МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«ИНГУШСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Кафедра: «Информационные системы и технологии»

ЛЕКЦИИ ПО ИНФОРМАТИКЕ

Учебно-методическое пособие

Предназначено для студентов первого курса физико-математического факультета спец. «математика» и « ИСиТ» (бакалавриат) очной и заочной форм обучения.

Магас

2022

Лекции по информатике: Учеб.-метод. пособие. – 105 с.

Излагается теоретический материал по дисциплине «Информатика». В конце каждого раздела приведены вопросы для самопроверки.

Предназначено для студентов первого курса физико-математического факультета спец. «математика» и « ИСиТ» (бакалавриат) очной и заочной форм обучения.

ВВЕДЕНИЕ

Цели и задачи дисциплины

Целями учебной дисциплины «Информатика» являются формирование представлений о сущности информации и информационных процессов, развитие алгоритмического мышления, представляющего собой необходимую часть научного взгляда на мир, изучение современных информационных технологий, демонстрация возможности использования полученных знаний в различных сферах деятельности человека.

Знание основных разделов дисциплины способствует повышению эффективности учебной деятельности студентов и их будущей профессиональной деятельности, а также положительному восприятию процесса информатизации общества.

Требования к уровню освоения содержания дисциплины

В результате изучения дисциплины студенты должны:

знать:

– основные понятия и определения информатики;

– основные принципы работы современного компьютера;

– технические средства обработки информации;

– программные средства обработки информации;

– основные понятия и способы моделирования;

уметь:

– использовать основные программные средства и информационные системы;

– моделировать различные процессы на компьютере;

владеть:

– способами и методами представления информации;

– технологиями решения задач с использованием компьютера.

1. ОСНОВНЫЕ ПОНЯТИЯ ТЕОРИИ ИНФОРМАЦИИ

1.1. Основные определения

Информатика – это дисциплина, изучающая структуру и общие свойства информации, закономерности и методы еѐ создания, хранения, поиска, преобразования, передачи и применения в различных сферах человеческой деятельности.

В настоящее время большинство операций с информацией совершается с помощью ЭВМ. Поэтому сведения о компьютерах и компьютерные технологии обработки информации являются важной составной частью дисциплины «информатика».

Понятие ***информация*** точно и однозначно не определяется, хотя используется повсеместно. Оно вводится путѐм объяснения, которое опирается на интуицию, здравый смысл или бытовое применение этого термина.

В Федеральном законе Российской Федерации от 27 июля 2006г. №149-ФЗ «Об информации, информационных технологиях и о защите информации» (http://www.rg.ru/2006/07/29/informacia-dok.html) дается следующее определение этого термина: «информация — сведения (сообщения, данные) независимо от формы их представления».

Толковый словарь русского языка Ожегова приводит 2 определения слова «информация»:

1. Сведения об окружающем мире и протекающих в нем процессах, воспринимаемые человеком или специальным устройством.

2. Сообщения, осведомляющие о положении дел, о состоянии чего-нибудь. (Научно-техническая и газетная информация, средства массовой информации — печать, радио, телевидение, кино).

Для количественного определения имеющейся информации самым удобным оказалось такое: это сведения, которые уменьшают неопределенность об окружающем мире и являются объектом хранения, преобразования, передачи и использования.

Энтропия – это мера неопределѐнности наших знаний об объекте или явлении. Энтропию иногда называют антиинформацией. *Например.* Если мы интересуемся сведениями о полностью засекреченном объекте или явлении, или получили зашифрованное сообще-

ние, а ключа к расшифровке не знаем, то для нас информация о нѐм равна нулю, а энтропия – максимальна.

Знания – это осознанные и запомненные людьми свойства предметов, явлений и связей между ними, а также способов выполнения тех или иных действий для достижения нужных результатов.

Сигнал (сообщение) **–** информационный поток, который в процессе передачи информации поступает к приѐмнику.

Данные – это зарегистрированные на материальном носителе сигналы.

Одна и та же информация может передаваться с помощью разных сообщений. Например, сведения о выпуске книги могут быть переданы с помощью телевидения, устного разговора, рекламных щитов и т. д. И, наоборот, одно и то же сообщение может нести различную информацию. Пример: сообщение на китайском языке несѐт какую-то информацию только для тех, кто этот язык знает.

Сведения, факты, данные – это знания, выраженные в сигналах, сообщениях, известиях, уведомлениях и т.д.

Информационные процессы – это хранение, передача и обработка данных.

Информационная революция – это преобразование общественных отношений из-за кардинальных изменений в сфере обработки информации.

Другими словами информационная революция означает скачок в развитии общества, новый уровень использования принципиально новых методов и средств переработки информации и процессов информационного взаимодействия в обществе, что создает основу для объединения интеллектуальных способностей человечества.

В настоящее время выделяется шесть основных информационных революций в истории развития человеческого общества. Они связываются со следующими событиями:

- появление человеческой речи; изобретение
- письменности; изобретение
- книгопечатания; изобретение радио,
- телефона, телевидения;
- изобретение микропроцессорных технологий и появление персональных компьютеров.
- Создание компьютерных сетей и, в частности, глобальной компьютерной сети Интернет.

1.2. Основные свойства информации

Свойства информации можно рассматривать в трех аспектах:

- ***технический*** – это точность, надежность, скорость передачи сигналов и т. д.;
- ***семантический*** – это передача смысла текста с помощью кодов. Например, при семантической отладке программы проверяются типы переменных, входящих в выражение: если переменная А текстовая, переменная В – числовая, то выражение А/В не имеет смысла, если же А и В – числовые переменные, то это выражение становится осмысленным;
- ***прагматический*** – это насколько эффективно информация влияет на поведение объекта.

С учетом этих факторов к основным свойствам информации относятся:

- ***полнота*** – достаточность набора данных для понимания информации и принятия правильных решений или для создания новых данных на еѐ основе. О полноте информации можно говорить, если какая-либо дополнительная информация об объекте будет уже избыточна;
- ***репрезентативность*** – имеющаяся информация и способ еѐ представления позволяет сформировать адекватное отражение свойств объекта. Непременным условием репрезентативности информации является поступление похожей информации из разных источников. Конечно полного совпадения информации, поступившей из разных источников, никогда не будет. Однако самые важные характеристики объекта весь объѐм поступившей информации будет отражать правильно;
- ***адекватность*** (***достоверность***) – степень соответствия реальному состоянию дел;

- ***актуальность*** – степень соответствия текущему моменту времени;
- ***доступность* (*понятность*)** –возможности получить нужную информацию и способ ее представления должен быть понятен еѐ получателю;
- ***ценность*** – степень важности для решения текущей задачи или дальнейшего применения в каких-либо видах деятельности человека.

1.3. Классификация информации

Поскольку носителями информации являются сигналы, то в качестве последних могут использоваться физические или социальные процессы различной природы. Например, процесс протекания электрического тока в цепи, процесс механического перемещения тела, количество людей на предприятии, имеющих высшее образование, продуктивность работы ученых и т. д. Сигналом, передающим информацию, служит значение одного или нескольких параметров регистрируемого процесса. В связи с этим существуют разные варианты классификации информации.

1. *По форме представления*:

- ***дискретная*** информация: характеризуется прерывистой, изменяющейся величиной, например, количество дорожно-транспортных происшествий, количество символов в том или ином алфавите, количество занятых байт в памяти компьютера и т. п. Сигнал, переносящий информацию, представляется последовательностью символов алфавита, принятого в данной предметной области;
- ***аналоговая*** информация: (непрерывная) представляется сигналом, измеряемый параметр которого может принимать любые промежуточные значения в определенных пределах. Например, температура тела человека, скорость автомобиля на определенном участке пути, воспроизведение звука на виниловой пластинке, так как звуковая дорожка на ней изменяет свою форму непрерывно, плавные переходы цветов на живописном полотне и т. п. Аналоговую информацию можно преобразовать в дискретную с некоторой потерей промежуточных значений.

