице. Рекомендуем вам познакомиться с ними самостоятельно.

Контрольные вопросы

1. Какие виды обработки данных в Excel вы знаете?
2. Что выполняет функция сортировки в Excel?
3. Для чего на практике применяется сортировка данных?
4. Какие способы сортировки вы знаете?

5. Расскажите, в чём состоит алгоритм сортировки данных.
6. Участвуют ли в сортировке данных заголовки полей?
7. Как можно выполнять сортировку по нескольким условиям?
8. Что выполняет функция фильтрации в Excel?
9. Для чего на практике используется фильтр в Excel?
10. Как можно включить функцию фильтра?
11. Расскажите, в чём состоит алгоритм использования фильтра.

Глава 10

Подготовка демонстрационных визуально воспринимаемых объектов

§10.1. Диаграммы и графики

Достаточно часто в процессе учебы или работы возникает необходимость в построении разнообразных диаграмм и графиков. Современный стиль оформления цифровых документов предполагает использование этих наглядных форм демонстрации различных зависимостей - результатов исследований, отчетных показателей и т.д. Таким образом, владение способами построения диаграмм и графиков необходимо не только при работе в области точных наук (математика, физика и др.), но и в гуманитарных дисциплинах.

Табличный процессор Excel предоставляет необходимый функционал для построения диаграмм и графиков различных зависимостей и соотношений. Для построения графика функции, заданной формулой в Excel необходимо выполнить следующие шаги:

1. Задать область определения данной функции. Для этого в начальной ячейке диапазона, в котором будут находиться значения аргумента функции, следует ввести первое значение независимого аргумента, а далее заполнить остальные ячейки с помощью арифметической прогрессии. Обратите внимание, что чем меньше значение разности (шага) прогрессии, тем точнее будет построен график.

2. Задать область значений данной функции. Для этого в начальной ячейке диапазона, предназначенного для хранения области значений, вводится формула, задающая данную функцию, а на остальные ячейки области значений формула распространяется с помощью специальной вставки.

3. Построить график. Для этого следует выделить диапазон, содержащий её область определения и область значений, а затем воспользоваться группой «Диаграмма» на вкладке «Вставка» ленты меню. В диалоговом окне «Вставка диаграммы» необходимо на вкладке «Все диаграммы» выбрать из списка команду «График».

В качестве примера построения графика функции, заданной формулой, рассмотрим следующую задачу.

Пример 1. На рабочем листе Excel построить график функции, заданной формулой $y = x^3 + 2$ на отрезке $[-5; 5]$.

Для решения этой задачи можно воспользоваться приведённым ниже алгоритмом:

1. Сначала необходимо разработать структуру таблицы на рабочем листе, выбрать размеры, формат и расположение её элементов. Пусть, например, в первой строке в объединённых ячейках A1 и B1 будет находиться заголовок «График функции $y = x^3 + 2$ », выровненный по центру по вертикали и по горизонтали. Шрифт заголовка Arial размером 14 пунктов. Под заголовком в столбце A разместим область определения с шагом 0,1, а в столбце B – область значений данной функции не меняя формата ячеек, заданного по умолчанию. Расположение графика Excel определит автоматически.

2. Практическое выполнение работы начните с запуска Excel и сохранения пока пустой книги (файла с расширением «.xlsx») в необходимую папку на жестком диске компьютера.

3. Объедините ячейки A1 и B1, задайте центрирование текста в объединённой ячейке и установите необходимый шрифт и его размер. После чего введите заголовок в объединённую ячейку.

4. В столбце A расположите значения аргумента с шагом 0,1. Для этого в ячейку A2 введите начальное значение аргумента -5, а затем воспользуйтесь алгоритмом вычисления арифметической прогрессии с разностью 0,1 и конечным значением 5 (этот алгоритм приведён в параграфе 8.2. этой книги). таким образом, диапазон A2:A102 будет заполнен значениями аргумента X с шагом 0,1.

5. В ячейку B2 введите формулу $=A2^3 + 2$ расчёта значений данной функции. Напомним, что символ «^» (крышка) означает возведение в степень.

6. Скопируйте в буфер обмена содержимое ячейки B2.

7. Введите команду F5 для открытия диалогового окна «Переход». В этом диалоге укажите диапазон B2:B102 для размещения области значений функции. После нажатия клавиши Enter данный диапазон будет выделен.

8. Раскройте контекстное меню и в нём выберите команду «Специальная вставка...». В раскрывшемся диалоге в группе радиокнопок выберите вариант «Формулы» и нажмите на кнопку «ОК». Теперь в диапазоне B2:B102 расположены значения данной функции.

9. Выделите диапазон A2:B102 с помощью того же диалога «Переход» (клавиша F5). Выделенный диапазон содержит область определения и область значений данной функции.

10. На ленте меню раскройте вкладку «Вставка» и в группе «Диаграммы» выберите команду «Вставить диаграмму». Для выполнения этого пункта алгоритма можно сначала ввести команду Alt +C, а затем ввести последовательно букву «I» и цифру «1».

11. В раскрывшемся диалоге «Вставка диаграммы» перейдите на вкладку «Все диаграммы» и выберите из списка команду «График».

После выполнения всех шагов алгоритма на рабочем листе будет построена кубическая парабола, являющаяся графиком данной функции.

В табличном процессоре Excel можно строить не только графики функций, но и диаграммы. Рассмотрим в качестве примера круговую диаграмму. Круговая диаграмма предназначена для оценки долей элементов в общем количестве. Диаграмма именно такого вида встречается в задачах Единого Государственного Экзамена по информатике и ИКТ. Процедура построения диаграммы значительно проще, чем построения графика функции. Рассмотрим алгоритм создания диаграммы на практическом примере.

Пример 2. На рабочем листе Excel постройте круговую диаграмму, наглядно демонстрирующую количественные соотношения отличников, хорошистов и троечников некоторого класса. В классе 4 отличника, 8 хорошистов и 4 троечника. Двоечников в классе нет!

Построить такую диаграмму можно используя нижеследующий алгоритм:

1. Разработайте структуру таблицы на рабочем листе. Пусть, например, в объединённых ячейках A1 и B1 находится отцентрированный (по вертикали и по горизонтали) заголовок «Диаграмма успеваемости класса», в ячейках столбца A ниже располагаются за-

головки строк с данными «Отличники», «Хорошисты», «Троечники», а в столбце В соответствующие числовые значения. Заголовки строк выравниваются по левому краю по горизонтали, по центру по вертикали и отображаются шрифтом Arial размера 14 пунктов полужирным начертанием. Числовые данные в ячейках В2, В3 и В4 центрируются по вертикали и горизонтали, отображаются шрифтом Arial обычного начертания размера 12 пунктов.

2. Запустите программу Excel, сохраните пустой файл (книгу) на диск дав ему имя «диаграмма успеваемости» и выполните необходимые операции по созданию описанной в предыдущем пункте алгоритма структуры таблицы. Введите все необходимые заголовки.

3. Введите числовые данные в ячейки В2, В3 и В4.

4. Поместите фокус на одну из ячеек с данными В2, В3 или В4. Обратите внимание, что если данные расположены в непрерывном диапазоне ячеек, то выделять весь диапазон не надо, достаточно поместить фокус в одну из его ячеек.

5. На вкладке «Вставка» в группе «Диаграмма» выберите в списке «Круговая» и нажмите Enter. Обратите внимание, что названия групп, диалогов и других объектов могут отличаться в различных версиях программы Excel, однако, общий принцип остаётся неизменным.

После выполнения приведённого выше алгоритма на рабочем листе появится круг, разделенный на три сектора, пропорциональные по площади соответствующим величинам.

Для контроля получившегося результата с помощью программы невизуального доступа к информации удобно пользоваться следующими командами программы JAWS for Windows:

- выбрать объект на листе – Ctrl + Shift + O;
- Выбрать активную диаграмму на текущем листе – Ins + Alt + C;
- Вывести выбранную активную диаграмму в окне виртуального просмотра – Ctrl + Ins + C.

Команда Ctrl + Ins + C выведет в отдельное текстовое окно информацию о созданной диаграмме. По этому окну можно перемещаться стрелками управления курсором и подробно изучить описание диаграммы. Прodelайте это самостоятельно.

В конце параграфа Рассмотрим пример решения теоретической задачи.

Пример 3. Дан фрагмент электронной таблицы:

	A	B	C
1		3	4
2	$=(A1 + B1 + 2)/(C1 - B1)$	$=(2 * C1 - 2)/A1$	$=B1 * C1/(B1 - A1)$

Какое целое число должно быть записано в ячейке A1, чтобы построенная после выполнения вычислений диаграмма по значениям диапазона ячеек A2:C2 показывала равенство этих значений?

Решение. Из условия задачи следует, что значения всех трёх ячеек диапазона A2:C2 равны. Обозначим искомое значение ячейки A1 через x и приравняем формулы ячеек B2 и C2, при этом подставляя вместо адресов ячеек B1 и C1 их значения. Получим уравнение:

$$(x + 3 + 2)/(4 - 3) = (2 * 4 - 2)/x;$$

После упрощения получим:

$$x - 5 = 6/x;$$

Из формулы, записанной в ячейке B2 следует, что в ячейке A1 не может находиться число 0. Поэтому имеем право умножить обе части уравнения на x , получив результате квадратное уравнение:

$$x^2 + 5x - 6 = 0;$$

Решив его получим два корня:

$$x_1 = -6, x_2 = 1;$$

Подставим каждый корень вместо A1 в формулы, помещённые в ячейки B2 и C2 и сравним их значения. Для $A1 = x_1 = -6$, имеем:

$6/(-6) = 12/(3 - (-6))$, т.е. $-1 = 4/3$, что не верно. Следовательно значение $A1 = -6$ не является решением задачи.

Для $A1 = x_2 = 1$, имеем:

$6/1 = 12/(3 - 1)$, $6 = 6$. Следовательно, значение $A1 = 1$ является решением задачи.

Ответ: 1.

Контрольные вопросы

1. Что такое график функции?
2. Что называется областью определения функции?
3. Что называется областью значений функции?

4. Для чего служат графики функций?
5. Опишите алгоритм построения графика функции.
6. Что такое диаграмма?
7. Какие виды диаграмм вы знаете?
8. Для чего используются диаграммы?
9. Опишите алгоритм построения круговой диаграммы.
10. Какие команды JAWS для изучения диаграммы вы знаете?

§10.2. Математические формулы

Аналогично случаю с графиками и диаграммами также часто в процессе учебы или работы возникает необходимость в записи математических формул различной степени сложности. В некоторых случаях текстовые документы предполагают использование формул и специальных знаков как содержательной части статьи, курсовой или контрольной работы и т.д. В простых ситуациях элементарные формулы можно ввести со стандартной клавиатуры или с помощью брайлевского дисплея, например, $2x + 3y = 22$.

Однако, часто бывают нужны более сложные формулы, ввести которые с помощью клавиатуры невозможно, для их ввода служит специальный функционал текстового процессора Word. Владение способами использования этого функционала для вставки формул, математических знаков, степеней, индексов, интегралов, логарифмов и т.п. необходимо не только при работе в области точных наук (математика, физика, химия и др.), но в некоторых случаях и в гуманитарных дисциплинах.

В текстовом процессоре MS Word для работы с подобными объектами используется встроенный редактор формул. Школьникам, студентам, преподавателям, ученым, а также профессионалам в различных других областях деятельности человека довольно часто приходится сталкиваться с необходимостью использования в учёбе или работе различных формул. Но если самые простые из них не требуют использования специальных символов и сложного форматирования, то встречаются случаи, когда без специальных средств Word грамотно и красиво записать в текстовом документе формулу невозможно. Следует подчеркнуть, что редактор формул только записывает фор-

мулы (создаёт их изображения), никаких вычислений он не производит. Все вычисления необходимо производить другими средствами. Ниже приведены описания основных шагов и примеры использования редактора формул.

В общем случае при работе с редактором формул необходимо выполнить следующие шаги:

1. Установите курсор в ту точку текстового документа, в которую необходимо поместить формулу или специальный символ.

2. Введите клавиатурную команду Alt += (равно) для включения редактора формул. После ввода этой команды на ленте справа от вкладки «Справка» появится вкладка «Уравнение». Обратите внимание, что JAWS сообщит только названия нажатых клавиш «Alt +=» и никаких других сообщений не выдаст.

3. Активируйте на ленте появившуюся вкладку «Уравнение» и выберите на ней необходимый объект (готовую формулу или специальный знак). После нажатия клавиши Enter выбранный объект появится в текстовом документе.

4. Введите с клавиатуры необходимую часть формулы, например, подкоренное выражение или основание степени.

5. Выключите редактор формул той же клавиатурной командой Alt +=, после чего вкладка «Уравнение» исчезнет с ленты.

После выполнения описанных выше действий в текстовом документе появится грамотно оформленная формула или математический символ, причём эта формула или символ будут адекватно прочитываться программой незрительного доступа к информации.

Обратите внимание, что команду включения редактора формул или ввод числовых или буквенных значений можно выполнять с помощью клавиатуры Перкинса брайлевского дисплея.

При помещении курсора на формулу содержимое контекстного меню меняется, в нём появляются команды «Линейный» и «Профессиональный». Эти команды служат для переключения режимов редактирования формулы и отображения её в не редактируемом естественном математическом формате. Таким образом, при возникновении необходимости отредактировать уже введённую ранее формулу, следует поместить курсор на редактируемую формулу и в кон-

текстном меню выбрать команду «Линейный», а для преобразования формулы в естественный математический вид без возможности редактирования следует выбрать команду «Профессиональный». Заметим, что описанные пункты в контекстном меню появляются только при включённом режиме редактирования формул.

Рассмотрим применение редактора формул более подробно на конкретном примере.

Пример 1. Решить квадратное уравнение $x^2 - 5x + 4 = 0$ и записать его решение в текстовом процессоре Word.

Приведём подробный алгоритм записи решения квадратного уравнения с помощью редактора формул Word:

1. Запустите текстовый процессор Word и сохраните пока пустой документ под именем «квадратное уравнение».

2. Установите курсор в то место документа, в котором надо записать уравнение (возможно, вы хотите включить решение уравнения в какой-либо документ, например, в решение контрольной работы).

3. Включите редактор формул введя команду Alt += (равно). Обратите внимание, что JAWS произнесет только имена нажатых клавиш «Alt + равно», о включении редактора формул сообщения не будет.

4. Активируйте на ленте вкладку «Уравнение». JAWS при этом произнесет: «Работа с уравнениями». Обратите внимание, что эта вкладка расположена правее вкладки «Справка», поэтому для её поиска быстрее перемещаться по ленте стрелкой влево.

5. На вкладке «Уравнение» раскройте группу «Индекс» и в ней нажмите кнопку «Квадрат X» (это первая кнопка в группе). JAWS произнесёт: «Структура уравнения квадрат X». После нажатия на эту кнопку в тексте в позиции курсора появится x^2 .

6. Допишите квадратное уравнение с помощью клавиатуры, т.е. наберите $- 5x + 4 = 0$. Заметим, что дописать формулу можно и с помощью клавиатуры Перкинса брайлевского дисплея.

7. Напишите поясняющую фразу «Вычисление дискриминанта:». Затем с новой строки начните писать формулу вычисления дискриминанта $D =$ (после знака равенства ничего писать не следует).

8. Снова активируйте вкладку «Уравнение» и в группе «Индекс» нажмите ту же кнопку «Квадрат X». Как и в пункте 4 этого алгоритма на экране появится x^2 .

9. Введённая только что формула $x^2\Phi$ будет находиться в режиме редактирования (режим «Линейный»). Установите курсор на букву x и удалите её нажав клавишу Del. Затем введите с клавиатуры обычными символами (-5) . На экране появится $(-5)^2$.

10. Далее допишите оставшуюся часть выражения для вычисления дискриминанта -4×4 . Здесь знак «минус» и цифры вводятся с клавиатуры, а для ввода знака умножения следует воспользоваться группой «Символы для уравнений» той же вкладки «Уравнение». В этой группе достаточно много математических знаков, среди которых перебором следует найти знак «символ умножения». Обратите внимание, что символов для вставки много и расположены они в несколько столбцов. Поэтому отыскать перебором необходимый символ достаточно сложно. В данном случае отыскания символа умножения удобно в группе двигаться стрелкой вправо.

11. В следующей строке после вычисления дискриминанта введите поясняющую фразу «Вычисление корней:».

12. На следующей строке введите выражение для вычисления корней данного квадратного уравнения

$$x = \frac{(5 \pm \sqrt{9})}{2}$$

Для этого удобно воспользоваться готовой формулой в группе «Уравнения» на всё той же вкладке «Уравнение». В этой группе двигаясь стрелкой вниз следует выбрать кнопку «Квадратное уравнение».

13. Теперь запишите результаты вычисления корней

$$x_1 = 1, x_2 = 4.$$

Заметим, что для записи нижних индексов можно воспользоваться клавиатурной командой Ctrl += (равно). Для этого следует ввести букву x , после неё записать нужную цифру (1 или 2), выделить эту цифру и ввести команду Ctrl += для преобразования её в нижний индекс.

14. Выключите редактор формул командой Alt +=.

Обратите внимание, что в процессе ввода обычного текста редактор формул может выключиться. Включён он или нет можно проверить по наличию на ленте вкладки «Уравнение».

Для ввода некоторых специальных символов может быть полезна команда Ins +4 программы невизуального доступа к информации JAWS. Эта команда раскрывает диалог «Выбор символа для вставки», с помощью которого можно вставить в текст документа Word некоторые обыкновенные дроби, специальные знаки, особо написанные буквы и др.

Контрольные вопросы

1. В каких ситуациях в документе Word могут понадобиться формулы? Приведите примеры.
2. Приведите примеры формул, которые можно ввести с помощью стандартной клавиатуры или брайлевского дисплея.
3. Приведите примеры формул, которые нельзя ввести с помощью стандартной клавиатуры или брайлевского дисплея.
4. Для чего служит редактор формул Word?
5. Какие основные шаги следует выполнить при работе с редактором формул Word?
6. Какая команда включает и выключает редактор формул Word?
7. Какие изменения произойдут на ленте меню после включения редактора формул Word?
8. Как можно редактировать введенную формулу?
9. Для чего служат команды контекстного меню «Линейный» и «Профессиональный»?
10. Какие способы создания индексов вы знаете?
11. Как можно узнать включён или нет редактор формул?
12. Как с помощью программы JAWS можно вставить в документ Word:
 - А) Знак градуса;
 - Б) Знак умножения;
 - В) Дробь три четвертых?

§10.3. Презентации

Презентация представляет собой современную форму коммуникации с аудиторией посредством визуального восприятия информации излагаемой докладчиком. Презентации стали неотъемлемой частью защиты курсовой или дипломной работы студента, выступления на

семинарах и конференциях, чтения лекций и т.д. Каждый студент в процессе обучения неоднократно выступает перед аудиторией сопровождая свою речь презентацией, воспроизводимой на экране компьютера или проектора.

Способность человека обрабатывать конкретные визуальные образы гораздо выше мыслительной переработки абстрактной информации. Зрительное восприятие по объёму воспринимаемой информации превосходит слуховое. Поэтому презентация стала неотъемлемой частью любого публичного выступления и определёнными навыками по её подготовки должен обладать каждый.

Презентация состоит из слайдов – отдельных кадров, последовательно выводимых на экран докладчиком. Слайды в сжатой лаконичной форме выражают конкретную мысль и иллюстрируют её различными графическими объектами (фотографиями, чертежами, диаграммами, графиками и т.п.). Как правило каждый слайд имеет отдельный заголовок. По очевидным причинам подготовить визуально воспринимаемую презентацию без опоры на зрение достаточно сложно. Однако, с использованием небольшой помощи зрячего ассистента обеспечить своё выступление качественной презентацией вполне возможно.

Следует учесть, что достаточно часто докладчики используют презентации, содержащие только текстовую информацию. Отсутствие графических объектов на слайде может быть оправдано его содержанием и не делает слайд хуже. Фотографии и другой иллюстративный материал помогает раскрыть содержание, но не заменяет его.

Текстовые слайды служат для быстрого восприятия аудиторией информационных блоков. Слушатели просматривают текст на слайде, чтобы убедиться, что они не упустили ничего важного в речи выступающего, а также для лучшего усваивания информации.