Для цифровой техники наиболее удобна дискретная форма представления информации.

2. *По области возникновения выделяют информацию*:

- ***механическую***, которая отражает процессы и явления неодушевленной природы;
- ***биологическую***, которая отражает процессы животного и растительного мира;
- ***социальную***, которая отражает процессы человеческого общества.

3. *По способу передачи и восприятия различают следующие виды информации*:

- ***визуальную***, передаваемую видимыми образами и символами;
- ***аудиальную***, передаваемую звуками;
- ***тактильную***, передаваемую ощущениями прикосновений;
- ***органолептическую***, передаваемую запахами и вкусами;
- ***машинную***, выдаваемую и воспринимаемую средствами вычислительной техники.

4. *Информацию, создаваемую и используемую человеком, по общественному назначению можно разбить на три вида*:

- ***личную***, предназначенную для конкретного человека;
- ***массовую***, предназначенную для любого желающего ею пользоваться (общественно-политическая, научно-популярная и т.д.);
- ***специальную***, предназначенную для использования узким кругом лиц, занимающихся решением сложных специальных задач в области науки, техники, экономики.

5. *По способам кодирования выделяют следующие типы информации*:

- ***символьную***, основанную на использовании символов – букв, цифр, знаков и т. д. Она является наиболее простой, но применяется только для передачи несложных сигналов о различных событиях. Примером может служить зеленый свет уличного светофора, который сообщает пешеходам и водителям автотранспорта о возможности начала движения;
- ***текстовую***, основанную на использовании комбинаций символов. Здесь так же, как и в предыдущей форме, используются символы: буквы, цифры, математические знаки. При этом в текстовой информации принципиально важен не только состав, но и порядок следования символов. Так, слова КОТ и ТОК имеют одинаковые буквы, но содержат различную информацию. Текстовая информация

чрезвычайно удобна и широко используется в деятельности человека: книги, брошюры, журналы, различного рода документы, аудиозаписи кодируются в текстовой форме;

- ***графическую***, основанную на использовании произвольного сочетания графических примитивов. К этой форме относятся фотографии, схемы, чертежи, рисунки, играющие большое значение в деятельности человека.

1.4. Количество информации как мера уменьшения неопределенности знаний

Подход к информации как к мере уменьшения неопределѐнности наших знаний позволяет количественно измерять информацию, полученную через некоторое сообщение.

Например, после сдачи зачета Вы получаете одно из двух информационных сообщений: "зачет" или "незачет", а после сдачи экзамена одно из четырех информационных сообщений: "2", "3", "4" или "5".

Информационное сообщение об оценке за зачет приводит к уменьшению неопределенности вашего знания в два раза, так как реализуется один из двух возможных вариантов. Информационное сообщение об оценке за экзамен приводит к уменьшению неопределенности вашего знания в четыре раза, так как получено одно из четырех возможных информационных сообщений.

Ясно, что чем более неопределенна первоначальная ситуация, тем больше мы получим новой информации при получении информационного сообщения о том, как она разрешилась (тем в большее количество раз уменьшится неопределенность знания).

Клод Шеннон предложил в 1948 году формулу для определения количества информации, которую мы получаем после получения одного из N возможных сообщений:

$$I = – (p_1\log_2 p_1 + p_2\log_2 p_2 + \dots p_i\log_2 p_i + \dots + p_N\log_2 p_N)$$

Здесь p_i – вероятность того, что будет получено именно i-е сообщение. Если все сообщения равновероятны, то все $p_i=1/N$, и из этой формулы получается формула Хартли:

$$I = \log_2 N$$

Для количественного выражения любой величины необходимо сначала определить единицу измерения. Так, для измерения длины

в качестве единицы выбран метр, для измерения массы - килограмм и т. д. Аналогично, для определения количества информации необходимо ввести единицу измерения.

Из формулы Хартли следует: если I=1, то N=2, то есть в качестве единицы измерения информации можно взять тот объѐм информации, который мы получаем при принятии сигнала о том, что же произошло в ситуации с двумя возможными исходами. Такая единица названа ***битом***.

Наряду с единицей бит иногда используют в качестве единиц информации количества, взятые по логарифмам с другими основаниями: ***дит*** – по десятичному логарифму (за единицу информации выбирается количество информации, необходимой для различения десяти равновероятных сообщений), ***нат*** – по натуральному основанию.

Используя формулу Хартли можно, также, зная количество информации, пришедшее с одним из равновероятных сообщений, определить, сколько сообщений вообще можно было ожидать в данной ситуации. Решив это уравнение относительно N, получим при равновероятных исходах:

$$I = \log_2 N \rightarrow N = 2^I$$

Например, на экзамене вы берете экзаменационный билет, и учитель сообщает, что зрительное информационное сообщение о его номере несет 5 бит информации. Если вы хотите определить количество экзаменационных билетов, то достаточно определить количество возможных информационных сообщений об их номерах из формулы Хартли:

$$5 = \log_2 N \rightarrow N = 2^5 = 32.$$

Таким образом, количество экзаменационных билетов равно 32.

Задача: Представьте себе, что вы управляете движением робота и можете задавать направление его движения с помощью информационных сообщений: "север", "северо-восток", "восток", "юго-восток", "юг", "юго-запад", "запад" и "северо-запад" (рис. 1.1). Какое количество информации будет получать робот после каждого сообщения?

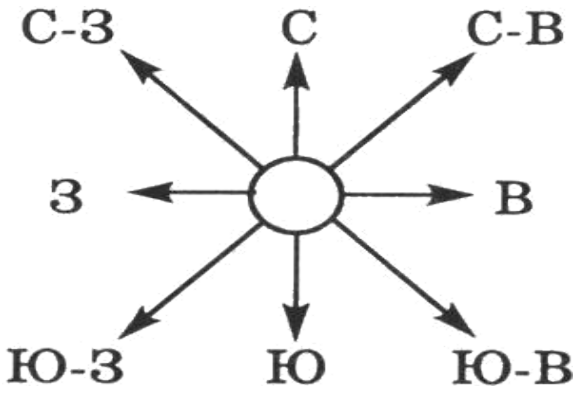


Рис. 1.1. Управление роботом с использованием информационных сообщений

В этой задаче робот может получить 8 разных информационных сообщений. Формула Хартли принимает вид уравнения относительно I:

$$8=2^I$$

Так как $8 = 2^3$, получаем

$$2^3=2^I \rightarrow I=3$$

Количество информации, которое несет роботу каждое информационное сообщение, равно 3 битам.

1.5. Алфавитный подход к определению количества информации

При алфавитном подходе к определению количества информации отвлекаются от содержания информации и рассматривают информационное сообщение как последовательность знаков определенной знаковой системы.

Представим себе, что необходимо передать информационное сообщение по каналу передачи информации от отправителя к получателю. Пусть сообщение кодируется с помощью знаковой системы, алфавит которой состоит из N знаков {1, ..., N} и вероятности появления каждого знака в сообщении равны.

В простейшем случае, когда длина кода сообщения составляет один знак, отправитель может послать N разных сообщений. Количество информации I, которое несет каждое сообщение, то есть один знак, можно рассчитать по формуле Хартли.

$$I = \log_2 N$$

Эта величина называется ***информационной емкостью знака***. С помощью этой формулы можно, например, определить информационную емкость знака двоичной знаковой системы:

$$I = \log_2 2 = 1 \text{ бит}$$

Интересно, что сама единица измерения количества информации "бит" (bit) получила свое название от английского словосочетания "binary digit" – "двоичная цифра".

Чем большее количество знаков содержит алфавит знаковой системы, тем большее количество информации несет один знак. В качестве примера определим количество информации, которое несет буква русского алфавита. В русский алфавит входят 33 буквы, однако на практике часто для передачи сообщений используются только 32 буквы (исключается буква "è").

С помощью формулы Хартли определим количество информации, которое несет буква русского алфавита:

$$N = 32 \rightarrow I = \log_2 32 \rightarrow I = \log_2 2^5 \rightarrow I=5 \text{ бит}.$$

Таким образом, информационная емкость буквы русского алфавита равна 5 битам (если считать, что все буквы используются в сообщении с равной вероятностью).

Количество информации, которое несет знак, зависит от вероятности его получения. Если получатель заранее точно знает, какой знак придет, то полученное количество информации будет равно 0. Наоборот, чем менее вероятно получение знака, тем больше его информационная емкость.

Сообщение состоит из последовательности знаков, каждый из которых несет определенное количество информации. Если знаки несут одинаковое количество информации, то количество информации $I_с$ в сообщении можно подсчитать, умножив количество информации $I_з$, которое несет один знак, на длину кода K (количество знаков в сообщении):

$$I_с = I_з * K$$

Например, каждая цифра двоичного компьютерного кода несет информацию в 1 бит. Следовательно, две цифры несут информацию в 2 бита, три цифры - в 3 бита и т. д. Количество информации в битах равно количеству цифр двоичного компьютерного кода (табл. 1.1).

Таблица 1.1.

Количество информации, которое несет двоичный компьютерный код

Двоичный компьютерный код	111	01	11	011	0001
Количество информации	3 бит	2 бит	2 бит	3 бит	4 бит

В русской письменной речи частота использования букв в тексте различна, так в среднем на 1000 знаков осмысленного текста приходится 200 букв "а" и в сто раз меньшее количество буквы "ф" (всего 2). Таким образом, с точки зрения теории информации, информационная емкость знаков русского алфавита различна (у буквы "а" она наименьшая, а у буквы "ф" - наибольшая) и информацию, которое несѐт текстовое сообщение, надо рассчитывать с учетом вероятности появления букв, входящих в него.

1.6. Единицы измерения информации

Бит – это минимальная единица измерения количества информации. Более крупные единицы формируются в информатике способом, который несколько отличается от принятых в большинстве наук. Первой более крупной, чем бит, единицей измерения информации, выбран ***байт***:

$$1 \text{ байт} = 8 \text{ бит} = 2^3 \text{ бит}.$$

Для измерения более крупных объемов информации используются приставки, применяемые в традиционной международной системе единиц СИ. В качестве множителей кратных единиц в ней используют коэффициент 10^n, где n = 3, 6, 9 и т. д. Это соответствует десятичным приставкам "Кило" (10^3), "Мега" (10^6), "Гига" (10^9) и т. д.

В компьютере информация кодируется с помощью двоичной знаковой системы, и поэтому в кратных единицах измерения количества информации используют коэффициент 2^n, а не 10^n. Так как $10^3 \approx 2^{10}$, для крупных единиц информации используются те же приставки, что и в системе СИ:

1 Килобайт (Кбайт) = 2^{10} байт = 1024 байт;

1 Мегабайт (Мбайт) = 2^{10} Кбайт = 1024 Кбайт=1 048 576 байт;

1 Гигабайт (Гбайт) = 2^{10} Мбайт = 1024 Мбайт = 1 073 741 824 байт;

1 Терабайт (Тбайт)= 2^{10} Гбайт = 1024 Гбайт$\approx 2^{40}$ байт;

1 Петабайт (Пбайт) = 2^{10} Тбайт = 1024 Тбайт ≈ 2^{50} байт.

1.7. Системы счисления

Для записи информации о количестве объектов используются ***числа***. Числа записываются с использованием особых знаковых систем, которые называются ***системами счисления***. Символы алфавита систем счисления называются ***цифрами***. Различают ***позиционные*** и ***непозиционные*** системы счисления.

В ***непозиционных*** системах значение цифры не зависит от положения в числе. Примером записи чисел в таких системах может служить римская система. В качестве цифр в ней используются некоторые буквы латинского алфавита. Количество, сопоставленное им, приведено в табл. 1.2.

Таблица 1.2.

Цифры римской системы счисления

.Римская цифра	I	V	X	L	C	D	M
Значение в метрической системе	1	5	10	50	100	500	1000

Число представляется как сумма или разность последовательности нужных цифр. Если слева от следующей стоит цифра, соответствующая меньшему количеству, она вычитается, если справа – прибавляется к числу. Пример:

$$IIXXX=10–1–1+10+10=28_{10}$$

В ***позиционных*** системах количественное значение цифры зависит от еѐ положения в числе. Обычно при записи числа в позиционных системах используют арабские цифры. Количество цифр, которое используется при этом, называется ***основанием*** системы. Оно определяет, во сколько раз различаются количества, соответствующие одинаковым цифрам, стоящим в соседних позициях числа, и указывается нижним индексом после последней цифры числа. Если основание системы, по которой записано число, не указано, по умолчанию считается, что оно равно десяти.

Количество, соответствующее числу, можно представить в виде многочлена по степеням основания. Цифры, из которых составляется число, это коэффициенты, на которые надо умножить соответствующие степени основания. Первая цифра справа – коэффициент при

нулевой степени основания. Далее справа налево перечисляются коэффициенты при первой, второй и т. д. степенях. Примеры:

$333_{10} = 3 * 10^2 + 3 * 10^1 + 3 * 10^0$;

$333_{12} = 3*12^2+3*12^1+3*12^0 = 3*144+3*12+3 = 471_{10}$ $1F3D_{16} = 1 * 16^3 + 15 * 16^2 + 3 * 16^1 + 13 * 16^0 = 7997_{10}$

$37_8 = 3 * 8^1 + 7 * 8^0 = 31_{10}$

$0110_2 = 0 * 2^3 + 1 * 2^2 + 1 * 2^1 + 0 * 2^0 = 6_{10}$

1Кб = 2^{10} байт = 10000000000_2 байт = 1024_{10} байт

Дробная часть числа раскладывается в многочлен по отрицательным степеням основания.

Алгоритмы перевода целого и дробного числа из одной позиционной системы в другую различны. Приведем в качестве примера алгоритм перевода целого числа *A* из привычной для нас системы по основанию 10 в число по основанию *k*. Для этого надо представить его как многочлен по степеням *k* (значения всех коэффициентов меньше *k*):

$$A = a_{n-1} * k^{n-1} + a_{n-2} * k^{n-2} + \dots a_1 * k^1 + a_0 * k^0$$

Коэффициенты при степенях *k* – это цифры числа A_k, обозначающие то же количество в новой системе:

$$A_k = a_{n-1}a_{n-2}\dots a_1a_0$$

Для того, чтобы определить их, на первом шаге разделим нацело число *A* на *k*:

$$A_{1ц} = A / k = a_{n-1} * k^{n-2} + a_{n-2} * k^{n-3} + \dots a_1 * k^0 .$$

a_0 – остаток от деления – это младшая цифра числа A_k. Теперь разделим нацело число $A_{1ц}$ на *k*. Аналогично предыдущему получим численные значения остальных коэффициентов.