Текст слайда должен быть кратким и точным. В предложении не нужны лишние слова, а в абзаце-предложения по той же причине, по которой не нужны лишние линии на картине и лишние детали в механизме. В недостаточно проработанном тексте есть слова, которые можно удалить из него без потери смысла. Текст должен быть

ясным, информативным, простым для восприятия и не содержать ничего лишнего.

Начать подготовку презентации следует с создания её плана. При разработке плана необходимо ответить на следующие вопросы:

- Какова цель вашей презентации;
- Каков будет общий заголовок презентации;
- Какой заголовок будет иметь каждый слайд;
- Есть ли необходимость помещать на слайд графический иллюстративный материал (если да, то какой именно)?

После подготовки структуры, текста и иллюстративного материала (если необходимо) можно приступать к созданию самой презентации. Специальных программ для создания презентаций достаточно много. Наиболее популярными среди них являются:

- Microsoft PowerPoint;
- Google Презентации (online инструмент);
- Keynote для Mac OS.

В настоящее время очень часто презентации используются для сопровождения докладов и других сообщений на дистанционных (online) мероприятиях, использующих специально для этих целей разработанные Интернет-площадки. При этом стандартным форматом для презентаций становится формат PDF. Как следствие все чаще презентации подготавливают не в специальной программе, а в привычном текстовом процессоре MS Word. Word позволяет сохранять файлы в формате PDF путём выбора соответствующего значения в комбинированном списке «Тип Файла» диалогового окна «Сохранить как». Напомним, что вызывается этот диалог командой F12.

В текстовом процессоре Word презентация подготавливается также, как текстовый документ. Необходимо только каждый слайд размещать на отдельной странице. Для завершения страницы и перехода на новую используйте команду Ctrl +Enter. Все необходимые приёмы форматирования абзацев и символов вам хорошо известны.

При подготовке презентации в текстовом процессоре Word придерживайтесь следующих правил:

- Ограничивайте объем текста одного слайда пятью – семью такими структурными элементами, как предложение, элемент списка и т.п.

- Если для выражения мысли необходимо больше текста, разбивайте его на несколько слайдов;
- Не используйте выравнивание по центру для больших блоков текста (визуально читать такой текст трудно);
- Перечисления оформляйте в виде списков;
- Не пользуйтесь эффектами анимации;
- Используйте черный текст на белом фоне;
- Количественные данные полезно визуализировать с помощью диаграмм, вставляемых как графические объекты (Об их подготовке было рассказано в параграфе 10.1. этой главы).

При подготовке презентации старайтесь соблюдать однотипность слайдов, оформляйте их в одном и том же стиле. Если ваша презентация содержит какие-либо графические объекты (фотографии, диаграммы, графики и т.п.) размещайте их на слайдах в одном и том же месте (слева вверху, справа внизу и т.п.). Презентация, на которой однотипные элементы занимают одно и то же место на каждом слайде, выглядит аккуратно.

В настоящее время при выступлении на мероприятии любого уровня (семинар, конференция, симпозиум и т.д.) принято обеспечивать доступность презентации и всего выступления в целом для слушателей с нарушением зрения. Для обеспечения такой доступности следует придерживаться приведённых ниже рекомендаций по оформлению презентации:

- Фон слайда однотонный, контрастный по отношению к цвету текста (например, чёрный текст на белом фоне);
- Текст оформляется шрифтом Arial обычного начертания достаточно большого размера (например, 18 – 24 пункта);
- Заголовки оформляются тем же шрифтом Arial, но полужирного начертания и большего размера (например, 24 – 32 пункта);
- В тексте слайда допускается выделение контрастным цветом слов, несущих особое смысловое значение;
- Необходимо избегать курсивного и подчеркнутого начертания;
- Рисунки и графики, используемые в презентации, должны быть максимально упрощены, с контрастом по цвету, без мелких элементов и деталей;

- На слайде рекомендуется использовать только один график, рисунок, или диаграмму для лучшего визуального восприятия;
- Если графическое изображение сопровождается текстом, то текст размещается отдельно (слева, выше, ниже изображения), а не на фоне изображения;
- Любое графическое изображение (рисунок, график, диаграмма, фотография, иллюстрация и т.п.) при выступлении обязательно должны сопровождаться коротким устным описанием;
- Полезно давать устные пояснения по изображению на слайде для облегчения ориентировки (для лиц с узким полем зрения) например: « На слайде вы видите диаграмму, состоящую из четырех блоков. Блок в нижнем правом углу....». «На слайде показаны результаты исследования в виде гистограммы. Столбики в левой стороне диаграммы показывают данные....., столбики на правой стороне.....»;
- Анимационные эффекты в презентациях не допускаются;
- Скорость смены слайдов должна обеспечить восприятие информации людьми с нарушенным зрением.

Подготовленная по вышеописанным правилам презентация может сделать ваше выступление более понятным, чётким и эффективным.

В конце параграфа приведём список некоторых полезных для практического использования символов, для ввода которых применяются клавиатурные комбинации (непосредственно на клавиатуре этих символов нет):

- Разрыв строки – Shift +Enter;
- Разрыв страницы – Ctrl +Enter;
- Разрыв столбца – Ctrl +Shift +Enter;
- Длинное тире – Alt +Ctrl +- (минус на цифровой клавиатуре);
- Короткое тире – Ctrl +- (минус на цифровой клавиатуре);
- Мягкий перенос – Ctrl +- (минус);
- Неразрывный дефис – Ctrl +Shift +- (минус);
- Неразрывный пробел – Ctrl +Shift +Пробел;
- Символ авторского права – Alt +Ctrl +C;
- Символ охраняемого товарного знака – Alt +Ctrl +R;
- Символ товарного знака – Alt +Ctrl +T;

- Многоточие – Alt +Ctrl +. (точка).

Контрольные вопросы

1. Для чего нужна презентация?
2. Где используются презентации?
3. Чем объясняется эффективность презентации?
4. Что такое слайд?
5. Какие информационные объекты могут располагаться на слайде?
6. Каким требованиям должен отвечать текст слайда?
7. С чего начинают подготовку презентации?
8. Что необходимо сформулировать при разработке презентации?
9. Какие специальные программы для подготовки презентации вы знаете?
10. В каком формате следует сохранять современную презентацию?
11. Можно ли самостоятельно подготовить презентацию без визуального контроля?
12. Какую программу вы будете использовать для подготовки презентации? Почему?
13. Каких рекомендаций надо придерживаться при подготовке слайдов?
14. Расскажите, как сделать презентацию доступной для слушателей с нарушением зрения.
15. Какие вы знаете символы, для ввода которых используются клавиатурные комбинации?

Глава 11

Разработка WEB-страниц без визуального контроля

§11.1. Основные теги HTML

Навигация по WEB-странице с помощью программы невидимого доступа к информации опирается на структурные элементы, соответствующие разметке данной страницы. Разметку WEB-страницы осуществляет её автор во время создания сайта. В этой главе изложены основные команды особого языка разметки для форматирования WEB-страниц. Эта информация полезна не только для создания собственных сайтов, но и для грамотной и быстрой навигации по сети Интернет.

Незрячим пользователям при отсутствии возможности охватить взглядом сразу всю страницу приходится формировать в сознании её образ, и, опираясь на него как на зрительный, осуществлять навигацию по странице. Для этого весьма полезно иметь представление о принципах формирования WEB-страниц. Таким образом целесообразно изучать приемы навигации параллельно с основами HTML-программирования. Незрячий пользователь должен иметь представление об основных элементах HTML, так как именно они являются ориентирами при навигации с помощью программ невидимого доступа к информации JAWS for Windows и NVDA.

Отрабатывать приёмы навигации без визуального контроля достаточно удобно и эффективно на самостоятельно созданных сайтах. Тот факт, что WEB-страница находится не на удалённом сервере сети Интернет, а на вашем Компьютере никак не повлияет на приемы навигации по ней. Такой способ обучения приводит к формированию прочных навыков, основанных на понимании структуры страницы, а не на механическом запоминании определенного набора команд.

WEB-страницы или HTML-страницы написаны на особом языке разметки, который называется HTML (Hyper Text Markup Language). HTML состоит из элементов, структурирующих информацию с помощью абзацев, фреймов, таблиц, списков и т. д. Эти элементы назы-

ваются Тегами. Основное отличие Web-страницы от обычного текста заключается в том, что она имеет возможность перехода (ссылку) на другие страницы.

Теги HTML часто имеют атрибуты, управляющие работой тега. С помощью атрибутов разработчики Web-страниц предоставляют информацию о данных на странице, такую, например, как описания рисунков или названия элементов управления. Атрибуты могут изменять внешний вид текста или рисунков. Они также могут изменять поведение элементов управления, например, заставлять ссылку открываться в новом окне.

Большинство тегов парные, т.е. состоят из открывающего и закрывающего. Между этими открывающим и закрывающим тегами можно использовать другие теги, придавая HTML иерархическую структуру. Другими словами один тег может находиться внутри другого.

Имена всех тегов HTML записываются в угловых скобках, причем перед именем завершающего тега после угловой открывающей скобки записывается знак «/» (косая черта).

Обратите внимание, что вводятся угловые скобки с помощью стандартной клавиатуры при нажатой клавише Shift в латинской раскладке. Находятся они на клавишах с буквами «б» и «ю». Причём JAWS называет угловую скобку открыто «меньше», а угловую скобку закрыто «больше», поскольку знаки неравенства и угловые скобки обозначаются одними и теми же символами.

С помощью брайлевского дисплея эти знаки вводятся следующим образом:

- < (меньше) – точки 56;
- > (больше) - точки 45;
- / (косая черта) – точки 34.

При изучении тегов могут быть полезны две следующие команды программы JAWS:

- Ins +Shift +F1 - получение информации о теге;
- Ctrl +Ins +Shift +F1 - получение дополнительной информации о теге.

При вводе данных команд открывается текстовое окно, отображающее теги и их параметры (атрибуты). Это окно можно изучать пе-

решающаяся по нему стрелками управления курсором. Для закрытия окна с информацией о тегах используйте клавишу Esc.

Перечислим некоторые основные теги языка HTML:

Тег <HTML> Сообщает браузеру (программе просмотра Интернет-страниц), что документ является Web-страницей. Этим тегом документ начинается и заканчивается, а все другие теги находятся внутри него, т.е. на более низком иерархическом уровне.

Пример:

<HTML>

Содержимое WEB-страницы

</HTML>

Тег <TITLE> не является частью отображаемого текста. Он будет отображаться как заголовок страницы или название окна. В HTML-документе может быть Только один тег <TITLE>. Этот тег должен использоваться для идентификации содержимого документа. Он помещается перед открывающим тегом <BODY> и не входит в его иерархию. При вводе команды Ins +T, помимо названия браузера JAWS сообщает именно строку, заключенную в тег <TITLE>.

Пример:

<TITLE>

Текст заголовка окна

</TITLE>

Тег <BODY> определяет видимую часть документа. В этом разделе располагается вся содержательная часть информации (текст статьи, фотографии, формы для заполнения, другие объекты). Этот тег находится внутри тега <HTML> и является вторым в иерархии. Тег имеет ряд необязательных атрибутов:

bgcolor – Устанавливает цвет фона документа;

text – Устанавливает цвет текста документа;

link – Устанавливает цвет ссылок;

vlink – Устанавливает цвет посещенных ссылок.

Пример:

<BODY bgcolor =»yellow» text =»blue»>

Текст страницы голубым цветом на жёлтом фоне.

</BODY>

Тег <P> создает новый абзац (параграф). Два или более тега <P>, идущих подряд, заменяются одним. Атрибуты:

align – выравнивает абзац относительно одной из сторон документа, значения:

left – выравнивание по левому краю (по умолчанию);

right – выравнивание по правому краю;

center – выравнивание по центру;

justify – выравнивание по ширине.

Пример:

<P align =«justify»>

Текст абзаца выровненный по ширине.

</P>

Тег <H> служит для создания заголовка. Всего существует 6 уровней заголовков – от H1 до H6. Заголовки уровня 1 являются самыми большими, и на каждом следующем уровне шрифт, которым набран текст заголовка, становится меньше. Атрибуты:

align – выравнивает заголовок в соответствии со следующими значениями:

center – по центру;

left – по левому краю;

right – по правому краю.

Пример:

<H3 align = «center»>

Центрированный заголовок третьего уровня.

</H3>

Тег <BIG> выводит текст крупным шрифтом.

Пример:

<BIG>

Текст крупным шрифтом.

</BIG>

Тег <SMALL> выводит текст мелким шрифтом.

Пример:

<SMALL>

Текст мелким шрифтом.

</SMALL>

Тег
 не требует парного закрывающего тега. Тег
 (break line) вставляет перевод строки.

Пример:

 На месте этого тега в окне браузера будет переход на новую строку.

Тег <NOBR> запрещает перевод строки. Бывают случаи, в которых возникает необходимость запретить перевод строки. Текст, заключенный внутри этого тега будет гарантированно располагаться в одной строке без переноса на другую. Длинная строка может не уместиться на экране, и тогда для ее визуального просмотра придется использовать горизонтальную прокрутку.

Пример:

<NOBR>

Этот текст в любой ситуации в окне браузера будет расположен в одну строку без переноса на другую.

</NOBR>

Тег <HR> добавляет в HTML-документ горизонтальную линию. Перед и после линии помещается пустая строка. Закрывающий тег не требуется. Атрибуты:

size – устанавливает высоту (толщину) линии;

width – устанавливает ширину линии в пикселах или процентах;

noshade – создает линию без тени;

color – задает цвет линии.

Пример:

<HR> В окне браузера на этом месте будет горизонтальная линия.

Тег задаёт шрифт текста, его цвет и размер символов.

Атрибуты:

color – устанавливает цвет текста;

face – определяет гарнитуру шрифта;

size – устанавливает Размер символов в пределах от 1 до 7.

(По умолчанию равен 3).

Пример:

Текст будет выведен мелким шрифтом.

Тег задает жирное начертание.

Пример:

Текст будет выведен жирным шрифтом.

Тег <I> задает курсивное начертание.

Пример:

<I>

Текст будет выведен курсивом.

</I>

Тег <S> задает зачеркивание.

Пример:

<S>

Текст будет выведен зачёркнутым.

</S>

Тег <U> задает подчеркивание.

Пример:

<U>

Текст будет выведен подчёркнутым.

</U>

Тег <TABLE> создает таблицу. Все прочие элементы таблицы должны быть вложенными в него. Допускается также вложение таблиц одна в другую, т.е. содержимым ячейки может быть другая таблица. Закрывающий тег обязателен. Атрибуты:

width – задает ширину таблицы или ячейки в пикселях или процентах;

align – Выравнивание таблицы относительно документа. Возможные значения: center, left, right;

bgcolor – Определяет цвет заднего фона таблицы;

border – Толщина рамки в пикселях. Если атрибут не указан, то таблица выводится без видимой рамки;

bordercolor – задает Цвет рамки;

cellspacing – Задаёт расстояние между ячейками таблицы;

cellpadding – Задаёт расстояние между содержимым ячейки и ее рамкой;

rules – Описывает рамки вокруг таблицы. Может принимать следующие значения:

all – Отображает все части рамки внутри таблицы;

co – Отображает все вертикальные рамки внутри таблицы;

groups – Отображает горизонтальные части рамки между группами таблицы;

none – Удаляет все рамки вокруг таблицы;

rows – Отображает все горизонтальные рамки внутри таблицы.

Summary – задает Описание таблицы для удобства пользователей программ невидимого доступа к информации.

Пример:

<TABLE>

Здесь располагаются теги для создания строк, столбцов и ячеек таблицы.

</TABLE>

Тег <TH> задаёт заголовок для столбца в таблице. Обычно, текст выделяется жирным шрифтом. Атрибуты:

bgcolor – Цвет фона;

bordercolor – Цвет рамки для элемента;

height – определяет высоту элемента в процентах или пикселях;

align – Выравнивает текст в ячейке по горизонтали. Возможные значения: left (по умолчанию), right, center;

valign – Выравнивает текст в ячейке по вертикали. Возможные значения: top – по верхнему краю; middle – по центру; bottom – по нижнему краю.

colspan – определяет количество столбцов, которое объединено в одной ячейке (по умолчанию =1);

rowspan – определяет количество строк, которое объединено в одной ячейке (по умолчанию=1).

Пример:

<TH>

Заголовок столбца

</TH>

Тег <TD> определяет отдельную ячейку в таблице. Закрывающий тег </TD> необязателен. Атрибуты:

height – определяет высоту ячейки в процентах или пикселях;
align – Выравнивает текст в ячейке (по умолчанию по левому краю);
valign – Выравнивает текст в ячейке по вертикали;
colspan – определяет количество столбцов, которое объединено в одной ячейке (по умолчанию =1);
rowspan – определяет количество строк, которое объединено в одной ячейке (по умолчанию =1).

Пример:

<TD>

Содержимое ячейки.

</TD>

Тег <TR> задаёт строку в таблице. Закрывающий тег </TR> обязателен. Атрибуты:

align – Выравнивает текст в ячейке: left, right, center;

valign – Выравнивает текст в ячейке по вертикали: top, middle, bottom.

Пример:

<TR>

Теги, определяющие строку.

</TR>

Тег задаёт маркированный список. атрибут type определяет вид маркера, который браузер помещает перед каждым элементом списка. Этот атрибут может отсутствовать или принимать одно из трёх значений:

circle – окружность;

disc – диск;

square – Квадрат.

Пример:

<UL type =circle>

Теги, определяющие элементы маркированного списка.

Тег задаёт Нумерованный список. В нумерованных списках каждый элемент снабжён номером, вид и начальное значение которого задаётся атрибутом type, который может принимать значения:

1 – Нумерация выполняется арабскими цифрами 1, 2, 3 и т.д. (задан по умолчанию);

a – Нумерация выполняется малыми латинскими буквами a, b, c и т.д.;

A – Нумерация выполняется большими латинскими буквами A, B, C и т.д.;

i – Нумерация выполняется малыми римскими цифрами i, ii, iii и т.д.;

I – Нумерация выполняется большими римскими цифрами I, II, III и т.д.;

Пример:

<OL type =»3»>

Теги, определяющие элементы нумерованного списка, начинающегося с цифры 3.

Тег (list item) задает каждый из пунктов списка. Конечный тег не является обязательным.

Пример:

Текст элемента списка.

Тег <A> служит для создания ссылок (гипертекста). Этот тег позволяет осуществлять переход от одного WEB-документа к другому или к другому фрагменту того же документа. Тег должен содержать либо атрибут name, либо href. Атрибуты:

name – задает имя элемента для ссылки в другую точку того же документа;

href – задает ссылку на другую страницу, на которую должен перейти пользователь, нажав на ссылке клавишу Enter.

Текст ссылки заключается между открывающим <A> и закрывающим тегом .

Пример:

Библиотека аудио книг

Работая в сети Интернет вы встречались на WEB-страницах с такими объектами, как Кнопка, Флажок, Поле редактирования и др. Такие элементы WEB-документа также задаются соответствующими тегами. Создание подобных объектов подчиняется описанным выше правилам использования тегов и при желании вы сможете освоить работу с ними самостоятельно, используя информацию из сети Интернет.

Контрольные вопросы

1. Для чего служит язык разметки HTML?
2. На чём основана навигация по WEB-странице без визуального контроля?
3. Можно ли пользоваться для освоения приёмов навигации WEB-страницей, если она находится на вашем компьютере?
4. Чем отличается WEB-страница от обычного текстового документа?
5. Что такое тег HTML?
6. Почему большинство тегов HTML парные?
7. Чем отличается открывающий тег от закрывающего?
8. Как записываются теги HTML?
9. Для чего служат атрибуты тега?
10. Какой тег содержит в себе все остальные теги WEB-страницы?
11. Какие теги HTML вы знаете?
12. Какие команды JAWS служат для получения информации о тегах в браузере?

§11.2. Создание простейших WEB-страниц

Как вы уже знаете WEB-страница (или HTML-страница) представляет собой обыкновенный текстовый файл с расширением htm или html , содержащий теги для управления отображением текста в браузере. Браузер (или интернет-обозреватель) анализирует имеющиеся в таком текстовом файле теги и выполняет предписываемые ими команды. При этом WEB-страница может быть получена из сети Интернет или находится на вашем компьютере.