Применим этот алгоритм для представления числа 333_{10} в пятеричной системе. Обозначим операцию деления нацело как *div*(*A*; *k*), где *A* – делимое, *k* – делитель:

1) $A_{1ц} = div\ (333; 5) = 66$, остаток $a_0 = 3$;
2) $A_{2ц} = div\ (66; 5) = 13$, остаток $a_1 = 1$;
3) $A_{3ц} = div\ (13; 5) = 2$, остаток $a_2 = 3$;
4) $A_{4ц} = div\ (2; 5) = 0$, остаток $a_3 = 2$;
5) $333_{10} = 2313_5$;
6) Проверка: $3 * 5^0 + 1 * 5^1 + 3 * 5^2 + 2 * 5^3 = 333$

Базовой системой счисления в вычислительной технике является двоичная система. Так как коды чисел и команд в ней слишком длинные, в документации используют более компактную запись по родственным основаниям: в восьмеричной или шестнадцатеричной системе. В восьмеричной системе для записи числа используются цифры 0, 1, 2,.., 7. В шестнадцатеричной системе арабские цифры 0, 1,2,…, 9 дополняются начальными буквами латинского алфавита.

Из табл. 1.3 видно, что если добавить слева незначащие ноли, то значение каждой цифры восьмеричной системы можно представить тремя, а шестнадцатеричной – четырьмя цифрами двоичной системы.

При переводе числа из двоичной системы в восьмеричную число справа налево разбивают на группы по три разряда, и каждую тройку двоичных цифр заменяют одной восьмеричной. Если в последней группе слева осталась только одна или две цифры, ее дополняют нолями.

Перевод в шестнадцатеричную систему делается аналогично, но двоичное число разбивают на группы по четыре цифры:

$$10001111111010_2 \rightarrow 10\ 001\ 111\ 111\ 010_2 \rightarrow 21772_8$$

$$10001111111010_2 \rightarrow 10\ 0011\ 1111\ 1010_2 \rightarrow 23FA_{16}$$

Таблица 1.3.

Таблица соответствия между начальными двоичными, восьмеричными, шестнадцатеричными и десятичными числами:

Основание системы счисления	Вид числа							
10	0	1	2	3	4	5	6	7
2	0000	0001	0010	0011	0100	0101	0110	0111
8	00	01	02	03	04	05	06	07
16	0	1	2	3	4	5	6	7
10	8	9	10	11	12	13	14	15
2	1000	1001	1010	1011	1100	1101	1110	1111
8	10	11	12	13	14	15	16	17
16	8	9	A	B	C	D	E	F

1.8. Характеристики основных типов данных

Наряду с информационным смыслом термином ***бит*** в вычислительной технике используют для обозначения наименьшего эле-

мента памяти, необходимого для хранения одного из двух знаков: «0» или «1», используемых для представления данных и команд внутри компьютера.

Бит очень удобен для использования ***двоичной формы*** представления информации. Для каждого типа информации (символьный, текстовый, графический, числовой) был найден способ представить е в едином виде как последовательность, использующую в разных комбинациях только два символа. Каждая такая последовательность называется двоичным кодом. Недостаток двоичного кодирования – длинные коды. Но в технике легче иметь дело с большим числом простых однотипных элементов, чем с небольшим числом сложных.

Двоичные символы могут представляться любыми способами: буквами А, Б; словами ДА, НЕТ, двумя разными, но устойчивыми состояниями системы и т. д. Для простоты записи были выбраны цифры0и1.

Характеристики основных типов данных, которые обрабатываются современными компьютерами, приведены в табл.1.4.

Таблица 1.4.

Основные виды данных, с которыми работает ЭВМ.

Тип данных	Название типа данных	Объем памяти	Диапазон значений
Целые числа (формат с фиксированной запятой)	Byte (короткое целое)	1 байт	От – 127 до + 127
	Integer	2 байта	От – 32768 до + 32767
	Long	4 байта	От – 2 147 483 648 до + 2 147 483 647
Вещественные числа (формат с плавающей запятой)	Single (7 – 8 значащих цифр)	4 байта	От $-1{,}4*10^{-45}$ до $+3{,}4*10^{38}$
	Double (15 – 16 значащих цифр)	8 байт	от $-5{,}0*10^{-324}$ до $+1{,}7*10^{308}$
Символьные	Первоначальные таблицы 8-битовых кодировок (ASCII, КОИ8-Р, СР1251, СР866, ISO 8859-5 и другие)	1 байт	256 символов
	Юникод	2 байта	65536 символов
Логические (Boolean)		1 байт	0; 1

1.9. Кодирование числовой информации в компьютере

Числа в компьютере переводятся в двоичную позиционную систему и размещаются в отведенном для них месте в формате с фиксированной или плавающей запятой.

В ***формате с фиксированной запятой*** каждому разряду памяти всегда соответствует один и тот же разряд числа. Например, при кодировании целых чисел, младший разряд числа размещается в последнем (крайнем справа) бите памяти, отведенной под код числа. То есть место для дробной части числа не предусматривается.

Для кодирования целых чисел используется три варианта кодировок: ***прямой***, ***обратный*** и ***дополнительный коды***. Для положительных целых чисел используется прямой код. Обратный и дополнительный коды добавляются при кодировании отрицательных чисел. Они позволяют заменить операцию вычитания на сложение, что существенно упрощает работу процессора и увеличивает его быстродействие.

В ***прямом коде*** первый бит памяти, отведѐнный под число, показывает знак числа: 0 – положительное, 1 – отрицательное. Остальные биты отводятся под двоичный код модуля числа. Примеры:

$$127_{10} \rightarrow 01111111_2;$$
$$-127_{10} \rightarrow 11111111_2;$$
$$1_{10} \rightarrow 00000001_2;$$
$$-1_{10} \rightarrow 10000001_2.$$

В ***обратном коде*** все двоичные цифры, кроме первой (указывающей на знак), инвертируют: заменяют $0 \rightarrow 1$, $1 \rightarrow 0$. Примеры:

$$-127_{10} \rightarrow 11111111_2 \rightarrow 10000000_2;$$
$$-1_{10} \rightarrow 11111110_2 \rightarrow 00000001_2.$$

Дополнительный код получают из обратного кода целого отрицательного числа, прибавляя к нему 1_2. Примеры:

$$-1_{10} \rightarrow 11111110_2 \rightarrow 10000001_2 \rightarrow 1111\ 1110_2;$$
$$-127_{10} \rightarrow 11111111_2 \rightarrow 10000000_2 \rightarrow 1000\ 0001_2$$

В ***формате с плавающей запятой*** представляются вещественные числа (предполагается, что они могут содержать дробную часть). В этом формате число заносится в память компьютера в экспоненциальной форме, то есть в виде двух сомножителей: ***мантиссы*** (дроби, в которой первая значащая цифра стоит сразу после запятой)

и основания системы счисления в соответствующей степени (***порядке***). Примеры для десятичной системы счисления приведены в таблице 1.5.

Таблица 1.5

Примеры перевода вещественных чисел в экспоненциальную форму

Вещественное число	Экспоненциальная форма	Мантисса	Порядок
98567	$0{,}98567*10^{5}$	0,98567	5
– 98567	$-0{,}98567*10^{5}$	–0,98567	5
98,567	$0{,}98567*10^{2}$	0,98567	2
– 0,0009856	$-0{,}9856*10^{-3}$	– 0,9856	–3

В байтах, отведенных для записи числа, выделяются определенные разряды для хранения всех фрагментов числа: знаков мантиссы и порядка, их абсолютных значений. Пример (код максимального положительного числа):

0	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Знак и значение порядка								Знак и значение мантиссы																

1.10. Кодирование текстовой информации в компьютере

Компьютерная обработка текстовой информации начала использоваться с середины 60-х годов. Помимо преимуществ, которые появляются при автоматическом внесении текстовых комментариев в результаты расчетных программ, создание, обработка и хранение текстовых документов в файловом виде представляет массу удобств.

При кодировании текста в память последовательно заносятся коды символов, составляющих текст, и команд, управляющих внешним видом и размещением этих символов. То есть если мы определяем числа 69 и 96 как текстовую информацию, коды этих чисел будут отличаться только порядком следования кодов цифр 6 и 9. Если же мы определяем их как числовую информацию, их коды будут совершенно различны, так как они представляют разные по величине числа.

Первоначально для кодирования одного символа использовался 1байт. В байт можно записать в $2^{8} = 256$ разных кодов (состояний). Эти состояния перенумерованы, и каждому сопоставляется какой-либо буквенный символ, графический элемент или команда, необходимая при оформлении текстовой информации. Такое соответствие называется ***кодовой таблицей.***

В настоящее время существуют и применяются разные варианты 8-битных кодовых таблиц. Наиболее популярные из них:

- ***ASCII*** – American Standart Code for Information Interchange – американский стандартный код для обмена информацией;
- ***КОИ8-Р*** – Код Обмена Информацией 8-битный с кириллицей;
- ***CP1251*** – (Code Page) – кодировка с кириллицей в Microsoft Windows;
- ***CP866*** – кодировка MSDOS;

ISO 8859-5 – International Standards Organization – Международная организация по стандартизации. Ещѐ один стандарт для кодов для кириллицы.

Множество кодовых таблиц вызвано тем, что с учетом разнообразия естественных языков и фирм, выпускающих программное обеспечение, 256 состояний одного байта недостаточно для того, чтобы закодировать все встречающиеся символы и способы форматирования текста.

При разработке всех кодовых таблиц использовано следующее соглашение: первая половина таблицы – это коды с 0 по 127 – интернациональна, т. е одинакова во всех вариантах кодировок. Первые 33 состояния (0–32) – это коды операций с текстом (перевод на новую строку, пробел, удаление последнего символа и т. п.). Затем состояния с 33 по 127 – это коды знаков препинания, арифметических действий, цифр, строчных и прописных букв ***латинского*** алфавита. Вторая половина кодовых таблиц отводится под знаки ***национальных и специальных*** алфавитов и ввода в текст графических элементов для оформления таблиц.

В конце 90-х годов появился новый международный стандарт ***Unicode***, который отводит под символ 2 байта. Каждый блок из 2-х байт может находиться в 2^{16} =65536 состояниях. Этого достаточно, чтобы в одной таблице собрать символы большинства алфавитов мира. Правда, длина текста удваивается, и скорость его обработки замедляется. Но, в связи с существенным увеличение памяти и быстродействия современных компьютеров, этот факт несущественен.

1.11. Кодирование графической информации в компьютере

Графическая информация в докомпьютерную эпоху регистрировалась и воспроизводилась в аналоговой форме. Чертежи, рисунки создавались с помощью сплошных линий и мазков разной величины и цвета.

Дискретное представление графики получается за счет того, что экран монитора разбивается на строки и колонки. Совокупность получившихся клеточек (точек) называется ***растром***, каждая точка – ***пикселем***. Количество строк и колонок в растре – это р***азрешение*** (***разрешающая способность***) ***экрана***. Типовые разрешения: 1024□768 , 800□600 пикселей. Первым указывается количество колонок, вторым – количество строк в растре.

Для монохромных изображений общепринятым считается кодирование цвета одного пикселя в 1байте. Это позволяет передать 254 оттенка серого плюс черный и белый цвета (всего 256 вариантов). Цветные изображения могут кодироваться разными способами в за-висимости от того, для какой цели создаѐтся рисунок.

Метод (система) RGB (***True color***) – от слов Red, Green, Blue удобен для изображений, рассматриваемых на экране, выводи-мых на устройство записи на киноплѐнку. Оттенки цвета создаются смешением лучей трѐх базовых цветов разной интенсивности. Под значение интенсивности каждого луча отводится 1 байт, т. е.различают 256 уровней интенсивности. Для совокупности трѐх лучей получается 256^3=16 777 216≈17 млн. разных вариантов, каждый из которых создает свой оттенок цвета.

Метод (система) CMYK – от слов голубой (Cyan), пурпурный (Magenta), жѐлтый (Yellow), чѐрный (blacK): удобен для изображений, которые предполагается печатать на бумаге. Он учитывает особенности полиграфии, в которой цвет получается смешением четырѐх красок. Для кодирования одного пикселя требуется 4 байта и можно передать 256^4 4 млрд. оттенков.

Для WEB-документов учитывать такое обилие оттенков неудобно, так как это приводит к файлам очень больших размеров, и их неудобно пересылать по сети. Поэтому в них используются так называемые ***индексированные цвета***: из всего обилия возможных комбинаций выбрано 256 базовых оттенков. Это позволило для запоминания цвета каждого пикселя использовать только 1 байт. Каждому со-

стоянию байта сопоставляется определенная комбинация интенсивностей базовых цветов. *.JPG, *.GIF, *.PNG- кодировки объединяют области рисунка, закрашенные близкими оттенками, и сохраняют для них усреднѐнный цвет. За счет этого размеры графических файлов существенно уменьшаются.

В компьютерной документации коды RGB- и CMYK-цвета представляются так:

1). Интенсивности каждого базового цвета перечисляются в том порядке, который использован в аббревиатуре используемой системы. Эти интенсивности представляются в 16-ричной позиционной системе (п. 1.9).

2). Перед полученным кодом оттенка размещают символ #.

Примеры составления кодов для некоторых цветов:

#000000 – чѐрный цвет – нулевая интенсивность каждого луча, т. е. нет ни одного цвета;
#B5B5B5 – какой-то оттенок серого цвета, т. к. интенсивности всех лучей одинаковы;
#FFFFFF – белый цвет – все цвета в максимальной интенсивности;
#CF35D1 – номера интенсивностей базовых лучей в десятичной системе: красного CF = 12 *16 + 15 = 207; зелѐного – 35 = 3*16+5=53;
синего – D1 = 13 *16 + 1 = 209.

1.12. Кодирование аудио информации в компьютере

Звук – это колебания физической среды. В повседневной жизни такой средой является воздух. Чаще всего звуковые колебания преобразуют в электрические с помощью микрофона. Представление о форме этого сигнала можно получить через программу Windows Player.

Звуковой (аудио) сигнал имеет аналоговую природу. Для того чтобы преобразовать его в дискретную форму используют специальный блок, входящий в состав звуковой карты компьютера, АЦП (аналого-цифровой преобразователь). Основной принцип его работы заключается в том, что интенсивность звукового сигнала фиксируется не непрерывно, а периодически, в определенные моменты времени. Частоту, характеризующую периодичность измерения, называют

частотой дискретизации. Считается, что для хорошего воспроизведения звука она должна, по крайней мере, в два раза превышать максимальную частоту волны, входящей в спектр звукового сигнала. Человеческое ухо воспринимает как звук колебания в диапазоне частот до 22 000 Гц. Следовательно, для хорошего воспроизведения музыки частота дискретизации должна быть не менее 44 000 Гц. При записи речи такое высокое качество воспроизведения не нужно. Определено, что речь воспринимается вполне разборчиво уже при частоте дискретизации 8 000 Гц.

Помимо дискретизации по времени АЦП проводит дискретизацию и по интенсивности звука, т. е. по амплитуде звукового сигнала. В АЦП закладывается сетка стандартных интенсивностей – ***глубина кодирования*** (256 или 65 536 уровней), и реальная интенсивность округляется до уровня, ближайшего по сетке.