Если выбрать в каком-либо файловом менеджере файл с расширением htm или html и нажать на нём Enter, операционная система по расширению файла определит, что открыть этот файл следует браузером, который сможет выполнить содержащиеся в нём команды. Ниже приведены примеры создания таких текстовых файлов.

Пример 1. Создайте простейшую WEB-страницу с именем «Моя первая страница!» и одной строкой текста: «Здесь основное содержимое документа.».

Алгоритм создания WEB-страницы:

1. Запустите файловый менеджер Total Commander и создайте используя команду F7 на диске D: папку «HTML».

2. Войдите в созданную папку и создайте в ней текстовый файл «firstpage.htm». Напомним, что для создания файла служит команда файлового менеджера TC (Total Commander) Shift +F4.

3. В окне запустившегося редактора введите следующий текст создаваемой WEB-страницы:

```
<HTML>
```

```
<TITLE>Моя первая страница!</TITLE>
```

```
<BODY>
```

```
Здесь основное содержимое документа.
```

```
</BODY>
```

```
</HTML>
```

4. Сохраните файл.

Для проверки результатов удобно не закрывать редактор с загруженным в него файлом, а с помощью клавиатурной команды Alt +Tab переключиться в ТС и нажав на сохранённом файле Enter открыть его в браузере. Обратите внимание, что файл будет открыт не текстовым редактором, а браузером (Интернет-обозревателем) в соответствии с расширением «htm». Если результаты проверки окажутся неудовлетворительными, то можно переключиться той же командой Alt +Tab в уже открытое окно редактора с текстом создаваемой WEB-страницы и исправить ошибки. После исправления ошибок не забудьте сохранить результат, поскольку при открытом файле исправления без сохранения будут только в окне редактора, а на диске останется ранее сохранённый вариант файла с ошибками.

Пример 2. Создайте WEB-страницу, содержащую нумерованный арабскими цифрами список учащихся вашего класса в алфавитном порядке.

Алгоритм создания WEB-страницы:

1. Создайте в папке «HTML» файл «listpupils.htm» любым удобным для вас способом. Обратите внимание, что если вы создавали файл в файловом менеджере Explorer, то в качестве расширения имени после введённого вами «htm» может быть автоматически до-

бавлено «txt». В этом случае файл будет необходимо переименовать, иначе откроется он не браузером, а текстовым редактором Блокнот.

2. Введите в окне редактора следующий текст на языке HTML, вставляя в элементы списка фамилии учеников своего класса и добавляя необходимое для полного списка количество элементов (тегов):

```
<HTML>
<TITLE>Список класса</TITLE>
<BODY bgcolor =»white» text =»black»>
<H1>Список учащихся класса</H1>
<OL type =1>
<LI>Первый по алфавиту ученик</LI>
<LI>Второй по алфавиту ученик</LI>
<LI>Последний по алфавиту ученик</LI>
</OL>
<HR>
</BODY>
</HTML>
```

3. Сохраните файл.

После выполнения этого алгоритма в папке «HTML» появится файл «listpupils.htm», содержащий WEB-документ со списком учеников вашего класса. Помните, что при проверке созданной WEB-страницы в браузере удобно окно редактора с её текстом оставить открытым и использовать команду Alt +Tab для переключения в окно файлового менеджера или в окно браузера.

При вводе в редакторе текста будущей WEB-страницы удобно парные теги записывать сразу, т.е. после открывающего тега вводите закрывающий и вставляйте между ними теги следующего уровня вложенности. Этот подход позволит предотвратить ошибки при самостоятельной работе.

В конце параграфа приведём более сложный пример WEB-страницы.

Пример 3. Создайте WEB-страницу, содержащую отформатированное стихотворение Пушкина А.С.

Алгоритм создания WEB-страницы:

1. Создайте в папке «HTML» файл «роем.htm».
2. Введите в редакторе в созданный файл следующий текст на языке HTML:

```
<HTML>
<TITLE>Стихотворение</TITLE>
<BODY bgcolor =»#AFEEEE» text =»black»>
<TABLE align=center width=50%>
<TR>
<TH align=center colspan=2>
***
</TH>
</TR>
<TR>
<TD colspan =2 align =left>
<FONT size =4>
О сколько нам открытий чудных
<BR>
Готовят просвещенья дух
<BR>
И опыт, сын ошибок трудных,
<BR>
И гений, парадоксов друг,
<BR>
И случай, Бог изобретатель...
</FONT>
</TD>
</TR>
<TR>
<TD width =20%></TD>
<TD>
<I>
Пушкин А.С.,
<SMALL>
1829
</SMALL>
```

```
</I>  
</TD>  
</TR>  
</TABLE>  
</BODY>  
</HTML>
```

3. Сохраните файл и проверьте результат работы в браузере.

Контрольные вопросы

1. Что представляет собой WEB-страница?
2. Какой программой открывается файл с расширением htm?
3. Какие способы создания текстового файла вы знаете?
4. Как можно переключаться между открытыми окнами приложений?
5. С помощью какого файлового менеджера вы создаёте текстовый файл? Почему?
6. Как следует изменить текст HTML-страницы в примере 2, чтобы нумерация учащихся осуществлялась римскими цифрами?
7. Что означает атрибут colspan в примере 3?
8. Что произойдёт, если в примере 3 между строками стихотворения убрать тег
?
9. Как можно смещать по горизонтали фамилию автора и год создания стихотворения?
10. Какая фраза будет иметь самый крупный шрифт в окне браузера в примере 2?
11. Как вы думаете, какое начертание будут иметь звёздочки перед началом стихотворения в браузере? Почему?

§11.3. Навигация по WEB-документу с помощью программ не визуального доступа к информации

Для просмотра Интернет-страниц (или WEB-страниц) служат специальные программы – Интернет обозреватели (браузеры). Такая программа получает по каналу связи HTML-файл с сервера, адрес которого указал пользователь. Полученный HTML-файл обрабатывается обозревателем в соответствии с указанными в нем тегами и

отображается на экране. При переходе по ссылке на другой документ, он, в свою очередь, поступает на компьютер пользователя и обрабатывается обозревателем. Существует несколько популярных Интернет обозревателей: Internet Explorer, Microsoft Edge, FireFox, Google chrome и др.

Приведем несколько часто используемых клавиатурных команд, работающих в большинстве браузеров:

- Открыть диалог «Открыть...» для указания адреса сайта – Ctrl +O;
- Переход к следующей ссылке или элементу управления – Tab;
- Переход к предыдущей ссылке или элементу управления – Shift +Tab;
- Перейти по ссылке – Enter;
- Возврат на предыдущую WEB-страницу – Alt +Стрелка влево;
- Переход на следующую WEB-страницу – Alt +Стрелка вправо;
- Обновить экран – F5;
- Открыть меню браузера – Alt;
- Заккрыть текущую вкладку – Ctrl +F4;
- Заккрыть браузер – Alt +F4.

Программа не визуального доступа к информации работает не с изображением на экране, а непосредственно с HTML-кодом загруженной страницы. JAWS или NVDA просматривает все теги, анализирует их и предоставляет пользователю удобную для восприятия без визуального контроля версию документа. После того, как браузер загрузит страницу, программа не визуального доступа начинает читать её последовательно сверху вниз. Если остановить речь в какой-либо точке, программа запоминает эту позицию в информации, хранящейся в буфере.

Программа JAWS читает WEB-страницы в режиме виртуального курсора. Виртуальный курсор функционирует аналогично РС-курору в текстовом редакторе, но сам виртуальный курсор при этом не виден визуально. Можно пользоваться стандартными командами чтения для перемещения по словам, строкам, предложениям или абзацам. Текст выделяется и копируется точно так же, как в текстовом редакторе.

Обратите внимание, что поскольку JAWS или NVDA просматривают HTML-код страницы, плохо написанный код может привести к тому, что страница будет прочитана неправильно.

В браузере ввод текста возможен только в полях редактирования. При переходе фокуса в такое поле программа незрительного доступа воспроизводит характерный звук. В обычном режиме ввод текста невозможен, клавиатура служит для ввода навигационных команд. Все эти команды можно вводить как со стандартной клавиатуры, так и с помощью брайлевского дисплея.

Приведем наиболее часто используемые команды программы незрительного доступа к информации (Большинство команд актуальны для JAWS и NVDA) для навигации по WEB-странице:

- Прочитать адресную строку – Ins + A;
- Вывести в отдельное окно список всех форм на текущей WEB-странице – Ins + F5;
- Вывести в отдельное окно список всех заголовков на текущей WEB-странице – INS + F6;
- Вывести в отдельное окно список всех ссылок на текущей WEB-странице – Ins + F7;
- Вывести в отдельное окно список всех фреймов на текущей WEB-странице – Ins + F9;
- Переместить фокус на следующую радиокнопку на текущей WEB-странице – A;
- Переместить фокус на следующую кнопку на текущей WEB-странице – B;
- Переместить фокус на следующий комбинированный список на текущей WEB-странице – C;
- Переместить фокус на следующее поле редактирования на текущей WEB-странице – E;
- Переместить фокус на следующее поле формы на текущей WEB-странице – F;
- Переместить фокус на следующий графический элемент на текущей WEB-странице – G;
- Переместить фокус на следующий заголовок на текущей WEB-странице – H;

- Переместить фокус на следующий элемент списка на текущей WEB-странице – I;
- Открыть диалоговое окно перехода на заданную строку на текущей WEB-странице – J;
- Вернуть фокус на строку, являющуюся текущей до перехода – Shift +J;
- Открыть диалоговое окно управления ориентирами – Ctrl+Shift+k;
- Переместить фокус на следующий ориентир на текущей WEB-странице – K;
- Переместить фокус на следующий список на текущей WEB-странице – L;
- Переместить фокус на следующий фрейм на текущей WEB-странице – M;
- Пропустить идущие подряд ссылки и переместить фокус на первый не ссылочный элемент на текущей WEB-странице – N;
- Переместить фокус на следующий абзац на текущей WEB-странице – P;
- Переместить фокус на следующий подобный элемент на текущей WEB-странице – S;
- Переместить фокус на следующую таблицу на текущей WEB-странице – T;
- Переместить фокус на следующую не посещенную ссылку на текущей WEB-странице – U;
- Переместить фокус на следующую посещенную ссылку на текущей WEB-странице – V;
- Переместить фокус на следующий флажок на текущей WEB-странице – X;
- Переместить фокус на следующую почтовую ссылку на текущей WEB-странице – \ (обратная косая черта);
- Переместить в таблице фокус в ячейку слева и прочитать её содержимое – Alt +Ctrl +Стрелка влево;
- Переместить в таблице фокус в ячейку справа и прочитать её содержимое • Alt +Ctrl +Стрелка вправо;
- Переместить в таблице фокус в ячейку сверху и прочитать её содержимое – Alt +Ctrl +Стрелка вверх;

- Переместить в таблице фокус в ячейку снизу и прочитать её содержимое – Alt +Ctrl +Стрелка вниз.

Для перехода к предыдущему элементу в однобуквенных командах добавляйте к указанным командам клавишу Shift.

Обратите внимание, что буквы команд как правило совпадают с названиями структурных элементов, к которым перемещают фокус, например, B – button (кнопка), L – List (список), P – paragraph (абзац) и т.д.

Нажмите и удерживайте клавишу Insert, одновременно быстро дважды нажмите F1, откроется окно справки. Чтобы открыть нужный раздел справки выберите его в списке и Нажмите клавишу F6.

Для получения справки по элементу, на который указывает фокус, используйте команду Ins +F1 (здесь нажатие однократное).

Для получения справки по клавиатурным командам браузера используйте команду Ins +W.

Для получения справки по командам программы невизуального доступа используйте команду Ins +H.

Для предоставления текста, сопровождающего графические объекты, служат атрибуты соответствующих тегов HTML. При правильном использовании этот текст предоставляет информацию, позволяющую понять роль данных объектов на WEB-странице. Программа невизуального доступа не может прочитать информацию из рисунка как такового, но она может прочитать этот сопроводительный текст.

Если страница хорошо проработана и все графические объекты имеют альтернативный текст, то навигация без визуального контроля по такой странице не должна вызывать трудностей, хотя потребует и значительно большего времени, чем визуальная.

При загрузке WEB-страницы программа невизуального доступа сообщает некоторую информацию о ней: количество областей, количество заголовков и количество ссылок. Внимательно слушайте такую информацию при загрузке страницы, что позволит получить некоторое первичное представление о её структуре. Затем, если на странице есть заголовки, можно прочитать их все используя клавишу H или команду Ins +F6. Если изучаемый интернет-ресурс предполагает диалог с пользователем, то на нем должны быть формы. Форму

образуют элементы, на которые пользователь может воздействовать, например, поля ввода, кнопки, флажки и т.д. Проверить наличие форм на странице можно используя клавиатурную команду **Ins +F5**. Она выведет в отдельное окно все элементы управления формы. Можно также посетить все элементы формы нажимая клавишу **F**. После этого вы будете иметь некоторое представление о структуре и содержимом страницы и используя команды навигации сможете приступить к поиску необходимой информации.

Обратите внимание, что при навигации по WEB-странице клавиатура не позволяет вводить символы. Нажатие буквенных клавиш приводит к переходу на соответствующий объект. Но при попадании в поле редактирования клавиатура переключается в режим ввода символов. При этом программа невизуального доступа воспроизводит характерный звуковой сигнал. При покидании поля ввода она также оповещает об этом воспроизводя соответствующий звуковой сигнал.

Контрольные вопросы

1. Для чего служит программа браузер?
2. Какие браузеры вы знаете?
3. Как в браузере загрузить заданную страницу?
4. Как в браузере переходить по ссылкам?
5. Что такое виртуальный курсор?
6. Как удобно запоминать однобуквенные команды перехода к структурному элементу на WEB-странице?
7. Какие команды чтения таблиц на WEB-странице вы знаете?
8. Как можно перемещать фокус к предыдущему элементу на WEB-странице?
9. Как можно осуществлять навигацию по WEB-странице с помощью брайлевского дисплея?
10. Какие команды получения помощи вы знаете?
11. С чего удобно начинать работу на незнакомой WEB-странице?

Глава 12

Информационно-коммуникационные технологии без визуального контроля

§12.1. Локальные вычислительные сети

Локальная вычислительная сеть (ЛВС) обычно создается для совместного использования различных цифровых устройств и данных в какой-либо одной организации. Иными словами, локальная сеть – это совокупность сетевых устройств (компьютеров, принтеров, накопителей данных и др.) и каналов связи, объединяющих такие устройства в структуру с определенной конфигурацией, а также сетевого программного обеспечения, управляющего работой сети. Объединение сетевых устройств в единую систему позволяет не только обмениваться данными, совместно использовать вычислительные мощности, но и обеспечить более эффективную эксплуатацию оборудования. Например, если в каком-либо офисе несколько сотрудников должны распечатывать документы на принтере, значительно дешевле организовать совместное использование одного сетевого принтера всеми сотрудниками. Такой подход к использованию оборудования снижает финансовые затраты и упрощает обслуживание.

Физический (геометрический) способ соединения сетевых устройств каналами передачи данных называется топологией. Топология в значительной степени определяет многие важные свойства сети, например такие, как надежность, производительность и др. В некоторых топологиях компьютер может быть связан каналом передачи данных только с одним соседним компьютером. В более сложных конфигурациях любой компьютер соединён каналом передачи данных с каждым другим элементом сети. Существуют разные подходы к классификации топологий сетей. Обычно различают три основных вида топологий:

- Топология типа «звезда»;
- Топология типа «кольцо»;
- топология типа «шина».

В топологии «звезда» информация между компьютерами (и другими устройствами) сети передается через единый центральный узел. В качестве центрального узла может использоваться компьютер или специальное устройство – концентратор (Hub).

Данная топология обладает высоким быстродействием, поскольку общая производительность сети определяется производительностью центрального узла, а быстродействие каждого компьютера сети не влияет на скорость её работы в целом. При этом каждый компьютер получает только те данные, которые предназначены непосредственно ему. Это повышает скорость его работы и обеспечивает конфиденциальность информации.

При соединении по топологии «звезда» при повреждении одного из соединений от сети отключается только один компьютер, а остальная часть сети продолжает функционировать.

Однако, у топологии «звезда» есть и недостатки. К их числу можно отнести низкую надежность, поскольку при выходе из строя центрального узла вся сеть прекращает функционировать.

Заметим, что топологию типа «звезда» можно рассматривать как разновидность «дерева», имеющего корень с ответвлением к каждому подключенному устройству.

При топологии «кольцо» все компьютеры подключаются к каналу передачи данных, замкнутому в кольцо.

Передача информации в такой сети происходит следующим образом. Маркер (специальный сигнал) последовательно, от одного компьютера к другому, передается до тех пор, пока его не получит тот, которому требуется отправить данные. Получив маркер, этот компьютер создает так называемый «пакет», в который помещает адрес получателя и данные, а затем отправляет этот пакет по кольцу. Данные проходят по кольцу через каждый компьютер, пока не окажутся у того, чей адрес совпадает с адресом получателя. После этого принимающий компьютер посылает источнику информации подтверждение факта получения данных. Получив подтверждение, передающий компьютер создает новый маркер и возвращает его в сеть.

Основное преимущество топологии «кольцо» состоит в высокой эффективности передачи сообщений, т.к. можно отправлять несколь-

ко сообщений друг за другом по кольцу. Другими словами компьютер, отправив первое сообщение, может отправлять за ним следующее сообщение, не дожидаясь, когда первое достигнет адресата.

К недостаткам данной топологии можно отнести низкую надёжность сети, поскольку отказ любого компьютера влечет за собой нарушение работы всей сети.

Следует также отметить, что при большом количестве компьютеров в сети скорость её работы замедляется, т.к. вся информация проходит через каждый компьютер, а их быстродействие ограничено. При этом общая производительность сети определяется производительностью самого медленного компьютера.

При топологии «шина» все клиенты подключены к общему каналу передачи данных. При этом они могут непосредственно вступать в контакт с любым компьютером, имеющимся в сети.

Передача информации в сети, организованной по топологии «шина», происходит следующим образом. Данные в виде сигналов передаются всем компьютерам сети. Однако информацию принимает только тот компьютер, адрес которого соответствует адресу получателя. При этом в каждый момент времени только один компьютер может вести передачу данных.

Важным преимуществом сети, построенной по типу топологии «шина», является её высокая надёжность, т.к. работоспособность сети не зависит от работоспособности отдельных компьютеров. При повреждении одного сетевого устройства с общей шиной, оно отключается от сети, при этом вся остальная сеть продолжает функционировать. Таким образом, топология «шина» достаточно устойчива, однако, при повреждении самой шины вся сеть выходит из строя.

К недостаткам топологии «шина» относится низкая скорость передачи данных, поскольку вся информация проходит по одному каналу (шине). Быстродействие сети зависит от числа подключенных компьютеров. Чем больше компьютеров подключено к сети, тем медленнее идет передача информации от одного компьютера к другому.

Выше были рассмотрены основные топологии ЛВС. Однако на практике при создании сети может использоваться сочетание нескольких топологий. Например, компьютеры в одном отделе мо-

гут быть соединены по схеме «звезда», а в другом отделе по схеме «шина», и между этими отделами проложена линия связи.

Все описанные выше сети могут быть организованы двумя способами: на основе одноранговой технологии и технологии «клиент-сервер» (сеть с выделенным сервером).

Сеть с выделенным сервером имеет центральный компьютер – сервер, с помощью которого происходит управление работой всей сети. Остальные, входящие в сеть компьютеры, называются рабочими станциями.

Обычно термином «сервер» обозначается сочетание аппаратных и программных средств, которое служит для управления сетевыми ресурсами. Сервер – это компьютер, предоставляющий услуги другим компьютерам сети. Сервер обеспечивает распределение доступа пользователей к различным сетевым устройствам и информационным ресурсам. На сервере могут быть записаны программы, которыми пользуются все компьютеры сети.

Достаточно часто в сетях с выделенным сервером его дисковая память является общим ресурсом. Серверы, доступным ресурсом которых является дисковая память, называются файл-серверами.

Каждый компьютер сети имеет уникальное сетевое имя, позволяющее однозначно его идентифицировать. Каждому пользователю серверной сети необходимо иметь свое сетевое имя и сетевой пароль. Имена компьютеров, сетевые имена и пароли пользователей хранятся на сервере.