Обратное преобразование закодированного таким образом звука в аналоговую форму, воспринимаемую человеческим ухом, производится блоком ЦАП (цифро-аналоговый преобразователь). По закодированным точкам время-интенсивность с помощью интерполяции рассчитывается гладкая непрерывная кривая, которая используется при восстановлении звукового сигнала. Для проведения расчетов, восстанавливающих вид звукового сигнала, выпускаются специализированные микропроцессоры, DSP (Digital Signal Processor).

1.13. Вопросы для самопроверки по теме 1

Задание № 1. Определите, сколько бит содержит сообщение: «на улице идѐт дождь».

Задание № 2. Определите, сколько бит может содержать сообщение: вылет самолѐта задерживается.

Задание № 3. Выберите верное утверждение:

1. в качестве материального носителя информации могут выступать знания, сведения или сообщения;
2. в качестве носителя информации могут выступать только световые и звуковые волны;
3. в качестве носителя информации могут выступать материальные предметы;
4. информационные процессы являются материальным носителем информации.

Задание № 4. Выберите верное утверждение: энтропия максимальна, если:

1. информация засекречена;
2. события детерминированы;
3. события равновероятны;
4. информация точна.

Задание №5. Выберите верное утверждение: Прагматический аспект информации рассматривает:

1. информацию с точки зрения еѐ практической полезности для получателя;
2. даѐт возможность раскрыть еѐ содержание;
3. показать отношения между смысловыми значениями еѐ элементов;
4. показывает количество информации в утверждении.

Задание № 6. Выберите верное утверждение: свойство информации, заключающееся в достаточности данных для принятия решения, есть:

1. объективность;
2. полнота;
3. содержательность;
4. достоверность.

Задание № 7. Выберите верное утверждение: информация достоверна, если она:

1. полезна;
2. отражает истинное положение дел;
3. достаточна для принятия решений;
4. используется в современной системе обработки информации.

Задание № 8. Выберите верное утверждение: сообщением в теории кодирования является:

1. электрический импульс, распространяемый в канале связи телефонной линии;
2. воспринятая, осознанная и ставшая личностно значимой информация;
3. набор данных, объединѐнных смысловым содержанием и пригодных для обработки и передачи по каналам связи;

4. процесс переноса или копирования данных по некоторым признакам с одного места на другое с целью сортировки, формирования результирующих документов.

Задание № 9. Выберите верное утверждение: цепочка костров, зажигающаяся при необходимости оповещения "Горит - да", "Не горит - нет" – это:

1. линия передачи сообщения;
2. неадекватное поведение людей;
3. способ обработки информации;
4. шифрование информации.

Задание № 10. Выберите верное утверждение: в вычислительной технике в качестве основной используется _____система счисления:

1. шестнадцатеричная;
2. десятичная;
3. восьмеричная;
4. двоичная.

Задание № 11. Выберите верное утверждение: сканирование книги является операцией _____ данных:

1. верификации;
2. преобразования;
3. архивирования;
4. транспортировки.

Задание № 12. Представьте десятичное число 1023 в двоичной системе.

Задание № 13. Расположите в возрастающей последовательности следующие единицы измерения информации: 1Кб, 1Мб, 1Тб, 1Гб.

Задание № 14. Определите, сколько двоичных разрядов необходимо для кодирования двадцати состояний.

Задание № 15. Сколько информации содержится в одном разряде двоичного числа?

Задание № 16. Какой объем памяти потребуется для хранения 32 символов в кодировке KOI-8?

Задание № 17. Определите последнюю цифру суммы чисел 55_8 и 56_8 в шестнадцатеричной системе счисления.

Задание № 18. Отсортируйте по возрастанию последовательность текстовых величин: 8б; 8а; 10а; 10б; 11а.

Задание № 19. Упорядочьте по убыванию последовательность чисел: 10 бит, 20 бит, 2 байта.

Задание № 20. Определите объѐм памяти, который потребуется для кодировки фразы «Я помню чудное мгновенье» в Unicode.

Задание № 21. Представьте в десятичной системе результат суммирования 111_2 и 111_2.

Задание № 22. Числа в двоичной системе имеют вид: 10101_2 и 1000_2. Какой вид имеет их разность?

Задание № 23. Представьте в двоичной системе результат вычисления 2^7+2^4+1.

Задание № 24. Упорядочьте по возрастанию последовательность чисел: 55_8, 55_{16}, 55_7. Ответ:.

Задание № 25. Упорядочить по убыванию последовательность чисел: 10 бит, 20 бит, 2 байта.

Задание № 26. Укажите, к какому типу относится информация, представленная в виде слов

Задание № 27. Определить информационный объѐм фразы «Я помню чудное мгновенье» при алфавитном подходе к определению количества информации, если считать, что информационная емкость буквы русского алфавита равна 5 битам. Ответ: 24 символа*5 бит=120 бит=15 байт.

2. ОСНОВЫ ЛОГИКИ И ЛОГИЧЕСКИЕ ОСНОВЫ КОМПЬЮТЕРА

2.1. Основные понятия алгебры логики

Алгебра логики – это раздел математики, изучающий логические высказывания и логические операции над ними.

Логическое высказывание – это любое повествовательное предложение, про которое можно однозначно сказать, истинно оно или ложно. Примеры: «студент 1501 группы», «в городе более миллиона жителей» (без указания названия города) – не являются логическими высказываниями. «Иванов – студент 1501 группы и живет в Санкт-Петербурге», «в городе Санкт-Петербурге более миллиона жителей» – это логические высказывания.

Слова и словосочетания «не», «и», «или», «если…, то», «тогда и только тогда» называются ***логическими связками***. С их помощью формируются ***составные высказывания*** из более простых. ***Элементарные высказывания*** – не содержат в себе других высказываний. В предыдущих примерах «в городе Санкт-Петербурге более миллиона жителей» – элементарное высказывание, а «Иванов – студент 1501 группы и живет в Санкт-Петербурге» – составное, так как его можно разбить на два более простых: «Иванов – студент 1501 группы», «Иванов живет в Санкт-Петербурге».

Для того чтобы исследовать общие характеристики высказываний и упростить их запись и анализ, абстрагируясь от предметной области, к которой они относятся, их обозначают буквами латинского алфавита, и рассматривают как логические переменные, принимаю-щие только два значения: «истина» и «ложь». Например, для приве-денных выше примеров:

A = «Иванов – студент 1501 группы»;
B = «Иванов живет в Санкт-Петербурге»;
A и B = «Иванов – студент 1501 группы и живет в Санкт-Петербурге».

2.2. Основные логические операции

Каждая логическая связка рассматривается как операция, результат которой зависит от значений входящих в неѐ переменных (то есть высказываний). Для упрощения записи вместо слов «истина» и «ложь» используют двоичные цифры: «истина» = 1, «ложь» = 0. Ос-

новные логические операции и их результаты при разных значениях высказываний приведены в табл.2.1.