Совокупность приемов разделения и ограничения прав доступа участников компьютерной сети к ресурсам называется политикой сети.

Обеспечением работоспособности сети и ее администрированием занимается системный администратор – человек, управляющий организацией работы компьютерной сети.

Рабочая станция – это индивидуальное рабочее место пользователя. На рабочих станциях устанавливается обычная операционная система. Кроме того, на рабочих станциях устанавливается клиентская часть сетевой операционной системы. Полноправным владельцем всех ресурсов рабочей станции является пользователь, тогда как

ресурсы файл-сервера могут быть использованы всеми пользователями.

В одноранговых сетях все компьютеры, как правило, имеют доступ к ресурсам других компьютеров, т. е. все компьютеры сети являются равноправными. Одноранговая ЛВС предоставляет возможность такой организации работы компьютерной сети, при которой каждая рабочая станция одновременно может быть и сервером.

Преимущество одноранговых сетей заключается в том, что разделяемыми ресурсами могут являться ресурсы всех компьютеров в сети и нет необходимости копировать все используемые сразу несколькими пользователями файлы на сервер. В принципе, любой пользователь сети имеет возможность использовать все данные, хранящиеся на других компьютерах сети, а также устройства, подключенные к ним.

Затраты на организацию одноранговых вычислительных сетей относительно небольшие. Однако при увеличении числа рабочих станций эффективность их использования резко уменьшается. Основным недостатком работы одноранговой сети заключается в значительном увеличении времени решения прикладных задач. Это связано с тем, что каждый компьютер сети обрабатывает все запросы, идущие со стороны других пользователей. Следовательно, в одноранговых сетях каждый компьютер работает значительно интенсивнее, чем в автономном режиме.

Одноранговые ЛВС используются только для небольших рабочих групп, а все сетевые архитектуры для крупномасштабных сетей поддерживают технологию «клиент-сервер».

Кроме сетевого оборудования для работы ЛВС требуется сетевая операционная система. В сетевой операционной системе имеются возможности обслуживания специфических задач сети. К сетевым операционным системам относятся некоторые версии MS Windows, NetWare, UNIX, Linux, FreeBSD и др.

Локальные компьютерные сети можно объединять друг с другом, даже если между ними большие расстояния. При соединении двух или более сетей между собой возникает межсетевое соединение и образуется глобальная компьютерная сеть.

Обычно для обеспечения устойчивого канала передачи данных на небольшие расстояния (в пределах здания) используется специальный кабель «витая пара».

Одним из наиболее мощных и надежных каналов передачи данных является оптоволоконное соединение. В оптоволоконных каналах связи используется известное из физики явление полного отражения света от внутренней поверхности световода, что позволяет передавать потоки света внутри оптоволоконного кабеля на большие расстояния практически без потерь. В качестве источников света в оптоволоконном кабеле используются светоиспускающие диоды (LED – Light Emitting Diode) или лазерные диоды, а в качестве приемников – фотоэлементы.

Оптоволоконные каналы связи, несмотря на их более высокую стоимость по сравнению с другими видами связи, получают все большее распространение, причем не только для связи на небольшие расстояния, но и на внутригородских и междугородных участках.

Контрольные вопросы

1. Что такое локальная вычислительная сеть?
2. Из каких элементов состоит ЛВС?
3. Для чего создаются ЛВС?
4. Что такое топология сети?
5. Какие основные виды топологий вы знаете?
6. Расскажите о топологии:
 - А) «звезда»;
 - Б) «кольцо»;
 - В) «шина».
7. Что такое канал передачи данных?
8. Какие способы организации сети вы знаете?
9. Что такое технология «клиент-сервер»?
10. Какие функции в сети выполняет сервер?
11. Что такое политика сети?
12. Чем занимается системный администратор?
13. Что такое рабочая станция?
14. Какие каналы передачи данных вы знаете?

15. На каком физическом явлении основана работа оптоволоконного канала передачи данных?

§12.2. IP-адрес и маска подсети

В предыдущем параграфе уже говорилось, что локальная вычислительная сеть состоит из сетевых устройств, связывающих их каналов передачи данных и программного обеспечения, обслуживающего сеть. Одной из основных функций сетевого программного обеспечения является приём и передача данных по сетевым каналам от одного (передающего) к другому (принимающему) устройству. При этом принимающее и передающее устройства должны работать в одном стандарте передаваемых данных. Принимающий компьютер получает данные, адресованные именно ему и интерпретирует их в соответствии с тем, как это определяет передающий компьютер.

Возможность передачи данных по сети обеспечивает протокол, который содержит набор правил для взаимодействия между сетевыми устройствами. Протокол включает в себя обращение одного сетевого устройства к другому по имени, обмен информацией в виде передачи данных и разрыв соединения по окончании передачи.

В настоящее время основной сетевой моделью передачи данных является TCP/IP. Эта модель определяет набор правил для передачи данных от отправителя к получателю. Она объединяет в себе два протокола:

- TCP – Transmission Control Protocol (протокол управления передачей данных). Этот протокол обеспечивает установку соединения между передающим и получающим устройствами;
- IP – Internet Protocol. Это протокол межсетевого взаимодействия. Протокол обеспечивает передачу данных в сети (не устанавливая соединение), а также объединение сетей, работающих на разных принципах и построение маршрута от отправителя к получателю.

Общеизвестно, что в глобальной сети Интернет каждое устройство имеет свой IP-адрес. IP-адрес представляет собой уникальный адрес устройства в сети, построенной на основе модели TCP/IP, и используется для идентификации компьютеров в сети. Т.е. IP-адресация служит для однозначного определения отправителя и получа-

теля данных. Аналогично тому, как производится доставка почтовой посылки от одного почтового адреса к другому, определяя страну, город, название улицы и номер дома получателя.

В настоящее время существует две версии IP-адресов: IPv4 и IPv6. Основное различие заключается в их размерах. IPv4 занимает 4 байта, а IPv6 – 16 байтов. Большой размер адреса версии 6 позволяет использовать гораздо больше уникальных сетевых адресов, чем IPv4. Именно с целью увеличения количества адресов IPv6 и был разработан, поскольку адресов размером 4 байта с развитием сетей стало не хватать для всех устройств. Между IPv4 и IPv6 существует ряд и других Различий, обсуждение которых выходит за рамки школьного курса информатики. Ниже в качестве примера будут рассмотрены IP-адреса версии 4.

Как уже было отмечено, длина адреса IPv4 составляет 4 байта или 32 бита. Для удобства IP-адреса принято записывать в десятичной форме в виде четырех групп цифр от 0 до 255 разделённых точками. Например:

222.171.193.25

Каждому десятичному значению, понятному для человека, соответствует его двоичная запись, являющаяся в некоторых случаях более удобной, чем десятичная:

222 = 110111102

171 = 101010112

193 = 110000012

25 = 000110012

Таким образом, в двоичной записи указанный IP-адрес будет иметь вид:

11011110.10101011.11000001.00011001

Заметим, что для краткости у десятичных значений индекс 10 обычно не записывается. А в записи IP-адреса в двоичной системе также не записывается индекс 2.

Следует различать внутренние и внешние IP-адреса. Построение глобальной сети осуществляется на основе объединения более мелких подсетей, характеризующихся одинаковой старшей частью внутренних IP-адресов. Например:

222.171.193.1

222.171.193.2

222.171.193.3

...

222.171.193.254

Т.е. IP-адрес состоит из двух частей –адреса подсети и адреса хоста. Хостом в данном случае является любое сетевое устройство, имеющее собственный IP-адрес в локальной сети. Если рассмотреть IP-адрес 222.171.193.2, то адресом подсети здесь будет 222.171.193.0, а адресом хоста – 0.0.0.2.

Можно сказать, что IP-адрес 222.171.193.2 принадлежит сетевому устройству под условным номером 2.

Для разделения IP-адреса на такие две части служит маска подсети. Как и IP-адрес, маска подсети состоит из четырёх групп цифр и имеет длину 32 бита. Обычно маска подсети записывается в виде:

255.255.255.0

В случае с приведённым выше адресом 222.171.193.2 эта маска соответствует тому, что значение 222.171.193.0 (первые 3 группы цифр) является адресом подсети, а 0.0.0.2 (последняя группа) является адресом хоста.

Данное значение IP-адреса и маски подсети можно записать в виде префикса:

222.171.193.2 /24

Эта запись означает, что первые 24 бита IP-адреса (первые три группы цифр) соответствуют адресу подсети, а последние 8 бит (четвертая группа цифр) соответствуют адресу хоста.

На практике не всегда первые три группы цифр определяют номер подсети, а последняя – адрес хоста. Часто при построении крупных локальных вычислительных сетей имеет место разбиение третьего байта IP-адреса на две части, первая из которых относится к адресу подсети, а вторая – к адресу хоста. В таком случае маска называется маской переменной длины.

Если маска подсети имеет префикс /20, например:

222.171.193.2 /20,

то к адресу подсети в данном случае относятся первые 20 из 32 бит IP-адреса, а оставшиеся 12 бит относятся соответственно к адресу хоста. В этом случае маска подсети будет:

255.255.240.0

Более наглядно эта маска будет выглядеть в двоичной записи:

11111111.11111111.11110000.00000000

Для определения адреса подсети используется логический оператор and. Если применить его последовательно к каждому двоичному знаку (биту) маски и IP-адреса, то в результате получится адрес подсети:

11111111.11111111.11110000.00000000

and

11011110.10101011.11000001.00000010

равно

11011110.10101011.11000000.00000000

Преобразовав результат в десятичный вид, получим адрес подсети:

222.171.192.0

Очевидно, что адрес хоста в десятичной форме будет:

0.0.1.2

В мировом масштабе распределением IP-адресов занимается международная некоммерческая организация ICANN (Internet Corporation for Assigned Names and Numbers) – «корпорация по управлению доменными именами и IP-адресами». ICANN распределяет диапазоны IP-адресов между региональными регистраторами, закрепленными за определенными территориями земного шара.

Для создания локальных вычислительных сетей зарезервированы специальные диапазоны IP-адресов, которые не используются во внешней сети. Эти адреса в границах своей подсети присваивают администраторы этих подсетей, либо маршрутизаторы. Для построения ЛВС рекомендовано использование следующих диапазонов частных IP-адресов:

10.0.0.0 /8

100.64.0.0 /10

172.16.0.0 /12

192.168.0.0 /16

Заметим, что это условие носит лишь рекомендательный характер и не является обязательным для исполнения. Однако, на практике в небольших локальных сетях (в квартире или в школе) используется диапазон:

192.168.0.0 /16

Для подключения локальной подсети к глобальной сети Интернет используется технология преобразования сетевых адресов NAT (Network Address Translation). При помощи данной технологии, проходя через маршрутизатор, IP-адрес локальной подсети заменяется глобальным IP-адресом, который присваивает интернет-провайдер. Начало использования технологии NAT явилось следствием нехватки адресов IPv4. Введение в практику IPv6 с огромным числом IP-адресов делает ненужным использование технологии NAT.

Чтобы узнать локальный IP-адрес своего компьютера и маску подсети, а также некоторую другую информацию о локальной сети, можно в командной строке Windows ввести команду `ipconfig`. Напомним, что для вызова командной строки в главном меню в пункте «Выполнить» следует ввести команду `CMD`. Активным станет окно с командной строкой (окно консоли), в которой уже следует ввести `ipconfig`. На экран будет выведено достаточно много информации, среди которой будут строки:

Локальный IPv6-адрес канала: `fe80::ec72:5319:ed2e:7c1b%11`

IPv4-адрес: `192.168.0.17`

Маска подсети: `255.255.255.0`

Напомним также, что при использовании программы невидимого доступа JAWS for Windows читать информацию в окне консоли следует в режиме JAWS-курсора.

Контрольные вопросы

1. Какую роль выполняет протокол передачи данных в компьютерной сети?
2. Что такое TCP/IP?
3. Какую роль играет IP-адрес?
4. Как выглядит IP-адрес?
5. В чём разница между IPV4 и IPV6?

6. Для чего был создан IPV6?
7. Почему в каждой группе адреса IPV4 могут находиться числа только от 0 до 255?
8. Что такое адрес подсети и адрес хоста?
9. С помощью чего IP-адрес разделяется на две части?
10. Как можно короче записать значение IP-адреса и маски подсети?
11. Что такое маска переменной длины?
12. Как можно получить адрес подсети по IP-адресу и маске?
13. Как можно узнать IP-адрес своего компьютера?

§12.3. Поиск информации в Интернет

Глобальная сеть Интернет предоставляет доступ к самой разнообразной информации, распределённой по сотням тысяч информационных серверов. Для решения задачи поиска информации предназначены специальные поисковые системы, позволяющие найти документы, содержащие необходимые пользователю сведения. Развитие таких поисковых систем началось с девяностых годов прошлого века.

Принципы работы популярных поисковых систем практически одинаковы. Пользователь, задав ключевые слова и условия поиска, в ответ получает список документов, соответствующих заданным параметрам. Этот список сортируется по определенным критериям так, чтобы вверху списка оказались те документы, которые наиболее соответствуют запросу.

Единой оптимальной схемы поиска в Интернет не существует. В любой поисковой системе результат будет зависеть от конструкции запроса. Чем грамотнее подобраны ключевые слова и параметры поиска, тем ближе к желаемому будут результаты.

В качестве примера ниже будет рассмотрена поисковая система Google. Как уже говорилось, существенных отличий другие распространённые системы иметь не будут.

Поисковая система Google оснащена специальными сервисами, призванными значительно упростить решение различных поисковых задач. Разработчики, стремясь облегчить поиск информации,

создали специальный язык поисковых запросов Google — специальные операторы и функции обработчика поисковых запросов.

Приведем некоторые основные конструкции поисковых запросов Google.

Оператор AND – Логическое «И». По умолчанию к каждому введенному в запросе слову поисковая система GOOGLE применяет операцию логического «И». Это значит, что на запрос «Пушкин стихотворения осень» Google выдаст только те страницы, которые одновременно содержат все три слова «Пушкин», «стихотворения» и «осень». Заметим, что в большинстве случаев результат подобного запроса приводит к искомой странице.

Обратите внимание, что кавычки в запросе указывать необязательно. Здесь они используются для отделения текста запроса от основного текста. Если текст запроса взять в кавычки, это будет означать поиск точного совпадения (подробнее об этом сказано ниже).

Оператор OR – Логическое «ИЛИ». Этот оператор будет искать страницы, содержащие хотя бы одну из фраз, объединяемых этим оператором. Например, пусть нужно найти журнал «Школьный вестник» или «Наша жизнь», тогда запрос будет выглядеть так: «Школьный вестник or Наша жизнь».

Оператор «» (кавычки) – Точное совпадение. Сложные алгоритмы поиска Google учитывают морфологию языка, различные особенности построения WEB-документа и не предполагают, что найденные страницы будут содержать в точности ту фразу, которая указана в строке поиска. Слова могут быть разбросаны по всей странице и даже иметь другую форму, что в большинстве случаев очень удобно. Но Если требуется найти точное совпадение, например, текст песни по одной известной строке, то следует взять текст запроса в кавычки.

Оператор + (плюс) – Выделение важных слов. Чтобы сделать акцент на одно или несколько слов нужно использовать «+». Это поможет системе понять, какие из ключевых слов более важны, и сформулировать результаты поиска точнее. Например, «Пушкин +стихотворения».

Оператор – (минус) – Исключение нежелательных слов. Полученные результаты нередко засоряет лишняя информация. Чтобы не тра-

тить время на ее просмотр, можно наложить на результаты поиска фильтр. Для этого нужно перед нежелательными словами поставить «-«. Например, «Наша жизнь +журнал -декабрь».

Оператор `site:` – Поиск по конкретному сайту. Если известно, что необходимая информация есть на определенном сайте, то можно ограничить поиск рамками только этого сайта. Для этого используется оператор `site:`. Этот прием очень удобен, если нужно найти информацию на каком-то ресурсе у которого плохо или вообще не работает внутренний поиск. Например, «`site:mgppri.ru` молодые ученые».

Оператор `related:` – Похожие страницы. Используя оператор `related:` можно находить похожие страницы. Это удобно для владельцев сайтов для определения дружественных или конкурирующих сайтов. Например, «`related:microsoft.com`».

Оператор `link:` – Ссылающиеся страницы. С помощью этого оператора можно найти все страницы, на которых есть ссылка на указанную после него. Это можно использовать для проверки популярности ресурса. Чем больше ссылающихся страниц, тем больше популярность проекта. Например, «`link:tiflocomp.ru`». Будут найдены все страницы, ссылающиеся на `tiflocomp.ru`.

Оператор `~` (тильда) – Использование синонимов. Если нужно чтобы в результаты поиска вошли синонимы определенного слова, то перед ним нужно поставить символ «~». Например, «почтовые ~клиенты». Будут найдены страницы с информацией о почтовых клиентах, о почтовых программах и т.д.

Оператор `filetype:` – Поиск документов конкретного типа. Этот оператор позволяет искать информацию в документах конкретного типа по его расширению. Например, «Пушкин `filetype:rtf`»

Оператор `define:` – Поиск определений. Этот оператор Позволяет найти определение неизвестного слова или понятия. Например, «`define:таксидермист`».

Здесь описаны только наиболее часто используемые операторы, полный их список можно найти на сайте `google.com`.

Заметим, что операторы языка поисковых запросов Google можно комбинировать для достижения более релевантных результатов. По-

исковая система Google это очень мощный инструмент поиска информации. Знание основных возможностей этого инструмента облегчает процедуру поиска.

Приведём алгоритм использования поисковой системы Google:

1. Запустите браузер с помощью ярлыка на «Рабочем столе» или любым другим способом.

2. Загрузите в браузер страницу www.google.com.

3. С помощью клавиши E найдите поисковую строку и убедитесь, что JAWS переключился в режим форм (JAWS должен издать характерный звук). Заметим, что как правило после загрузки страницы фокус сразу оказывается в поисковой строке и никаких дополнительных действий предпринимать не надо.

4. Введите поисковый запрос и нажмите клавишу Enter.

5. На страницу будут выведены результаты поиска, чтение которых удобно начать перемещаясь по заголовкам с помощью клавиши H.

6. Выбрав необходимый результат, перейдите на его страницу нажав Enter на соответствующей ему ссылке.

7. Если найденная таким образом страница вас не удовлетворяет, вернуться обратно ко всем результатам поиска можно закрыв текущую вкладку командой Ctrl +F4.

Ещё раз подчеркнём, что при использовании других поисковых систем принципиальных отличий не возникнет.

Контрольные вопросы

1. Зачем нужны поисковые системы?
2. Опишите общий принцип работы поисковой системы.
3. Как добиться лучшего результата поиска?
4. Какие операторы поисковых запросов вы знаете?
5. Что изменится, если ключевые слова запроса взять в кавычки?
6. Сколько операторов может быть в одном запросе?
7. Опишите алгоритм поиска информации с помощью поисковой системы.

§12.4. Сервисы сети Интернет

Глобальная сеть Интернет предоставляет пользователям не только возможность поиска нужной информации, но и другие сервисы и ресурсы, которые давно стали неотъемлемой частью современной жизни. Основные сервисы Интернет предоставляют пользователям самый широкий набор возможностей. Список сервисов постоянно растёт, расширяя спектр возможностей пользователей сети Интернет. Практически каждый пользователь Интернет встречался с такими сервисами, как:

- электронная почта;
- форумы, общение в реальном времени;
- мессенджеры;
- файловые архивы;
- облачные хранилища данных;
- IP-телефония;
- Интернет-радио и Интернет-телевидение.

Этот список далеко не полон и не исчерпывает всех возможностей сети Интернет. Ниже описан один из наиболее востребованных сервисов.

Электронная почта (E-Mail) – этот сервис обеспечивает возможность обмена письменными сообщениями одного пользователя с одним или группой абонентов. Электронная почта по своему действию похожа на обычную, поэтому освоение принципов ее работы не должно вызвать особых затруднений у пользователя. Её главное отличие в том, что пересылаются не физические предметы (письма, бандероли, посылки), а их информационные образы. Поэтому нельзя рассматривать электронную почту как альтернативу почтовым службам, существующим во всех уголках планеты. Можно говорить лишь о дополнении одного вида связи другим.