Таблица 2.1

Основные логические операции (иерархия сверху вниз):

Название	Обозначение	Результат
Отрицание, инверсия (связка «не»)	A, $\bar{A}$	A = 0 $\bar{A}$ = 1 A = 1 $\bar{A}$ = 0
Конъюнкция, лог. умножение (связка «и»)	$\bar{A}$ B, A&B, A ∧ B	A=1,B=1 A B = 1, в остальных случаях – = 0
Дизъюнкция, лог. Сложение (связка «или»	A v B, A + B	A=0,B=0 A v B = 0, в остальных случаях – = 1
Импликация (связки «если…, то», «из… следует», … влечѐт…»	A → B	A=1,B=0,тоA B = 0, в остальных случаях = 1
Эквиваленция, двойная импликация (связки «тогда и только тогда», «необходимо и достаточно», «…равносильно…»,	AB;AB ↔	A=1,B=1 A ~ B = 1 A=0,B=0 A ~ B = 1 A=1,B=0A B=0 ~ A=0,B=1 A ~ B = 0

Анализ составных логических высказываний удобно делать с помощью ***таблиц истинности***. В них представляют все возможные комбинации значений элементарных высказываний, которые входят в составное, и его результирующее значения для каждой из них. Примеры:

Конъюнкция		
A	B	F=A•B
0	0	0
0	1	0
1	0	0
1	1	1

Дизъюнкция		
A	B	F =A+B
0	0	0
0	1	1
1	0	1
1	1	1

Импликация		
A	B	F=A→B
0	0	1
0	1	1
1	0	0
1	1	1

Эквиваленция		
A	B	F=A~B
0	0	1
0	1	0
1	0	0
1	1	1

Анализ логического высказывания А и не В и не А ($A \cdot \neg B \cdot \neg A$)

A	B	$Y_1 = \neg A$	$Y_2 = \neg B$	$Y_3 = A \cdot Y_2$	$Y_4 = Y_3 \cdot Y_1$
0	0	1	1	0	0
0	1	1	0	0	0
1	0	0	1	1	0
1	1	0	0	0	0

Ответ: выражение тождественно ложно.

Анализ логического высказывания А и не А или В ($A \cdot \neg A \vee B$)

A	B	$Y_1 = \neg A$	$Y_2 = A \cdot Y_1$	$Y_3 = Y_2 \vee B$
0	0	1	0	0
0	1	1	0	1
1	0	0	0	0
1	1	0	0	1

Ответ: значение выражения совпадает со значением В при любом А.

Также как и для чисел, существуют законы, позволяющие производить тождественные преобразования сложных логических выражений к более понятному и удобному виду.

Задание: представить в символах логики высказывание «Если завтра будет дождь, то я возьму зонтик или никуда не пойду».

Ответ: А = «Завтра будет дождь»; В = «Я возьму зонтик»; С = «Я никуда не пойду». $A \rightarrow (B \vee C)$

2.3. Логические основы ЭВМ

Данные в компьютере запоминаются в двоичной системе, и результаты проверки логических выражений тоже можно представлять в двоичной системе. Из этого следует, что одни и те же устройства компьютера могут применяться для обработки и хранения как числовой информации, так и логических переменных. Элементарные схемы компьютера реализуют обработку сигнала по функциям И, ИЛИ, НЕ.

С помощью этого набора можно реализовать любую логическую функцию, описывающую работу устройств компьютера. Рассмотрим, к примеру, схему сложения одноразрядных двоичных чисел с учетом возможности переноса в старший разряд:

$$0+0=0;0+1=1;1+0=1;1+1=10_2$$

Если оба слагаемых равны единице, то сумма становится двузначной. В том разряде, в котором находились слагаемые, получается 0, а единица переходит в старший разряд. Введем обозначения: слагаемые – A и B, значение суммы в разряде слагаемых – S, перенос в старший разряд – P, перенос из младшего разряда – P_0,. Таблица истинности для значения, этого примера выглядит так:

Слагаемые		**Сумма в разряде слагаемых**	**Перенос в старший разряд**
A	**S**	**B**	**P**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Из этой таблицы видно, что значение переноса можно получить по операции конъюнкции: P = A•B. Значение суммы в трех первых строках получается по операции дизъюнкции, а последнее значение – это инверсия P, и всю таблицу можно описать функцией S = (A v B) • ⌉ (A•B). Эта функция описывает работу одноразрядного полусумматора процессора.

В полном одноразрядном сумматоре учитывается третье слагаемое: величина, переносимая в этот разряд из младшего разряда. Обозначим ее P_0. Таблица истинности с учетом этой величины выглядит следующим образом:

Слагаемые			**Сумма в разряде слагаемых**	**Перенос в старший разряд**
A	**B**	**P_0**	**S**	**P**
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

Из таблицы видно, что перенос в старший разряд надо делать тогда, когда значение 1 принимают хотя бы два слагаемых. Этому соответствует формула P = (A• B) v (B• P_0) v (A •P_0). Для значения суммы во всех строках, за исключением последней, подходит форму-

ла $S_1 = (A \vee B \vee P_0) \bullet P$. Для последней строки годится выражение $S_2 = (A \bullet B \bullet P_0)$. Окончательная формула, которая описывает результаты суммирования в одноразрядном сумматоре при всех возможных вариантах, выглядит так:

$$S = (A \vee B \vee P_0) \bullet \overline{P} \vee (A \bullet B \bullet P_0)$$

Так же как и в электрических схемах, для базовых типов логических преобразований установлены условные обозначения, которые облегчают понимание сложных логических схем. Они отражают тип преобразования информации, отвлекаясь от структуры электронной схемы, которая реализует это преобразование. На рис. 2.1 приведены условные обозначения этих преобразований.

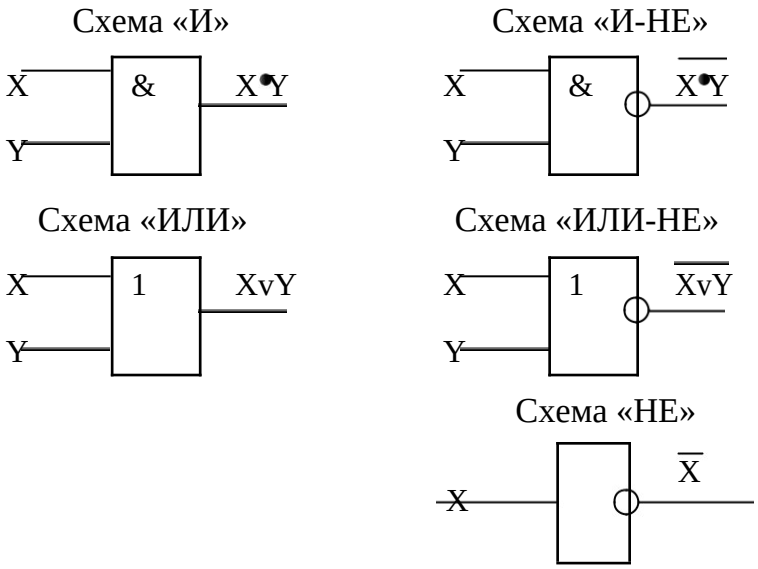


Рис. 2.1. Условные обозначения базовых логических элементов

На рис. 2.2 Приведена логическая схема триггера. Это устройство может хранить 1 бит информации. Триггеры используются как разряды оперативной памяти и памяти процессора. В обычном состоянии триггер хранит сигнал 0. Для записи 1 на вход S подается сигнал 1. Пройдя по схеме он формирует на выходе Q сигнал 1 и устойчиво хранит его после того, как сигнал S исчезнет. Для того, чтобы сбросить этот сигнал

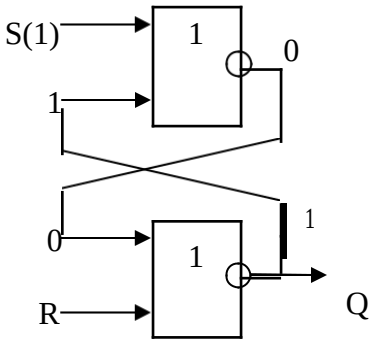


Рис. 2.2. Логическая схема триггера

и подготовиться к приему нового на вход R подается сигнал 1, который приводит триггер к «нулевому» состоянию.