Основным достоинством электронной почты является оперативность доставки писем. Обычно электронные письма достигают любой точки земного шара за несколько секунд. При этом она позволяет передавать не только текст, но и файлы относительно небольших размеров любого типа (фотографии, программы, музыку и др.).

Электронная почта не использует географическую адресацию. Адрес электронной почты выглядит так:

имя_почтового_ящика@имя_домена,
например, mike112@mail.ru.

Символ «@» — это разделитель, показывающий, где в адресе заканчивается имя пользователя и начинается имя домена.

Электронная почта может работать по принципу клиент-сервер. На компьютере пользователя стоит клиентская почтовая программа, которая периодически связывается с почтовым сервером, на котором зарегистрирован электронный почтовый ящик. В ходе сеанса связи происходит отправка подготовленной ранее исходящей корреспонденции и получение входящей. После этого сеанс связи заканчивается и компьютеры разъединяются. Создание писем, работа с входящей почтой производится пользователем с помощью той же клиентской программы на своем компьютере без подключения к Интернет.

Существует достаточно много почтовых программ, часть которых распространяется бесплатно. Все они в значительной части похожи и могут различаться по внешнему виду, дополнительным возможностям и по степени соответствия принятым стандартам, например, The Bat, Microsoft Outlook и др.

Настройки различных почтовых клиентов могут отличаться друг от друга. Однако, основные настройки являются общими для всех программ. Так, для того, чтобы настроить почтовую программу для работы с определенным почтовым ящиком, необходимо задать следующие параметры:

- Зарегистрированный электронный адрес;
- Адрес SMTP-сервера;
- Адрес POP3-сервера (или IMAP-сервера);
- Имя пользователя и пароль для доступа к почтовому ящику (имя пользователя и пароль задаются при создании почтового ящика).

Каждая почтовая программа после установки автоматически создает три папки: для входящих писем, для отправленных (в ней сохраняются копии отправленных писем) и папку для удалённых писем (эта папка полезна в случае ошибочного удаления письма). Обычно почтовые программы позволяют создавать дополнительные папки

для детальной сортировки корреспонденции. Также в почтовых клиентах присутствует возможность автоматической фильтрации входящей почты. Например, автоматически направлять в папку для удалённых писем сообщения с определенного адреса. Дополнительные возможности можно изучать по мере необходимости. Они подробно описаны в документации почтовой программы.

Пользоваться электронной почтой можно и не имея на своем компьютере специальной программы. Современные почтовые серверы предоставляют возможность завести бесплатный почтовый ящик и работать с ним, используя только браузер через WEB-интерфейс. Использование WEB-интерфейса позволяет просматривать почту с любого компьютера, подключенного к сети Интернет.

Заметим, что программы невидимого доступа к информации JAWS for Windows и NVDA одинаково хорошо поддерживают работу как с помощью почтового клиента, так и через WEB-интерфейс.

Независимо от способа использования электронной почты электронные письма имеют стандартный вид. Как и обычное письмо, электронное состоит из конверта, называемого заголовком письма, и самого текстового сообщения— тела письма.

Заголовок письма содержит определённую служебную информацию. Некоторая информация вписывается в заголовок автоматически и предназначена для правильной маршрутизации письма, уведомления почтовой программы получателя о дате, времени отправки письма, об имеющихся вложенных файлах и т.д. Часть информации вводится пользователем при создании письма:

- Кому: (To:) – адрес получателя письма;
- Копия: (Cc:) – перечень адресов, на которые необходимо отправить копию данного письма;
- Скрытая: (Bcc:) – адреса, на которые тоже нужно выслать копию, но так, чтобы про это не знал основной адресат;
- Тема: (Subject:) – тема письма.

Из перечисленных полей только поле «Кому:» обязательно к заполнению.

Знак «:» (двоеточие) обычно присутствует после имени поля как в почтовом клиенте, так и в WEB-интерфейсе. После имени поля

в круглых скобках указаны обозначения полей в англоязычном интерфейсе.

Тело письма содержит собственно текст сообщения. К письму можно прикрепить файлы, которые будут отправлены адресату вместе с письмом в виде приложения к нему. Для этого нужно нажать с помощью клавиши Пробел соответствующую кнопку и в раскрывшемся стандартном окне в списке выбрать файл с жесткого диска.

В конце параграфа коротко рассмотрим ещё одну службу Интернет – Chat. Chat («чат») в переводе с английского означает «дружеский разговор, беседа, болтовня». В сети Интернет за данным термином закрепилось значение «общения в режиме реального времени». Существуют специальные серверы, предоставляющие WEB-интерфейс для общения, т.е. нет необходимости устанавливать специальное программное обеспечение, а достаточно зайти на страницу такого сервера с помощью любого браузера и зарегистрироваться – ввести псевдоним, под которым пользователь будет работать. После этого можно начинать общение.

В настоящее время существует достаточно широкий выбор различных платформ, реализующих как текстовый, так и голосовой Chat. К сожалению, не все они обеспечивают комфортную работу с программами невизуального доступа. Вы можете самостоятельно попробовать несколько вариантов и выбрать наиболее предпочтительный для себя.

Контрольные вопросы

1. Какие сервисы сети Интернет вы знаете?
2. Что такое электронная почта?
3. Какое достоинство электронной почты вы считаете основным? Почему?
4. Как выглядит адрес электронной почты?
5. Какие два способа работы с электронной почтой вы знаете?
6. Какие почтовые программы вы знаете?
7. Из каких полей состоит конверт электронного письма?
8. Чем отличаются поля «Копия:» и «Скрытая копия:»?

9. Какие поля обязательны для заполнения при создании электронного письма?

10. Что можно передавать по электронной почте кроме текстовых сообщений?

11. Что такое Chat?

Глава 13

Компьютерное моделирование

§13.1. Что такое моделирование

Очевидно, что некоторые процессы, явления и объекты окружающего мира крайне трудно или невозможно изучать непосредственно. В этом случае для изучения создаётся модель объекта. Каждый реальный объект имеет набор различных свойств, которые необходимо рассматривать при его изучении. Для решения практических задач в процессе построения модели выделяются главные, наиболее существенные для изучения реального объекта свойства. Так, например, модель самолета должна иметь геометрическое соответствие оригиналу для сохранения изучаемых аэродинамических свойств, модель атома – адекватно отражать физические взаимодействия, архитектурный макет города – ландшафт, взаимное расположение улиц и зданий и т.д.

Модель – это новый объект, адекватно отражающий существенные при решении данной задачи особенности изучаемого реального объекта, явления или процесса.

Рассмотрим несколько известных моделей:

1. Модель части солнечной системы – теллурий. Это наглядная модель, с помощью которой можно продемонстрировать годовое движение Земли вокруг Солнца и суточное вращение Земли вокруг своей оси, объяснить смену времен года и такие явления, как солнечное и лунное затмения. Это материальная модель, представляющая в уменьшенном виде реальный объект (систему – солнце, земля луна).

2. Известные из курса физики модели – материальная точка, математический маятник и идеальный газ. Это абстрактные модели, позволяющие решать физические задачи, получая при этом практически значимые результаты, соответствующие поведению реальных объектов.

3. Компьютерный симулятор управления самолётом. Это информационная модель, позволяющая в безопасных условиях виртуаль-

ной (компьютерной) реальности получить некоторый опыт управления самолётом.

Таким образом, модель необходима для изучения явления, процесса или объекта в лабораторных безопасных условиях. При этом изучение в реальной ситуации невозможно, существенно затруднено или опасно. Например, при разработке нового самолётного двигателя необходимо проверить его поведение в сложных полетных условиях. Проверка в реальности подвергнет опасности жизнь летчика-испытателя. Однако, используя информационную модель можно на компьютере проверить работу двигателя в любых полётных условиях. Следует подчеркнуть, что данное компьютерное моделирование должно опираться на физические законы, а также математические уравнения и формулы, описывающие работу двигателя.

В разных науках явления, процессы и объекты исследуются с различных точек зрения, в следствие чего создаются различные типы моделей. В механике изучаются процессы движения и взаимодействия материальных тел, в химии изучается внутреннее строение веществ, в биологии – поведение живых организмов и т.д. Например, человек в разных науках исследуется с использованием различных моделей. В механике его можно рассматривать как материальную точку, в химии – как объект, состоящий из различных химических веществ, в биологии – как систему, стремящуюся к самосохранению и т.д.

Возможно также, что разные объекты описываются одной моделью. Так, в механике различные материальные тела (от планеты до песчинки) часто рассматриваются как материальные точки.

Один и тот же объект может иметь множество моделей, а разные объекты описываться одной моделью.

Модель создаётся для достижения определённой цели проводимого исследования (получения конкретного результата). Свойства строящейся модели выбираются в зависимости от поставленной цели исследования. Такие свойства являются существенными для данной модели с точки зрения цели моделирования. Существенность и несущественность определенных свойств и признаков – понятия относительные, они зависят от решаемой задачи. Модель создается для

получения информации об процессе, явлении или объекте, необходимой для решения конкретной практической задачи.

Никакая модель не может полностью заменить реальный объект. Но при решении конкретной практической задачи, когда важно выяснить определенные свойства изучаемого объекта, модель является полезным, а иногда единственным возможным инструментом исследования.

Моделирование – это метод познания, состоящий в создании и исследовании моделей реальных процессов, явлений или объектов.

Решение любой практической задачи всегда связано с исследованием, преобразованием некоторого объекта (материального или информационного) или управления им.

Все многообразие моделей делится на три вида:

- Материальные модели – некие реальные предметы, макеты, муляжи, уменьшенные или увеличенные копии, воспроизводящие внешний вид моделируемого объекта, его структуру (глобус, модель кристаллической решетки, радиоуправляемая модель самолета, велотренажер);

- Абстрактные модели (геометрическая точка, материальная точка, математический маятник, идеальный газ);

- Информационные модели – описание моделируемого объекта на каком-либо формальном языке (словесное описание, чертежи, карты, формулы, компьютерные программы и т.д.).

Процесс создания модели можно разделить на следующие шаги:

- Постановка цели моделирования;
- Анализ всех известных свойств моделируемого объекта;
- Выделение существенных свойств и признаков моделируемого объекта (для одного и того же объекта при разных целях моделирования существенными будут являться разные свойства);
- Выбор формы представления модели, которая полностью соответствует цели.

Следует отметить, что не существует единого для всех случаев способа (правила, алгоритма) выделения существенных свойств объекта. В некоторых случаях приходится построить несколько разных моделей с различными наборами свойств, прежде чем будет достиг-

нута цель моделирования. От правильности и полноты выделения существенных свойств моделируемого объекта зависит соответствие построенной модели заданной цели, т.е. её адекватность цели моделирования.

Формализация – это процесс построения информационной модели с помощью формальных языков. Результатом формализации является информационная модель. Информационные модели будут рассмотрены в следующих параграфах этой главы.

Модели играют весьма важную роль в проектировании и создании различных технических устройств, машин и механизмов, зданий, электрических цепей и т.д. Без предварительного создания чертежа невозможно изготовить даже простую деталь.

В процессе проектирования зданий и иных сооружений кроме чертежей часто изготавливают макеты. В процессе разработки летательных аппаратов поведение их моделей в воздушных потоках исследуют в аэродинамической трубе. Разработка электрической схемы обязательно предшествует созданию электрических цепей и так далее.

Развитие науки невозможно без создания теоретических моделей – теорий, законов, гипотез и пр., отражающих строение, свойства и поведение реальных объектов. Создание новых теоретических моделей иногда коренным образом меняет представление человечества об окружающем мире (гелиоцентрическая система мира Коперника, модель атома Резерфорда, модель расширяющейся Вселенной и пр.).

Контрольные вопросы

1. Что такое модель?
2. Для чего создаются модели?
3. Приведите примеры известных моделей.
4. Может ли один и тот же объект иметь различные модели? Почему?
5. Может ли одна модель описывать разные реальные объекты? Почему?
6. Какие свойства моделируемого объекта можно считать существенными?
7. Какие виды моделей вы знаете?

8. На какие шаги можно разделить процесс создания модели?
9. Что такое формализация?

§13.2. Численные методы решения задач

Численные методы являются весьма популярной и значимой областью современной математики. Эта отрасль математической науки связана с практическими вычислениями на компьютере, позволяющими находить решение сложнейших задач с заданной степенью точности. Математически аппарат объединённый с вычислительными возможностями компьютера позволяет моделировать сложнейшие физические, химические, технические, биологические и многие другие процессы.

Если возникает необходимость исследовать какой-либо сложный физический или биологический процесс, например, законы кровообращения или движения большого количества взаимодействующих тел. Эти процессы описываются математическими уравнениями, численные методы предлагают алгоритмы их решения с заданной точностью, а уже по этим алгоритмам разрабатываются программы, с помощью которых на компьютере вычисляются искомые решения.

Подчеркнём, что возникающие при решении практических задач уравнения (алгебраические уравнения высоких степеней, дифференциальные уравнения, уравнения в частных производных и т.д.) достаточно часто решить методами чистой математики путём преобразований нельзя. Большинство реальных процессов описывается сложными нелинейными уравнениями, решить которые очень трудно или невозможно в принципе. В этой ситуации применяются численные методы, позволяющие найти приближённое решение уравнения с заданной точностью.

Только в исключительных случаях математические уравнения, описывающие реальные процессы, можно решить в явном виде.

В широком смысле под численными методами можно понимать интерпретацию математической модели на формальном компьютерном языке (в виде программы). Например, если математическая модель представлена в виде дифференциального уравнения, то численные методы позволяют получить разностное уравнение, приближающее

исходное дифференциальное с заданной точностью. Чтобы использовать компьютер для реализации численных методов, необходимо разработать соответствующую программу, способную производить математические вычисления (просчитывать разностную схему).

Итак, практическое применение численных методов состоит из трёх этапов:

- Построение математической модели (уравнения, описывающего изучаемый процесс);
- Разработка метода (вычислительного алгоритма) решения этого уравнения;
- Разработка компьютерной программы, реализующей данный алгоритм.

Метод или алгоритм решения математического уравнения (системы уравнений) называют дискретной моделью. Одной из разновидностей дискретной модели является разностная схема – разрабатываемый в рамках численных методов способ решения задачи.

Одной математической модели (системе уравнений) можно сопоставить несколько различных дискретных моделей (численных методов её решения). Однако, не все такие дискретные модели подходят для практического использования. Вычислительные алгоритмы должны удовлетворять определённым требованиям. Можно выделить два основных требования к вычислительным алгоритмам (дискретной модели):

- Соответствие исходной задаче;
- Эффективная компьютерная реализуемость.

Как уже говорилось выше, на языке математики формальные модели записываются с помощью уравнений, точное решение которых основывается на поиске равносильных преобразований математических выражений, позволяющих выразить искомую величину с помощью формулы. Точные решения существуют только для некоторых уравнений определенного вида, например, для линейных, квадратных, некоторых тригонометрических и др. Поэтому для практического решения большинства уравнений необходимо использовать численные методы с заданной точностью.

В качестве примера такого метода рассмотрим итерационный метод «деления пополам».

Предположим, что уже построена математическая модель, представляющая собой некоторое алгебраическое уравнение $f(x) = 0$ и задана точность d , с которой нужно найти его корень, тогда алгоритм решения методом «деления пополам» будет следующим:

1. Необходимо убедиться, что данное уравнение $f(x) = 0$ отвечает некоторым условиям: известен отрезок $[a; b]$ на котором уравнение имеет корень (хотя бы один) и на концах этого отрезка функция принимает значения разных знаков $f(a) \cdot f(b) < 0$. Если хотя бы одно условие не выполнено, метод «деления пополам» не применим.

2. Отрезок $[a; b]$ делится пополам некоторой точкой $c = (a + b)/2$. В результате исходный отрезок разбивается на два отрезка $[a; c]$ и $[c; b]$. Очевидно, что искомый корень находится на одном из этих отрезков.

3. Для выбора отрезка, содержащего корень, вычисляется знак функции $f(x)$ в точках a , b и c . Затем выбирается тот отрезок, на концах которого функция принимает разные знаки. Найденный таким образом новый отрезок отвечает всем условиям пункта 1 и переобозначив точки можно считать его отрезком $[a; b]$.

4. Длина выбранного в предыдущем пункте отрезка сравнивается с удвоенной заданной точностью $2 \cdot d$. Если длина отрезка больше $2 \cdot d$, то алгоритм повторяется с пункта 2, если – меньше, то вычисляется корень как его середина $x_0 = (a + b)/2$.

Таким образом, принцип метода «деления пополам» состоит в последовательном уменьшении первоначального отрезка $[a; b]$, на котором существует корень уравнения, до длины, зависящей от заданной точности. Процесс сводится к последовательному делению отрезков пополам точкой $c = (a + b)/2$ и отбрасыванию той половины первоначального отрезка $[a; c]$ или $[c; b]$, на котором корня нет. Выбор нужной половины отрезка основывается на проверке знаков функции на его концах. Выбирается та половина, на концах которой произведение значений функции отрицательно, т.е. где функция пересекает ось абсцисс.

Этот итерационный процесс продолжается до тех пор, пока длина отрезка не станет меньше удвоенной точности. Деление этого отрезка пополам дает значение корня $x_0 = (a + b)/2$ с заданной точностью.

Контрольные вопросы

1. Что представляют собой численные методы?
2. Как поступают, если найти точное решение какого-либо уравнения нельзя?
3. Приведите примеры уравнений, точное решение которых найти принципиально невозможно.
4. Из каких этапов состоит практическое применение численных методов?
5. Что такое дискретная модель?
6. Что такое итерация?
7. Опишите суть метода «деления пополам».

§13.3. Построение имитационной модели

Имитационное моделирование основано на воспроизведении с помощью компьютера процесса функционирования какой-либо системы на протяжении некоторого времени с учетом её взаимодействия с внешней средой. При имитационном моделировании реализация модели осуществляется компьютерной программой, сохраняющей логическую структуру имитируемой системы. Это позволяет по исходным данным получить сведения о состояниях системы в заданные моменты времени и даёт возможность оценить её характеристики.

Очевидно, что поскольку имитационная модель выполняется на компьютере, она является частным случаем информационной модели.

Имитационное моделирование позволяет осуществлять многократные испытания модели с нужными входными данными, чтобы определить их влияние на выходные значения. При таком моделировании компьютер используется для численной оценки модели, а с помощью полученных данных рассчитываются ее реальные характеристики.

Имитационная модель – это логико-математическое описание реального объекта, которое может быть использовано для экспериментирования на компьютере в целях проектирования, анализа и оценки функционирования данного объекта.

Имитационное моделирование - это метод исследования, при котором изучаемая система заменяется моделью с достаточной точностью описывающей реальную систему. С имитационной моделью проводятся эксперименты с целью получения информации о реальной системе.

Имитационное моделирование изучает математические модели в виде алгоритмов, воспроизводящих функционирование реальной исследуемой системы путем последовательного выполнения большого количества элементарных операций. Для получения результата компьютер выполняет программу, реализующую данную имитационную модель, т.е. проводит виртуальный эксперимент, близкий к реальности.

В отличие от имитационного моделирования, в аналитических моделях необходимо находить решение (возможно приближённое) некоторого уравнения или системы уравнений. Имитационные модели неспособны формировать свое собственное решение в том виде, в каком это имеет место в аналитических моделях, а могут лишь служить в качестве средства для анализа поведения реальной системы в определяемых экспериментатором условиях.

Таким образом, имитационная модель – это универсальное средство исследования сложных систем, представляющее собой логико-алгоритмическое описание поведения отдельных элементов системы и правил их взаимодействия, отображающих последовательность событий, возникающих в моделируемой системе.

Имитационное моделирование может применяться в самых различных сферах деятельности человека:

- Проектирование и анализ производственных систем;
- Оценка различных систем вооружений ;
- Определение требований к оборудованию и протоколам сетей связи;

- Проектирование и анализ работы транспортных систем, например: аэропортов, автомагистралей, портов и метрополитена;
- Оценка проектов создания различных организаций массового обслуживания, например: центров обработки заказов, заведений быстрого питания, больниц, отделений связи;
- Модернизация различных процессов в деловой сфере;
- Анализ финансовых и экономических систем;
- При подготовке специалистов и освоении новой техники на имитаторах (тренажёрах).

Наиболее широкое применение имитационное моделирование получило при исследовании систем с дискретным характером функционирования, например, различных систем вооружения.

К достоинствам метода имитационного моделирования можно отнести следующие особенности:

- Большинство сложных реальных систем с вероятностными параметрами нельзя точно описать с использованием математических моделей;
- Имитационное моделирование обычно гораздо дешевле, чем проведение экспериментов с реальными системами (в ряде случаев эксперимент с реальной системой не возможен);
- Имитационное моделирование позволяет изучить длительный интервал времени функционирования системы в сжатые сроки или, наоборот, изучить более подробно работу системы в развернутый интервал времени.

Процесс создания имитационной модели можно разделить на следующие пять шагов:

1. Формулировка цели моделирования;
2. Построение концептуальной модели;
3. Выбор среды моделирования и разработка программы;
4. Выполнение эксперимента;
5. Обработка, анализ и интерпретация данных эксперимента.

В качестве примера рассмотрим имитационное моделирование процесса стрельбы из артиллерийского орудия.

1. Формулировка цели моделирования: требуется с использованием имитационной модели определить угол α наклона ствола орудия к горизонту, дающий наибольшую дальность полёта снаряда.

2. Построение концептуальной модели: Опираясь на уравнения равноускоренного движения, изученные в курсе физики, можно вывести формулу, выражающую зависимость дальности полёта снаряда от угла наклона ствола орудия

$$S = v^2 \cdot \sin(2 \cdot \alpha) / g$$

где S – дальность полёта снаряда, v – начальная скорость полёта снаряда, g – ускорение свободного падения. (самостоятельно выведите эту формулу).

3. Выбор среды моделирования и разработка программы: Имитационная модель будет строиться в среде табличного процессора Excel. Для работы модели необходимо создать следующую таблицу: в столбце А в ячейках от А1 до А91 расположены значения угла α от 0° до 90° с шагом 1° ; в столбце В в ячейках от В1 до В91 расположена формула для вычисления дальности полёта снаряда с соответствующим углом наклона ствола орудия (угол берётся в качестве ссылки на ячейку слева); значение начальной скорости v можно поместить в ячейку D1 и в формуле использовать ссылку на эту ячейку; значение $g = 9,8 \text{ м/с}^2$ можно помещать в качестве константы в каждую формулу. Для создания этой таблицы вам могут понадобиться материалы главы 8.

4. Выполнение эксперимента: На этом этапе вносятся различные данные и с каждым набором данных проводится эксперимент. В ячейку D1 вносится значение скорости, например, 600 м/с, 650 м/с, 700 м/с и т.д. После внесения каждого значения вычисления выполняются автоматически и в столбце В будет вычислена дальность полёта снаряда в зависимости от угла α наклона ствола орудия, находящегося в той же строке в ячейке слева. Обратите внимание, что данные в таблицу следует вносить без наименований.

5. Обработка, анализ и интерпретация данных эксперимента: После введения каждого значения начальной скорости полёта снаряда необходимо определить угол наклона ствола орудия, соответствующий максимальной дальности полёта снаряда. Для отыскания это-

го значения удобно упорядочить строки таблицы по убыванию по значениям столбца В. После этого в первой строке окажется значение угла, соответствующее максимальной дальности полёта снаряда (должно получиться значение $\alpha = 45^\circ$). Для выполнения операций этого этапа вам может понадобится материал главы 9.

Описанная выше имитационная модель позволяет сделать вывод, что максимальная дальность полёта снаряда достигается при угле наклона ствола орудия 45° не зависимо от начальной скорости снаряда.

Контрольные вопросы

1. Что представляет собой имитационная модель?
2. На чём основано имитационное моделирование?
3. К какому виду моделирования относится имитационная модель? Почему?
4. В чём разница между имитационным и аналитическим моделированием?
5. В каких сферах деятельности человека применяется имитационное моделирование?
6. Почему наиболее широкое применение имитационное моделирование получило при исследовании систем с дискретным характером функционирования?
7. Какие достоинства имеет метод имитационного моделирования?
8. На какие шаги можно разделить процесс создания имитационной модели?
9. Расскажите как работает имитационная модель стрельбы из артиллерийского орудия.

Глава 14

Управление программами невизуального доступа к информации

§14.1. Настройки основных параметров

Программы невизуального доступа к информации как правило имеют несколько способов настройки параметров своей работы. Некоторые параметры можно настраивать с помощью клавиатурных комбинаций. Приведём на примере программы JAWS for Windows наиболее часто используемые среди них:

- Ins +V – открывает диалог «Быстрые настройки», в котором можно настроить большое количество параметров работы активного в настоящий момент приложения;
- Ins +Ctrl +S – открывает диалог «Выбор голосового профиля», с помощью которого для активного в настоящий момент приложения можно установить любой из созданных ранее голосовых профилей (установить синтезатор, темп речи и другие параметры);
- Alt +Ctrl +PgUp – увеличивает темп речи синтезатора для активного в настоящий момент приложения;
- Alt +Ctrl +PgDn – уменьшает темп речи синтезатора для активного в настоящий момент приложения;
- Ins +2 (цифра 2 вводится клавишей на верхнем ряду основной клавиатуры) – циклически переключает озвучивание ввода информации с клавиатуры («ничего», «символы», «слова», «Символы и слова»);
- Ins +Пробел, затем F11 (двухтактная команда) – выключает экран компьютера (для включения экрана следует повторить команду).

Заметим, что все команды кроме последней изменяют параметры работы JAWS только для активного приложения, т.е. при выходе из него эти настройки будут сброшены.

Команд, аналогичных приведённым, достаточно много и изучать их удобно по мере необходимости опираясь на документацию программы невизуального доступа к информации.

Более широкие возможности по управлению параметрами работы программы JAWS for Windows дают диспетчеры. Диспетчеры – это отдельные программы-утилиты, каждая из которых ориентирована на управление какой-либо группой настроек JAWS.

Приведем список наиболее часто используемых диспетчеров, и команд их вызова:

- Центр настроек – Ins +6;
- Диспетчер клавиатуры – Ins +8;
- Диспетчер фреймов – Ins +9;
- Диспетчер скриптов – Ins +0;
- Диспетчер словаря – Ins +D;
- Маркировщик графики – Ins +G.

Обратите внимание, что цифры, указанные в командах вызова диспетчеров, набираются на верхнем (цифровом) ряду основной клавиатуры. Список всех диспетчеров JAWS можно вызвать используя клавиатурную команду Ins +F2.

Все параметры, настраиваемые с помощью диспетчеров, хранятся в соответствующих файлах. Расширение файла указывает на то, к какому диспетчеру относятся эти параметры. А имя файла с параметрами совпадает с именем файла прикладной программы, для которой предназначены эти параметры. Так например, параметры JAWS для текстового процессора Microsoft Word хранятся в файлах с именами «Winword», поскольку именно такое имя имеет основной файл программы Word.

Среди файлов с параметрами JAWS встречаются файлы с именами «default». Эти файлы содержат общие параметры для всех прикладных программ.

При загрузке какой-либо программы JAWS отыскивает параметры, определяющие её работу, по следующему алгоритму:

1. В папке с персональными настройками отыскивается файл с именем программы, для которой требуется определить настройки и в нём отыскиваются необходимые параметры работы JAWS.

2. Если файл (или параметры в нём) не найден, то ищется файл для этой программы в папке с общими настройками.

3. Если файл (или параметры) опять не найден, просматривается файл с именем «default» в папке с персональными настройками.

4. Если параметр ещё не определён, просматривается файл с именем «default» из папки с общими настройками.

Файлы с настройками JAWS имеют специальный формат, и не рекомендуется изменять их содержимое без специальных утилит, отвечающих за управление данными параметрами.

Коротко познакомимся с несколькими наиболее часто используемыми диспетчерами JAWS.

Центр настроек – это наиболее часто используемый диспетчер. Вызывается он клавиатурной командой `Ins +6`. В диспетчер будет загружен файл настроек активной в момент запуска диспетчера прикладной программы. Перечень доступных в этом диспетчере настроек зависит от того, какая именно прикладная программа была активна. Для загрузки файла по умолчанию (для всех приложений), служит команда `Ctrl +Shift +D`, которую следует вводить после загрузки центра настроек.

Все настройки в этом диспетчере сгруппированы в структуру «дерево». Перемещаться по нему следует стрелками вертикального управления курсором, а раскрывать «ветви» стрелкой вправо. Соответственно закрываются «ветви» стрелкой влево.

Коротко опишем некоторые полезные настройки файла «для всех приложений» (по умолчанию):

- Пользовательские – управляет озвучиванием ввода и озвучиванием информации с экрана;
- Web, HTML, PDF – управляет чтением Интернет-страниц и PDF-документов;
- Режим форм – управляет режимом ввода текста на WEB-страницах (в формах);
- Обработка текста – управляет чтением обычного текста (например, чтением больших букв, произнесением чисел и дат и т.д.);
- Многословность речи – управляет уровнем информативности сообщений JAWS;

- Схемы речи и звуков – схема – это набор правил, по которым JAWS озвучивает элементы управления, атрибуты, шрифтовые характеристики и т.д.;

- Непрерывное чтение – управляет расстановкой пауз, оповещением о пустых строках, указанием больших букв при непрерывном чтении текста;

- Графика и символы – управляет чтением графических объектов и специальных символов;

- Брайль – управляет отображением информации на брайлевском дисплее;

- Фокус ввода и курсор – управляет взаимодействием JAWS с различными указателями (например, с указателем мыши);

- Синтезатор – синтезаторы русской речи управляются не здесь, а в соответствующем диалоге основного окна JAWS;

- Пунктуация – задает названия знаков (например, JAWS может произносить: «левая круглая скобка», а может: «круглая скобка открывается»);

- Специальные голоса – управляет голосовыми характеристиками для чтения различных элементов (например, можно настроить чтение курсивного текста более высоким по тембру голосом);

- Классы окон – позволяет назначить метки окнам (для квалифицированных пользователей);

- Клавиатура – управляет различными параметрами клавиатуры (например, позволяет задать метки клавиш, т.е. текст, который JAWS будет произносить при нажатии той или иной клавиши);

- Пользовательские цвета – управляет реакцией JAWS на цветовые выделения (например, на выделение цветом активного пункта меню. Для квалифицированных пользователей);

- Изучить – настраивает службы поиска (для квалифицированных пользователей);

- Анализатор текста – управляет особой функцией JAWS, которая позволяет обнаруживать в тексте непарные скобки, изменения шрифтовых характеристик или форматирования и т.д.;

- Система распознавания текста – управляет встроенной в JAWS оптической системой распознавания текста (OCR);

- Жесты – управляет параметрами работы с сенсорным экраном;
- Эхо мыши – управляет озвучиванием движения мыши (по умолчанию выключено);
- Разное – здесь включается «виртуальное меню в лентах», о котором говорилось в главе 7;
- Последние измененные настройки – здесь отображаются 25 последних настроек, что позволяет их легко отменить.

Обратите внимание, что закрывается центр настроек клавишей Escape.

Достаточно часто пользователи, работающие с брайлевским дисплеем, изменяют некоторые его настройки, установленные по умолчанию. В качестве примера приведём алгоритм изменения параметров работы брайлевского дисплея с помощью центра настроек JAWS:

1. Запустите центр настроек командой Ins +6.
2. Убедитесь, что открыт файл с настройками по умолчанию с помощью команды Ins +T) читать заголовок окна). Если загружен другой файл, введите команду Ctrl +Shift +D для загрузки файла с настройками по умолчанию.
3. В открывшемся окне в структуре «дерево» двигаясь стрелкой вниз найдите ветвь «Брайль» и раскройте её стрелкой вправо. В ней найдите ветвь следующего уровня «Общие» и так же раскройте.
4. Двигаясь стрелкой вниз найдите пункт «Отображать текст в режиме восьмиточечного брайля» и клавишей пробел снимите эту настройку.
5. Далее продолжая двигаться стрелкой вниз найдите ветвь «Дополнительно» и раскройте её стрелкой вправо.
6. Первой настройкой в этой ветви будет «Расположение статусных ячеек». Клавишей пробел измените её на «нет». Это позволит увеличить количество отображаемых на брайлевской строке символов.
7. Завершите настройку клавишей Enter.

В процессе работы у вас могут сформироваться свои предпочтения настроек брайлевского дисплея. В этом случае взяв приведенный алгоритм за образец, выполните их с помощью центра настроек.

Контрольные вопросы

1. Какие способы настройки параметров программы JAWS for Windows вы знаете?
2. Что такое «быстрые настройки» программы JAWS?
3. Как вводятся двухтактные команды JAWS?
4. Что такое «голосовой профиль»?
5. Для чего служат диспетчеры JAWS?
6. Какие диспетчеры JAWS вы знаете? Какой командой они вызываются?
7. Как можно вызвать список всех диспетчеров JAWS?
8. Какие параметры работы JAWS содержат файлы с именем default?
9. Какие параметры работы JAWS позволяет изменять центр настроек?
10. В каком диспетчере программы JAWS находятся настройки брайлевого дисплея?
11. Расскажите как отключить седьмую и восьмую точки на брайлевском дисплее.
12. Как вы думаете, почему в описанном алгоритме настройки брайлевого дисплея требуется выполнять изменения параметров в файле по умолчанию Default.

§14.2. Словарь подстановок

При чтении некоторых слов синтезатором речи их звучание может быть не достаточно внятным, тогда это слово можно заменить на другое по написанию, звучание которого будет отвечать требованиям пользователя. Например, сокращение «изд-во» можно заменить на полное слово «издательство». Для реализации такой возможности программы невизуального доступа к информации имеют специальные словари подстановок.

В программе невизуального доступа JAWS for Windows эту функцию выполняет диспетчер словаря. Диспетчер словаря используется в том случае, когда необходимо изменить произношение слова, фразы, аббревиатуры или отдельного символа. Необходимость такой замены может возникнуть, если синтезатор речи делает ошиб-

ки в произношении или пользователю для удобства работы необходимо назначить расшифровку аббревиатур.

При добавлении записи в словарь можно выбрать, сохранить ли эту запись в словаре активной в данный момент прикладной программы или в словаре для всех программ (по умолчанию). Если новая словарная запись сохраняется в словаре активной программы, то JAWS будет использовать её для замены слова только во время работы данной программы. Если запись сохраняется в словаре по умолчанию, то JAWS будет осуществлять замену слова в соответствии с этой записью при работе с любой прикладной программой.

Приведем алгоритм добавления словарной статьи в словарь JAWS:

1. Запустите программу, для которой необходимо создать словарную статью, например, редактор Word.

2. Поместите курсор на слово, которое необходимо добавить в словарь. Если заменяемого слова в тексте нет или поместить на него курсор неудобно, то его можно будет в дальнейшем ввести с клавиатуры.

3. Вызовете диспетчер словаря командой **Ins + D**.

4. После запуска диспетчера сразу откроется диалог, в котором фокус уже установлен на кнопке «Добавить». Нажмите ее используя клавишу пробел.

5. Откроется следующий диалог «добавление словарного определения», курсор будет находиться в поле редактирования «Действительное слово», причем там уже будет то слово, на котором был курсор в момент запуска диспетчера. При необходимости слово можно изменить или ввести другое.

6. Перейдите с помощью клавиши **Tab** в следующее поле редактирования «Замещающее слово», куда следует ввести слово или выражение, которым будет замещаться при чтении исходное слово.

7. Нажмите клавишу **Enter** для завершения операции. После чего диалог добавления словарной записи закроется и фокус окажется в основном окне Диспетчера словаря.

8. Сохраните сделанные изменения в словаре, нажав в окне Диспетчера клавиатурную команду **Ctrl + S** или выбрав в меню «Файл» команду «Сохранить».

9. Закройте окно диспетчера словаря командой Alt +F4.

По умолчанию Диспетчер словаря открывает словарь активной в момент запуска программы. Если необходимо занести словарную запись в словарь по умолчанию, то нужно после загрузки Диспетчера ввести клавиатурную команду Ctrl +Shift +D.

Словарь JAWS позволяет осуществлять замену слова на звук. Есть так же возможность указывать только корень заменяемого слова и правила его чтения, а все словоформы будут обрабатываться автоматически. Есть и еще некоторые возможности, которые здесь описываться не будут. При накоплении определенного опыта вы сможете освоить их самостоятельно используя справочную информацию JAWS.

Можно дать следующие общие рекомендации по работе с диспетчерами JAWS:

1. Чтобы задать параметры для работы с конкретной программой, а не для работы со всеми программами, вызывайте нужный диспетчер в тот момент, когда активной является данная программа.

2. Если необходимо внести изменения в файл, используемый по умолчанию (default), так, чтобы эти настройки срабатывали при работе со всеми прикладными программами, то следует в окне диспетчера ввести команду Ctrl +Shift +D.

3. После внесения изменений в параметры, файл необходимо сохранить. Если файла с параметрами для данной программы ещё нет, то диспетчер предложит его создать. В этом случае, не изменяйте предлагаемое имя создаваемого файла.

Некоторые параметры интуитивно понятны и не вызывают затруднений при выборе их значений, другие могут представлять определённые сложности. Для разъяснения назначения конкретного параметра обратитесь к справочной системе JAWS или задайте вопрос специалисту в этой области.

Контрольные вопросы

1. Зачем нужен словарь подстановок в программе невизуального доступа к информации?

2. Какой диспетчер JAWS выполняет функции словаря подстановок?

3. С помощью какой команды можно вызвать диспетчер словаря JAWS?

4. Как создать словарную статью в диспетчере словаря JAWS:

А) Для конкретной программы;

Б) Для всех программ.

5. Расскажите как можно добавить в словарь JAWS новую статью.

6. Как вы думаете, почему в приведённом алгоритме добавления словарной статьи в некоторых пунктах употребляется термин «фокус», а в других – «курсор»?

7. Расскажите, что ещё позволяет делать диспетчер словаря JAWS.

8. Какие общие рекомендации по работе с диспетчерами JAWS вы знаете?

§14.3. Диспетчер скриптов JAWS for Windows

При работе на персональном компьютере пользователь постоянно выполняет определенные действия, например, вводит клавиатурные команды, устанавливает «флажки» в диалогах, вводит текст в поля редактирования и т.д. Наборы таких часто используемых операций можно объединять в скрипты (или макросы).

Скрипты – это последовательности отдельных операций, которые можно использовать для управления широким диапазоном компьютерных процессов, например, вставка подписи под письмом с помощью одного нажатия клавиш или озвучивания определенной области экрана при появлении там текста и т.д.

Программа невизуального доступа к информации JAWS использует много различных скриптов, которые определяют ее поведение в той или иной ситуации. Например, в главном файле скриптов есть скрипт, Привязанный к клавише Стрелка вниз и этот скрипт сообщает JAWS, что необходимо прочитать новую строку. Этот конкретный скрипт носит название SayNextLine и активизируется при нажатии стрелки вниз. Он довольно сложен, поскольку должен проанализировать окно какого типа в настоящий момент активно и если это текстовый редактор, то озвучить строку, на которую переместился курсор при нажатии стрелки вниз. В этом файле скриптов есть функция SayLine(), она вызывается после того, как курсор перемещается

на следующую строку. Именно эта функция скрипта фактически выполняет чтение строки.

Функция – это скрипт, который не закреплен за конкретной клавишей и который при вызове возвращает определенное значение.

JAWS всегда находится под управлением скриптов. Файлы скриптов JAWS – это наборы отдельных скриптов, которые загружаются для работы с конкретной программой. Эти файлы автоматически загружаются каждый раз при запуске какой-либо прикладной программы.

Есть два типа файлов скриптов – по умолчанию (Default) и для прикладных программ. Файл скриптов по умолчанию загружается при запуске JAWS и является активным во всех сессиях. Файл скриптов для прикладной программы размещается поверх файла скриптов по умолчанию при загрузке данной программы. JAWS знает какой файл скриптов прикладной программы загружать, поскольку имя файла скриптов для неё (без расширения) совпадает с именем самой программы. Например, файл скриптов по умолчанию называется default.jss (текст скрипта), а программа notepad.exe будет иметь файл скриптов notepad.jss.

Когда пользователь закрывает прикладную программу, то файл её скриптов будет выгружен и активными станут все настройки по умолчанию, пока не будет загружен другой файл скриптов.

В программе невидимого доступа к информации JAWS for Windows существует возможность писать собственные скрипты для конкретных целей, а также возможность модифицировать отдельные скрипты, поставляемые вместе с программой JAWS.

Для модификации имеющихся или создания собственных скриптов следует использовать диспетчер скриптов JAWS. Вызвать его можно клавиатурной командой `Ins +0`. Команды диспетчера скриптов организованы в виде классического меню. Для входа в меню следует использовать клавишу `Alt`. В окне диспетчера будет загружен файл скриптов активной в момент его запуска прикладной программы. Если активная прикладная программа не имеет скриптов, то будет загружен файл по умолчанию Default.jss.

В диспетчере скриптов доступны следующие клавиатурные команды:

- Ctrl +N – создать новый файл;
- Ctrl +O – открыть файл;
- Ctrl +Shift +D – открыть файл по умолчанию (Default.jss);
- Ctrl +S – скомпилировать и сохранить файл скриптов;
- Ctrl +W – сохранить файл скриптов без компиляции;
- Ctrl +E – создать скрипт;
- Ctrl +Del – удалить скрипт;
- Ctrl +I – вставить вызов функции;
- Ctrl +Shift +I – вставить выполнение скрипта;
- F2 – следующий скрипт;
- Shift +F2 – предыдущий скрипт;
- Ctrl +G – перейти на строку с заданным номером;
- Ctrl +L – вывести список скриптов;
- Alt +F4 – закрыть диспетчер скриптов.

В окне редактора диспетчера скриптов будут также доступны все клавиатурные команды редактирования текста.

Контрольные вопросы

1. Что такое скрипты?
2. Какую роль выполняют скрипты в программе JAWS for Windows?
3. Какая разница между скриптом и функцией?
4. Сколько скриптов может содержать файл скриптов (файл с расширением jss)?
5. Какие виды файлов скриптов вы знаете?
6. Что происходит с файлом скриптов прикладной программы при её закрытии?
7. Для чего нужен диспетчер скриптов JAWS?
8. Какие клавиатурные команды диспетчера скриптов вы знаете?

§14.4. Разработка пользовательских скриптов

Рассмотрим приемы работы с диспетчером скриптов JAWS for Windows на примере создания скрипта вставки адреса электронной почты в любой прикладной программе:

1. Запустите диспетчер скриптов командой `Ins +0`.
2. Чтобы возможность вставки адреса электронной почты была в любой прикладной программе, загрузите с помощью команды `Ctrl +Shift +D` файл скриптов по умолчанию. После этого курсор окажется в окне редактирования, где уже будет присутствовать текст созданных разработчиками JAWS скриптов.
3. В окне редактирования перейдите в конец текста командой `Ctrl +End`.
4. Откройте диалог создания скрипта командой `Ctrl +E`.
5. В первом поле открывшегося диалога «Новый скрипт» введите имя создаваемого скрипта, например, «Email» (имя выбирается произвольно).
6. Перейдите в следующее поле клавишей `Tab` и установите в нём клавишей пробел флажок, разрешающий назначение клавиатурной команды на выполнение этого скрипта.
7. В следующем поле редактирования введите краткое описание скрипта, например, «Адрес электронной почты» (текст описания выбирается произвольно). Эта информация будет озвучиваться при включенном режиме клавиатурной помощи, который включается командой `Ins +1`.
8. В следующем поле вводится более подробное описание скрипта, которое будет озвучено при быстро дважды выполненном нажатии команды вызова скрипта в режиме клавиатурной помощи. Введите в нём, например, «Вставка адреса моей электронной почты в любой прикладной программе».
9. В следующем комбинированном редакторе можно ничего не выбирать, а перейти клавишей `Tab` далее.
10. В этом поле необходимо нажать комбинацию клавиш, которая будет вызывать данный скрипт. Нажмите комбинацию `Ctrl +Alt +1` и перейдите клавишей `Tab` на кнопку «Ok».
11. Клавишей пробел активируйте кнопку «Ok», диалоговое окно закроется и курсор окажется в тексте будущего скрипта между строками «Script Email ()» и «EndScript».

12. Здесь следует вписать текст скрипта на языке скриптов JAWS. Мы впишем всего одну команду `TypeString("onyx@google.com")` (адрес электронной почты вписывается нужный вам для работы).

13. Теперь следует скомпилировать (т.е. превратить в исполняемую программу) и сохранить созданный скрипт с помощью команды `Ctrl + S`. Если диспетчер скриптов сообщает об ошибке, проверьте правильность написания единственной строки скрипта. Обратите внимание, что в конце строки с именем функции (после круглой скобки закрыто) точку ставить не следует.

14. Закройте диспетчер скриптов командой `Alt + F4`.

Скрипт создан. Теперь его можно использовать в любом текстовом редакторе вводя команду `Ctrl + Alt + 1`.

Контрольные вопросы

1. Какие способы запуска диспетчера скриптов JAWS вы знаете?
2. Как вы думаете, какой файл скриптов будет загружен в диспетчер, если запускать его находясь на «Рабочем столе»?
3. Как называется диалог создания скрипта в диспетчере скриптов?
4. Обычно пишут «E-Mail», однако в приведённом в параграфе алгоритме имя скрипта записано без тире «Email». Как вы думаете почему?
5. Как вы думаете, почему для вызова скрипта вставки подписи в алгоритме указана клавиатурная комбинация `Ctrl + Alt + 1`? Можно ли указать комбинацию `Ctrl + 1`?
6. Что означает термин «скомпилировать»?

Глава 15

Цифровые технологии незрительного доступа к информации

Люди с нарушением зрения при работе с информацией используют специфические технические средства, которые объединяются термином «тифлоинформационные средства», т.е. средства, позволяющие инвалидам по зрению получать, создавать, обрабатывать и передавать информацию. Самым распространенным видом тифлоинформационных средств в настоящее время является персональный компьютер, оснащенный программой незрительного доступа к информации и брайлевским дисплеем, хотя существуют и автономные цифровые тифлоинформационные средства. В этой главе вы сможете познакомиться с наиболее популярными и часто используемыми такими автономными устройствами.

§15.1. Чтение плоскочастных документов

Одним из наиболее важных этапов развития тифлоинформационных средств стало создание технологии незрительного доступа к плоскочастной информации (OCR – optical text recognition). Это событие существенно расширило круг доступной для слепых информации. Люди с глубоким нарушением зрения получили возможность чтения практически любой обычной плоскочастной книги. В настоящее время доступ к плоскочастной информации можно осуществлять как с помощью компьютера, оснащенного сканером (или камерой) и соответствующим программным обеспечением, так и с помощью автономных читающих устройств.

Читающее устройство (или, как его называли ранее, читающая машина) – это устройство для чтения плоскочастной информации с бумажного носителя, а также для доступа к информации, представленной в электронном виде, включая фотографии текста, PDF-документы без текстового слоя и т.д. Читающие устройства позволяют работать и с электронной информацией в текстовом виде, однако, для чтения такой информации существует достаточно много более

удобных портативных устройств, читающие машины в основном используют для чтения плоскопечатной информации на бумажном носителе.

До недавнего времени эти устройства были оснащены сканером, под крышку которого помещалась раскрытая книга или лист с отпечатанным текстом, и после нажатия одной кнопки печатный текст озвучивался с помощью синтезатора речи. Современные читающие устройства оснащены не сканером, а камерой на штативе, под которую следует помещать плоскопечатный текст. Камера читающей машины автоматически отслеживает момент переворачивания страницы книги под объективом, и чтение начинается через несколько секунд. Устройство просто в использовании и не требует для обучения большого времени.

Приведем список основных возможностей читающей машины:

- Автоматически сканирует и читает печатный материал на русском или иностранном языке;
- Имеет возможность выбора скорости, громкости, а также голоса чтения;
- Озвучивает каждый шаг перемещения по меню;
- Автоматически определяет момент переворачивания страницы с последующим распознаванием изображения;
- Обладает возможностью сканирования книг в переплете с сохранением разбиения на страницы;
- Сохраняет файлы на жестком диске или USB накопителе для последующей работы;
- Имеет возможность подключения к монитору для увеличения размера шрифта, изменения цвета текста и фона, добавления расстояния между буквами и подсветки слов во время визуального чтения;
- Имеет возможность Подключения брайлевского дисплея для чтения книги по брайлю;
- Способна воспроизводить аудиокниги, включая формат DAISY.

Таким образом, читающие машины предоставляют доступ к плоскопечатному тексту с помощью синтеза речи, рельефно-точечной системы Брайля и увеличенного визуального изображения. Что позволяет использовать эти устройства как totally слепым пользо-

вателям, так и имеющим остаточное зрение. Подчеркнем, что преобразовывать в речь или в брайл читающая машина может только печатный литературный текст (например, художественную литературу), рукописные тексты и математические формулы можно только увеличить, а отобразить на брайлевском дисплее или воспроизвести с помощью синтезатора речи нельзя.

Напомним, что все, описанные выше функции читающего устройства, можно реализовать на персональном компьютере с соответствующим программным обеспечением. Например, для доступа к плоскочечным текстам на бумажном носителе и к электронным графическим изображениям текстовой информации можно использовать персональный компьютер с подключённым сканером и установленными программами Fine Reader и JAWS for Windows или NVDA.

Контрольные вопросы

1. Что такое тифлоинформационные средства?
2. Какую роль сыграла разработка технологии невизуального доступа к плоскочечной информации?
3. Что такое технология OCR?
4. Что такое читающая машина?
5. Какие основные функции имеет читающая машина?
6. Что необходимо для осуществления невизуального доступа к плоскочечной информации с помощью персонального компьютера?
7. Какие ограничения имеет современная технология невизуального доступа к плоскочечной информации?

§15.2. Брайлевские принтеры

Брайлевские принтеры (Embosser) – это специальные достаточно дорогие устройства для печати рельефно-точечным шрифтом Брайля. С помощью брайлевского принтера можно распечатать учебный материал, рельефные рисунки и схемы, раздаточный материал и т.д. В брайлевском принтере имеется печатающая головка с электромагнитными молоточками, которые и «накалывают» на листе брайлевской бумаги рельефные точки брайлевских символов. Достаточно

большое количество печатающих молоточков обеспечивает высокую скорость печати. При работе брайлевского принтера создается сильный шум, поэтому часто его помещают в специальный шумопоглощающий шкаф.

В нашей стране уже более 25-ти лет используются принтеры шведской компании Index Braille. Заметим, что эти принтеры не работают со стандартными текстовыми редакторами (MS Word, Блокнот и др.). Для подготовки текстов к печати по брайлю с их помощью следует использовать специальное программное обеспечение или готовить тексты вручную без автоматизации.

Принтеры компании Index Braille получают на вход последовательность байтов, т.е. наборы из восьми нулей и единиц. Каждый такой байт изображается соответствующим набором рельефных (брайлевских) точек. Точки восьмиточечного брайлевского символа соответствуют битам переданного в принтер байта, т.е. если в определенной позиции стоит единица, то соответствующая точка воспроизводится, а если нуль, то точка не воспроизводится. Кодовая таблица соответствия байтов и изображаемых на печати рельефных символов встроена в принтер.

Таким образом, принтер напечатает не то, что отображено на экране компьютера, а набор рельефно-точечных символов, соответствующих переданным байтам. Причём расположение точек и расстояние между ними задаются в настройках самого принтера. Подготовка текстов к распечатке на таком принтере обычно требует специального программного обеспечения.

Для работы с принтером Index Braille удобно использовать прилагаемую к нему программу IbPrint. Эта программа позволит протестировать принтер, а также распечатывать на нем подготовленные с помощью других программ текстовые файлы. Установка программы не требует от пользователя никаких действий, достаточно её запустить. Установить IbPrint можно также через интернет, пройдя по ссылке www.indexbraille.com/downloads/Software.aspx.

Хотя у принтеров Index Braille существует режим печати графики, но как правило используются они для печати только текстовой информации, высококачественных рельефных изображений с их помощью

не изготовить. Принтеры Index Braille удобны для печати текстовой информации, содержащей как литературный текст, так и математические формулы. Акцент на печать рельефной графики имеют принтеры Tiger, выпускаемые американской компанией ViewPlus. Принтеры Tiger поставляются вместе с программным обеспечением того же названия. в основе работы принтеров Tiger лежит графический способ печати, как и у всех современных обычных (для плоской печати) принтеров.

Поскольку принтеры семейства Tiger основаны на графическом способе печати, то на самом принтере нет элементов управления. Вся подготовка текста или изображения происходит на компьютере, а принтер напечатает то, что отображено на экране. У принтеров Index Braille напротив есть достаточно много элементов управления на самом устройстве.

Принтеры семейства Tiger способны делать точки разной высоты в зависимости от цвета области на оригинал-макете рисунка. При подготовке графики к рельефной печати высота точек автоматически соотносится с интенсивностью цвета. Чем ярче и интенсивнее цвет, тем выше данный участок рельефного изображения.

Принтеры Tiger печатают рельефные рисунки с разрешением 20 точек на дюйм. Этого вполне достаточно для хорошего рельефного рисунка.

Обычный брайлевский текст также рассматривается принтером как графика. Т.е., брайлевский шрифт представляет собой стандартный шрифт, символы которого состоят из точек. Если текст, подготовленный для такого принтера распечатать на обычном принтере, то вместо брайлевских точек на бумаге будут обычные, плоскопечатные точки.

Верно и обратное: если на принтеры Tiger отправить текст, набранный плоским шрифтом, то на выходе получится рельефное изображение соответствующих плоскопечатных букв. Если шрифт сделать достаточно крупным, то эти буквы вполне можно воспринимать тактильно. Таким образом, незрячему можно наглядно показать рельефное изображение плоскопечатных букв, различное их начертание и виды шрифтов.

В нашей стране получил распространение принтер Tiger Emprint этой линейки. Его основным преимуществом является возможность совмещения рельефной и цветной печати. Т.е. рисунок, изготавливаемый этим принтером, является одновременно рельефным и цветным, что очень полезно для пользователей с остаточным зрением. Но использовать его для печати текстов неудобно, так как в этом принтере нет двусторонней печати, и работает он только с бумагой формата А4.

В отличие от принтеров Index Braille принтеры Tiger работают со стандартными текстовыми редакторами. Однако, для печати брайлевских текстов необходим специальный точечный шрифт и предварительная подготовка текста в редакторе.

Таким образом, можно сделать вывод, что для печати текстовой информации эффективнее использовать принтеры Index Braille, а для изготовления рельефных рисунков принтеры линейки Tiger.

Контрольные вопросы

1. Что такое брайлевский принтер?
2. Каков механизм печати брайлевских принтеров?
3. Зачем некоторые брайлевские принтеры помещают в шумопоглощающий шкаф?
4. Какие брайлевские принтеры вам известны?
5. Как вы думаете, почему принтеры Index Braille не работают со стандартными текстовыми редакторами?
6. Что такое брайлевская кодировка символов?
7. Какую функцию выполняет программа IbPrint?
8. Как вы думаете, зачем нужны элементы управления на принтере Index Braille?
9. Чем отличается принцип печати принтеров семейства Index от принципа печати принтеров семейства Tiger?
10. Как осуществляют цветопередачу принтеры семейства Tiger?
11. Что произойдёт, если передать для печати принтеру семейства Tiger обычный текст?
12. Какие особенности имеет принтер Tiger Emprint?

§15.3. Формат DAISY

Как правило, люди с глубоким нарушением зрения пользуются книгами, статьями и другими текстовыми материалами в следующих доступных для них формах представления информации:

- издания рельефно-точечным шрифтом Брайля;
- цифровая аудиозапись (MP3, LKF, DAISY и др.);
- Электронные форматы хранения текстов (TXT, RTF, DOC, DOCX, HTML, FB2, PDF и др.).

Рельефно-точечная система Брайля вам хорошо знакома, а в предыдущем параграфе вы познакомились и с принтерами, осуществляющими брайлевскую печать. Работа с электронными документами была рассмотрена в главе 7. В этом параграфе будет описываться цифровая аудио запись «говорящих» книг.

Формат MP3 хранения аудиоинформации хорошо всем известен. В этом формате существует достаточно много «говорящих» книг, музыки и другой аудиоинформации. Воспроизводить MP3-файлы можно как на обыкновенном плеере и компьютере, так и на смартфоне.

LKF-это формат, предназначенный специально для «говорящих» книг, записываемых для библиотек слепых. Формат LKF (Логос-Книга-Файл) имеет криптозащиту, которая позволяет воспроизводить LKF-книги только на специальных тифлофлэшплеерах. Это сделано в целях защиты авторских прав и возможности бесплатного распространения таких «говорящих» книг среди незрячих читателей. Доступ к библиотеке LKF-книг можно получить на сайте AV3715.ru, предварительно зарегистрировавшись в региональной библиотеке слепых и получив там пароль и логин.

Существует достаточно большой выбор как отечественных, так и импортных аппаратных плееров для воспроизведения LKF-книг или рассматриваемых иже DAISY-книг. Один из таких плееров будет рассмотрен в следующем параграфе.

Наряду с аппаратными, есть и программные плееры для воспроизведения LKF-книг. Эти плееры предназначены для мобильных устройств под управлением операционной системы Android.

Формат DAISY в нашей стране распространён значительно меньше, хотя по своим возможностям существенно превосходит формат

LKF. Слово DAISY – это акроним, расшифровывающийся как Digital Accessible Information System (Цифровая доступная информационная система). Он совмещает в себе преимущества аудиозаписи и электронного документа. Более правильно называть «говорящими» книги именно в этом формате.

Для разработки стандарта DAISY в 1996 году в США был создан международный консорциум, в который вошли представители организаций слепых, библиотек, разработчиков специального программного обеспечения и издателей из разных стран мира. Хотя на конгрессе, положившем начало консорциуму, присутствовали представители нашей страны, в итоге Россия в консорциум не вошла.

Стандарт DAISY имеет следующие особенности:

- Высокая степень сжатия аудиофайлов.
- Возможность навигации по книге, что и позволяет говорить о такой аудиозаписи как о «говорящей» книге. Предусмотрено шесть уровней навигации, т.е. работая с такой книгой можно переходить к любому разделу, главе, параграфу, абзацу, предложению и даже слову. Конкретный набор структурных элементов, по которым можно осуществлять навигацию зависит от того, как была подготовлена данная книга.
- Сочетание и синхронизация различных форм представления информации. Книга в формате DAISY может содержать как обычный текст, так и аудиодорожку и графические объекты. Например, учебник, начитанный диктором, может сопровождаться полным или частичным текстовым вариантом. Пользователь имеет возможность в любой момент выбрать наиболее удобный способ работы. Возможно слушать аудиокнигу в дикторском исполнении или читать текстовый файл при помощи синтезатора речи.
- Возможность шифрования данных согласно спецификациям поставщика книг. Это позволяет защитить DAISY-книгу от несанкционированного копирования. Например, чтобы прочитать книги DAISY, записанные Библиотекой Конгресса США, необходим ключ к системе шифрования, который может получить только читатель этой библиотеки.

- Возможность контекстного поиска. Это значит, что если возникает необходимость найти какое-либо слово в DAISY-книге, его можно набрать на клавиатуре (в том числе и на клавиатуре тифлофлэшплеера) и проигрыватель автоматически переместится в то место текста, где встретилось это слово.

- Доступность DAISY-книг в режиме OnLine. С 2010 года читатели зарубежных специализированных библиотек могут находить по каталогам и скачивать книги через сеть Интернет. Специальные библиотеки нашей страны предоставляют такую возможность только для LKF-книг.

Существуют DAISY-книги, содержащие только текст без аудиодорожки. Такие книги удобно производить автоматически. Так, например, записываются на западе газеты и журналы. Это существенно ускоряет и удешевляет процесс подготовки материала. Поскольку все современные DAISY-плееры снабжены одним или несколькими встроенными синтезаторами речи, использование таких книг не вызывает затруднений.

Способов воспроизведения книг, записанных в стандарте DAISY, достаточно для того, чтобы каждый желающий смог выбрать для себя наиболее подходящий вариант. Из аппаратных средств предлагаются настольные, довольно простые в управлении устройства и портативные плееры, которые можно носить с собой.

Для нежелающих обзаводиться дополнительными техническими устройствами существуют программные решения – DAISY-проигрыватели, которые можно установить на компьютер или смартфон. Специальные портативные аппаратные средства и программное обеспечение для широко распространенных мобильных телефонов обеспечивают автономность и мобильность. Пользоваться DAISY-книгой можно дома, в транспорте, на отдыхе и т.д.

Изготовление DAISY-книг не требует централизованного производства. Такую книгу можно сделать и в домашних условиях. Для этого будет вполне достаточно компьютера с установленным на нем специальным программным обеспечением.

На сегодняшний день DAISY активно используется практически во всем мире. В данном формате издаются учебники, справочники, художественная литература и даже периодика.

Контрольные вопросы

1. Какие формы представления текстовой информации для незрячих вы знаете?
2. Что такое «говорящая» книга?
3. В чём разница между форматами MP3 и LKF?
4. Как можно получить доступ к библиотеке LKF-книг?
5. Какие преимущества имеет формат DAISY?
6. Каковы основные особенности DAISY-книг?
7. Что такое контекстный поиск?
8. С помощью каких средств можно воспроизводить DAISY-книгу?

§15.4. Тифлофлэшплеер

В настоящее время выбор аппаратных плееров для воспроизведения «говорящих» книг в различных форматах достаточно широк. Такие устройства принято называть тифлофлэшплееры (ТФП). Основной набор функций у всех таких устройств приблизительно одинаков:

- Возможность воспроизведения «говорящих» книг в форматах LKF, DAISY, MP3 и других;
- Возможность чтения встроенным синтезатором речи электронных книг в форматах DOC, DOCX, RTF, TXT, HTML, PDF и других;
- Встроенный диктофон с возможностью записи в формате DAISY;
- Озвученное меню;
- Возможность изменения скорости воспроизведения как текстовых, так и аудио книг без изменения тембра голоса;
- Возможность подключения к сети Интернет и работы с книгами в режиме OnLine;
- Возможность прослушивания интернет-радио.

Наиболее популярными в нашей стране являются плееры:

- Тифлофлэшплееры СОЛО-1 и СОЛО-2 Российской компании Круст;

- Тифлофлэшплееры ElecGeste DTBP-101, DTBP-301 и DTBP-202 Российской компании Элекджест;
- Тифлофлэшплеер Victor Reader Stream и Victor Reader Stratus Канадской компании Humanware;
- Тифлофлэшплеер PLEXTALK Pocket (PPT) Японской компании Shinano Kenshi.

Рассмотрим подробнее плеер Victor Reader Stream компании Humanware. Эта компания одной из первых выпустила на рынок плеер с возможностью воспроизведения книг в формате DAISY. С тех пор устройство было значительно усовершенствовано и в настоящее время по праву занимает одно из лидирующих мест в рейтинге подобных устройств.

Опишем внешний вид плеера Victor Reader Stream и назначение его кнопок.

Переднюю панель плеера можно условно разделить на верхнюю и нижнюю части. В верхней части расположены 5 рядов по 3 кнопки в каждом, причем верхний ряд отделен от остальных чуть большим промежутком.

Левая верхняя кнопка «Перейти на страницу» позволяет перейти на желаемую страницу. Над этой кнопкой находится небольшое отверстие встроенного монофонического микрофона. Справа от кнопки «Перейти на страницу» находится кнопка «Онлайн», включающая и выключающая режим полёта (Wi-Fi) и переключающая между стандартной книжной полкой плеера и книжной полкой в интернете. Справа сверху над кнопкой «Онлайн» находится светодиод, который светится янтарным светом в процессе работы Wi-Fi (беспроводная сеть). Светодиод гаснет, когда режим полёта включён.

Справа от кнопки «Онлайн» находится кнопка «Закладка» для установки закладки и перехода к ранее установленной закладке. Эта кнопка также служит для переключения между большими и малыми буквами при вводе пароля.

Ниже трех описанных кнопок через небольшой промежуток располагаются еще четыре ряда кнопок. Эти кнопки представляют собой 12-кнопочную клавиатуру телефонного типа с двумя выпуклыми точками на кнопке с цифрой 5. Эта цифровая клавиатура использу-

ется для перемещения по структуре DAISY-книги, а также для ввода закладки, страницы или номеров заголовков.

Под цифровой клавиатурой проходит выпуклая горизонтальная линия, разделяющая верхнюю и нижнюю части передней панели плеера. На нижней части располагаются четыре кнопки, по расположению похожие на курсорные стрелки обычной компьютерной клавиатуры. Кнопка «Воспроизведение/Стоп» расположена внизу между кнопками треугольной формы «Перемотка назад» и «Перемотка вперед».

Над кнопкой «Воспроизведение/Стоп» находится кнопка «Сон». Однократное нажатие этой кнопки вызывает объявление времени и даты. Многократное нажатие активирует различные таймеры режима сна. По достижении установленного времени плеер автоматически выключится.

С левой стороны плеера, ближе к верхней части, находится кнопка «Питание/Переключение». Для включения плеера следует нажать и удерживать эту кнопку до характерного звукового сигнала. Для выключения устройства ее также следует нажать и удерживать до воспроизведения двойного сигнала. Под кнопкой «Питание/Переключение» находится зелёный светодиодный индикатор. Этот светодиод светится постоянно, пока плеер включён, и мигает во время подзарядки батареи.

Во время работы плеера, нажатие кнопки «Питание/Переключение» позволяет переключать режимы управления громкостью, скоростью и установкой тона/высоты голоса. Под зелёным светодиодом находятся две треугольные кнопки. Это кнопки «Вверх» и «Вниз», используемые для увеличения или уменьшения, соответственно выбранному с помощью кнопки «Питание/переключение» режиму управления громкостью, скоростью или высотой голоса. Например, во время воспроизведения кнопки «Вверх» и «Вниз» изменяют громкость, а если перед их использованием один раз кратковременно нажать кнопку «Питание/Переключение», то с помощью кнопок «Вверх» и «Вниз» можно изменять скорость воспроизведения. При каждом кратковременном нажатии кнопки «Питание/Переключение» плеер будет объявлять изменяемый параметр. Выполненные

таким образом настройки сохраняются между сеансами работы. Отдельные настройки громкости сохраняются для встроенного динамика и для наушников.

Если плеер не реагирует на нажатие любых кнопок и не выключается, удерживайте кнопку питания нажатой 7 секунд для принудительного сброса плеера.

С правой стороны плеера, ближе к верхней части, находится гнездо стереофонического микрофона, которое может быть использовано для подключения внешнего микрофона или как линейный вход. Под гнездом микрофона находится кнопка «Запись» с красным кружком и выпуклой точкой в центре.

На верхнем торце плеера находится слот SD-карты памяти, а за ним располагается динамик. Справа от SD-слота располагается разъём для наушников, который также может быть использован для подключения внешних динамиков.

В центре нижнего торца находится порт микро-USB. Под ним расположена маленькая выпуклая точка. Этот порт используется для зарядки плеера, а также для подключения к персональному компьютеру или внешнему накопителю.

Плеер распознаёт различные типы книг, которые сохранены в отдельных структурах папок, называемых книжными полками. Для выбора книжной полки следует нажимать кнопку 1 цифровой клавиатуры, после чего встроенный синтезатор речи объявит активную книжную полку. Для переключения к следующей необходимо ещё раз нажать цифру 1. Для перемещения по книжной полке (для выбора конкретной книги) используйте кнопки 4 и 6. Список книг на полке представляет собой замкнутое кольцо. За исключением книжной полки заметок, объявляться будут только не пустые книжные полки. На SD-карте каждая книжная полка содержится в зарезервированной папке с именем, начинающимся с «\$VR». Внутри каждой из таких зарезервированных папок (книжных полок) могут быть создаваемые пользователем подпапки, содержащие различные книги. В корневую папку карты памяти можно заносить и другие папки и файлы, но только содержание зарезервированных \$VR-папок определяется плеером как книжные полки. При установки в плеер не защищенной

от записи карты памяти зарезервированные \$VR-папки будут созданы автоматически.

Контрольные вопросы

1. Чем тифлофлэшплеер отличается от обычного плеера?
2. Каковы основные функции тифлофлэшплеера?
3. Какие аппаратные тифлофлэшплееры для воспроизведения DAISY-книг вы знаете?
4. Каким средством воспроизведения «говорящих» книг вы пользуетесь? Почему?
5. Что такое «книжная полка» для тифлофлэшплеера Victor Reader Stream?

§15.5. Портативный тифлокомпьютер

Тифлокомпьютеры – это особый класс специализированных устройств, на которых могут работать незрячие пользователи. Они не имеют экрана и стандартной клавиатуры для работы визуальными методами, но, благодаря этому, тифлокомпьютеры имеют небольшие (в сравнении с ноутбуком) размеры. Тифлокомпьютеры общаются с пользователем только с помощью речевого выхода и встроенного брайлевского дисплея. Сегодня на российском рынке наибольшей популярностью пользуется тифлокомпьютер ElBraille отечественной компании «Элита Групп».

Тифлокомпьютер ElBraille представляет собой комбинированное устройство, состоящее из брайлевского дисплея Focus 14 Blue (или Focus 40 Blue) и док-станции, являющейся полнофункциональным компьютером под управлением операционной системы Windows. В ElBraille используется программа невидимого доступа к информации JAWS for Windows. Очевидно, что эффективно использовать такой тифлокомпьютер можно только после особой подготовки пользователя, но удобство его эксплуатации оправдывает потраченное на учебу время.

Тифлокомпьютер оснащен всеми видами беспроводной связи, имеет встроенный микрофон, картридер, а также на нём расположе-

ны все необходимые разъёмы для подключения монитора, стандартной клавиатуры и внешних USB-устройств.

Заметим, что на нем очень удобно не только вести запись текстового и аудио материала, но и, в отличие от обычного компьютера, выполнять брайлевские записи формул и математических расчетов. Выглядит это так же, как и на брайлевской печатной машинке, но дополнительно появляется возможность редактирования. При необходимости математические расчеты можно распечатать на брайлевском принтере.

Познакомимся с внешним видом ElBraille. Для удобства эксплуатации тифлокомпьютер может быть помещен в кожаный чехол с ремнем для переноски. Извлеките устройство из чехла и расположите его так, чтобы вдоль верхнего (дальнего от вас) края корпуса была расположена надпись шрифтом Брайля “ElBraille”. Слева и справа от этой надписи находятся встроенные динамики.

Под надписью “ElBraille” размещены в одну линию 6 кнопок (слева направо): E1, E2, «Уменьшение громкости», «Увеличение громкости», E3, E4. О назначении кнопок E1 – E4 будет рассказано ниже.

Ниже (ближе к вам) под этими кнопками располагается съемный брайлевский дисплей Focus. Он вложен в док-станцию и удерживается двумя замками.

Рядом с левым нижним углом брайлевского дисплея на слегка наклонной части корпуса док-станции находится небольшая утопленная кнопка «Питание», возле неё имеется светодиодный индикатор и микрофон. Над этой наклонной частью корпуса расположена магнитная крышка, прикрывающая USB-разъём для подключения брайлевского дисплея. Без необходимости открывать ее не следует.

На правом торце устройства расположены следующие элементы (от передней панели к задней): кнопка замка, удерживающего брайлевский дисплей (второй замок расположен симметрично на левом торце корпуса), разъём для подключения гарнитуры, слот для SD-карты, USB-разъём, разъём для подключения блока питания.

На левом торце расположена кнопка замка брайлевского дисплея, за ней разъём Mini HDMI для подключения внешнего монитора и разъём для установки сим-карты.

Элементы управления, находящиеся на брайлевском дисплее Focus вам уже хорошо знакомы, поясним назначение элементов управления, находящихся на Док-станции.

Кнопки увеличения и уменьшения громкости служат для управления общей громкостью устройства.

Короткое нажатие кнопки E1 вызывает специальное меню, содержащее наиболее часто используемые приложения. Длительное нажатие E1 вызывает аварийное меню. Его вызов сопровождается характерным звуковым сигналом. В меню всего три пункта:

- Перезапуск JAWS;
- Завершение работы;
- Перезагрузка.

Назначение каждой из этих команд очевидно. Обратите внимание, что аварийное меню озвучивается самостоятельно, без использования JAWS. Таким образом, даже если JAWS по каким-либо причинам перестанет работать, оно будет озвучиваться.

Для перемещения по пунктам упомянутых выше меню следует использовать навигационные кнопки-качельки по краям брайлевской строки, а для активации пункта меню следует нажать круглую кнопку, расположенную над качелькой. В данной ситуации разницы между левой и правой качельками нет. Для выхода из меню без выбора команды следует ввести клавиатурную команду 1356 +пробел, что соответствует клавише Escape на стандартной клавиатуре.

Однократное короткое нажатие E2 позволяет узнать уровень заряда батареи. Нажав E2 быстро дважды, можно получить информацию о состоянии беспроводного подключения к сети Интернет.

Однократное короткое нажатие E3 позволяет узнать текущее время, а быстрое двойное нажатие – текущую дату.

E4 запускает специально разработанное для ElBraille приложение ElNotes, предназначенное для работы с небольшими заметками.

Заметим, что приведённые выше команды назначены по умолчанию. С помощью утилиты ElBraille «Редактор клавиатуры» можно изменить назначения этих кнопок.

Для включения ElBraille нажмите и удерживайте кнопку питания до начала воспроизведения отрывистых звуковых сигналов и ви-

брации. Через некоторое время сигналы прекратятся, а на брайлевском дисплее появится надпись “Focus 14”. Спустя еще несколько секунд будет воспроизведено приветственное сообщение JAWS for Windows. После этого ElBraille готов к работе.

Существует несколько способов выключения ElBraille. Например, можно воспользоваться следующим алгоритмом:

1. Перейдите на «Рабочий стол» свернув все окна используя двухтактную команду 48 +Пробел, а затем 1348 (латинская буква m), что соответствует клавиатурной команде Win +M на стандартной клавиатуре компьютера.

2. Далее вводится также двухтактная команда 168 +Пробел, затем 145 (буква d), что соответствует команде Alt +F4 на стандартной клавиатуре компьютера.

3. Раскроется диалог «Завершение работы Windows», в комбинированном списке которого с помощью одной из навигационных кнопок-качелек следует выбрать команду «Завершение работы». Перед использованием навигационной кнопки-качельки необходимо изменить ее режим с помощью расположенной над ней круглой кнопки на «Режим списка». Для выбора команды используйте кнопку 8, что соответствует клавише Enter стандартной клавиатуры.

При выключении устройства будут воспроизведены два коротких звуковых сигнала разной высоты и кратковременно включится вибрация.

Для выхода из диалога без выключения или перезагрузки используйте команду 1356 +Пробел, что соответствует клавише Escape на стандартной клавиатуре.

Очевидно, что используя второй пункт этого алгоритма можно завершить любую прикладную программу.

Все управление прикладным программным обеспечением осуществляется с помощью кнопок брайлевского дисплея, работающего в сочетании с программой JAWS for Windows. Никаких отличий от работы с настольным компьютером с помощью брайлевского дисплея нет.

Владея приёмами работы с брайлевским дисплеем управление тифлокомпьютером ElBraille не должно вызвать никаких затруднений.

Контрольные вопросы

1. Чем отличается тифлокомпьютер от обычного ноутбука?
2. Какие тифлокомпьютеры вы знаете?
3. Из каких элементов состоит ElBraille?
4. Как вы думаете, чем удобнее использовать ElBraille, по сравнению с обычным ноутбуком с подключённым брайлевским дисплеем?
5. Какие элементы управления расположены на верхней панели устройства?
6. Как управляется ElBraille?
7. Как включить и выключить ElBraille?
8. Как запускать программы на ElBraille?
9. Как завершать программы на ElBraille?

СОДЕРЖАНИЕ

Глава 1

Информация и информационные процессы.....	1
§1.1. Информация в современном мире.....	1
§1.2. Формы представления информации.....	8
§1.3. Различные системы счисления.....	10
§1.4. Представление чисел в компьютере	17
§1.5. Измерение объема информации.....	21

Глава 2

Кодирование информации в компьютере.....	25
§2.1. Кодирование текстовой информации	25
§2.2. Кодирование звуковой информации	28
§2.3. Кодирование графической информации.....	32
§2.4. Коды с самоисправлениями.....	36
§2.5. Минимальные коды.....	43

Глава 3

Алгебра логики.....	46
§3.1. Логические функции.....	46
§3.2. Таблицы истинности.....	51
§3.3. Законы алгебры логики	55
§3.4. Преобразование логических выражений	59

Глава 4

Теория алгоритмов	62
§4.1. Алгоритмизация процессов	62
§4.2. Основные алгоритмические конструкции	66
§4.3. Свойства алгоритмов.....	68
§4.4. Формы представления алгоритмов	72
§4.5. Машина Тьюринга	77

Глава 5

Программирование и анализ данных без визуального контроля	82
§5.1. Структуры данных.....	82
§5.2. Массивы и списки	93
§5.3. Реализация рекурсивных алгоритмов.....	99
§5.4. Работа с файлами.....	107

Глава 6

Файловые менеджеры и программы невизуального доступа к информации	114
§6.1. Операции над файлами и папками	114
§6.2. Групповые операции	125
§6.3. Шаблоны имен	129
§6.4. Свойства файлов.....	132

Глава 7.

Текстовый процессор Word и программы невизуального доступа к информации	138
§7.1. Стили	141
§7.2. Оглавление и сноски	146
§7.3. Поиск и замена	150
§7.4. Таблицы в текстовом документе	152
§7.5. Поля и макросы	157
§7.6. Режим быстрой навигации JAWS for Windows.....	163

Глава 8

Табличный процессор Excel и программы невизуального доступа к информации	167
§8.1. Форматирование таблицы	169
§8.2. Автоматический ввод данных.....	173

§8.3. Формулы и функции	176
§8.4. Абсолютные и относительные ссылки.....	180
§8.5. Управление листами	184
Глава 9	
Реляционные базы данных и программы невидимого доступа к информации	186
§9.1. Основные понятия баз данных	186
§9.2. Табличное представление реляционной базы данных.....	189
§9.3. Сортировка и поиск данных.....	194
Глава 10	
Подготовка демонстрационных визуально воспринимаемых объектов.....	199
§10.1. Диаграммы и графики	199
§10.2. Математические формулы.....	204
§10.3. Презентации	208
Глава 11	
Разработка WEB-страниц без визуального контроля.214	
§11.1. Основные теги HTML.....	214
§11.2. Создание простейших WEB-страниц.....	223
§11.3. Навигация по WEB-документу с помощью программ невидимого доступа к информации.....	227
Глава 12	
Информационно-коммуникационные технологии без визуального контроля	233
§12.1. Локальные вычислительные сети	233
§12.2. IP-адрес и маска подсети.....	239
§12.3. Поиск информации в Интернет	244
§12.4. Сервисы сети Интернет.....	248

Глава 13

Компьютерное моделирование	253
§13.1. Что такое моделирование.....	253
§13.2. Численные методы решения задач	257
§13.3. Построение имитационной модели	260

Глава 14

Управление программами невизуального доступа к информации	265
§14.1. Настройки основных параметров	265
§14.2. Словарь подстановок.....	270
§14.3. Диспетчер скриптов JAWS for Windows.....	273
§14.4. Разработка пользовательских скриптов	275

Глава 15

Цифровые технологии невизуального доступа к информации	278
§15.1. Чтение плоскпечатных документов	278
§15.2. Брайлевские принтеры	280
§15.3. Формат DAISY.....	284
§15.4. Тифлофлэшплеер	287
§15.5. Портативный тифлокомпьютер	291
СОДЕРЖАНИЕ	296

ISBN 978-5-6045213-9-7

**При реализации проекта используются средства
государственной поддержки, выделенные в качестве гранта
Фонда Президентских Грантов**



ПРИ ПОДДЕРЖКЕ
**ФОНДА
ПРЕЗИДЕНТСКИХ
ГРАНТОВ**

Распространяется бесплатно!

Редактор пособия: Соколов Владимир Вячеславович

Подписано к печати 15.11.2022 г.

Формат 60\90\8 Усл.печ. л. 37.5

Тираж 150 экз.

Заказ № 1284

Издательство: ЧУ «ЦКТВОС «АРГУС»

Типография ООО Астра Полиграфия.

Адрес: 127282, Москва, ул. Полярная, 33 Б, стр. 2