2.4. Вопросы для самопроверки по теме 2

Задание № 1.Найдите среди заданных логических функций тождественно ложную:

1. АинеАилинеА;
2. АинеВинеА;
3. АинеАилиВ;
4. АинеВилиА.

Задание № 2.Укажите, какой логической операции она соответствует приведенная таблица истинности:

A	B	F
1	1	1
1	0	1
0	1	1
0	0	0

Задание №3. Укажите, каким высказыванием является утверждение:"2+3=4".

Задание № 4. Найдите среди заданных логических функций тождественно истинную:

1. АинеАилиВ;
2. А или не В или не А;
3. АинеАилинеА;
4. АинеВилиА.

Задание № 5. Определите, при значениях выполняется равенство Not A AND B = 1.

Задание № 6. Укажите, как называется логическая операция A∪B:

Задание № 7.Укажите, как обозначается логическое умножение для простых высказываний А и В:

Задание № 8. Представьте в символах логики высказывание «Если завтра будет дождь, то я возьму зонтик или никуда не пойду».

Задание № 9. Укажите название информации, представляемой с помощью слов естественного языка.

Задание № 10. Укажите название логической операции A B

1. инверсия
2. конъюнкция
3. дизъюнкция
4. импликация

Задание № 11. Укажите, какой логической операции соответствует приведенная таблица истинности:

1	0
0	1

3. ТЕХНИЧЕСКИЕ СРЕДСТВА РЕАЛИЗАЦИИ ИНФОРМАЦИОННЫХ ПРОЦЕССОВ

3.1. Этапы развития вычислительной техники

Историю совершенствования механизмов, облегчающих вычисления, можно разделить на три основных этапа:

1) ***механический***: регистрируются механические перемещения элементов конструкции. Так как при этом можно предусмотреть любое количество различимых состояний, конструкции этого этапа ориентированы на десятичную систему счисления. В истории развития этих механизмов можно выделить следующие этапы:

– *простейшие ручные приспособления* (период с IV тысячелетия до н.э). К ним относятся палочки, счѐты абак: глиняная пластинка с желобами, в которых определѐнным образом раскладывались камешки, русские счѐты: камешки нанизаны на проволоку;

– *вычислительные устройства*: арифмометры разных конструкций (с середины XVII века). Первый удобный для расчетов арифмометр создал Блез Паскаль в 1642 году. Его машина могла выполнять сложение и вычитание чисел с 6 – 8 разрядами и имела небольшие габариты. Следующий этап в принципиальном усовершенствовании арифмометров принадлежит Лейбницу. В 1673 году он представил машину, которая могла выполнять четыре арифметических действия;

– *автоматизация вычислений* – механические устройства, работающие по заданной программе. Идея разделения информации на команды и данные принадлежит Чарльзу Бэббиджу, который в 1822 году представил машину, которая могла рассчитывать таблицы не очень сложных функций.

В механических арифмометрах использовался принцип работы часового механизма: система взаимосвязанных зубчатых колес разного диаметра, в которой поворот каждого колеса на один зубчик соответствовал изменению на единицу определенного разряда числа.

2) ***Электромеханический –*** в счетных устройствах используются электромагнитные реле (первая половина XX века). Первая машина такого типа была построена немецким инженером Конрадом Цузе в 1941году. В 1943 году появились машины Марк-1, затем Марк-2, созданные американцем Говардом Эйкеном. Эти машины

выполняли арифметические операции с 23-значными десятичными числами и работали гораздо быстрее механических.

3) ***Электронный –*** регистрируются не механические смеще-ния, а состояния элементов конструкции. При этом оказалось удобнее всего использовать не десятичную, а двоичную систему счисления (включено/выключено, заряжено/разряжено, есть контакт/нет контакта). Первая машина такого типа, ENIAC (Electronic Numeral Integrator And Computer), была создана в США под руководством группы специалистов Говарда Эйкена, Дж. Моучли, П. Эккерта и введена в эксплуатацию 15.02.1945 г.

По элементной базе выделяют 5 поколений ЭВМ (периоды указаны условно):

– ***первое поколение*** – на электровакуумных лампах (1945–1955 г.г.);

– ***второе поколение*** – на транзисторах (1955-1965 г.г.);

– ***третье поколение*** – на микросхемах. Разрабатываются семейства машин с единой архитектурой, что приводит к программной совместимости, т. е. при появлении новой марки ЭВМ отпала необходимость переписывать заново все программы, которые были разработаны для предыдущей марки (1965–1970 г.г.);

– ***четвѐртое поколение*** – на интегральных схемах. Это существенно увеличило скорость работы, уменьшило энергоѐмкость, стоимость и габариты ЭВМ. Происходит переход к персональным ЭВМ. Создаются многопроцессорные и многомашинные комплексы (с 1970 г.);

– ***пятое поколение*** – суперкомпьютеры на больших интегральных схемах. Используются магнитные, лазерные, голографические принципы различения состояний. Машины этого поколения ориентированы на логическое программирование (обслуживание экспертных систем, плохо формализованных задач).

3.2. Принципы работы электронной вычислительной системы

Переход от механического принципа работы счѐтных элементов к электронному в развитии вычислительной техники ознаменовался резким увеличением быстродействия и объѐма памяти компьютеров. Это учѐл Джон фон Нейман в своѐм докладе при формулиров-

ке основных принципов, по которым должны функционировать и строиться компьютеры (1945 г.):

1. Возможность ввода программы и исходных данных в память компьютера.
2. Использование двоичного кода для внутримашинного представления команд и данных.
3. Считывание первой команды из ячейки памяти и организация еѐ выполнения.
4. Организация автоматического управления последовательностью выполнения последующих команд.

Для реализации этих принципов компьютер должен быть снабжѐн:

1. Внешним устройством для ввода/вывода информации.
2. Арифметико-логическим устройством для выполнения арифметических и логических операций.
3. Устройством управления для организации порядка выполнения программ.
4. Запоминающим устройством для хранения программ и данных.

Принципиальная схема взаимодействия перечисленных выше устройств представлена на рис. 3.1.

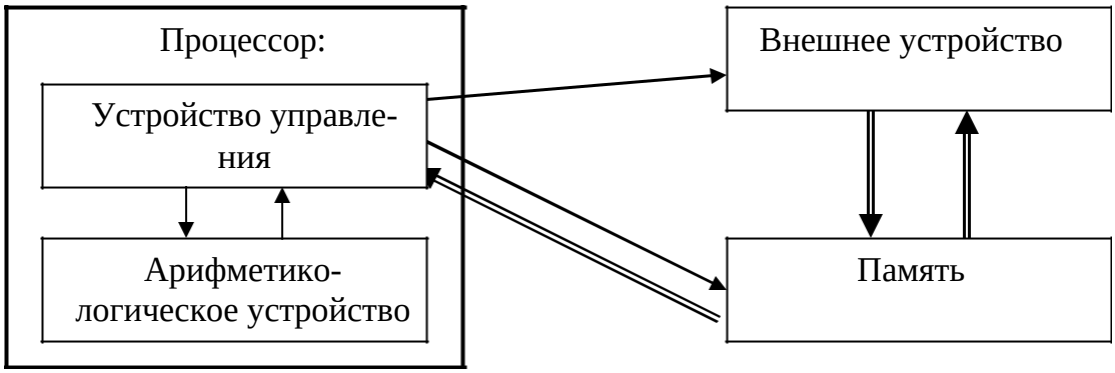


Рис. 3.1. Общая схема организации связей в компьютере (одинарные стрелки – пути и направления передачи управляющих сигналов, двойные – пути и направления движения информации).

Принципы, изложенные в докладе Джона фон Неймана используются в компьютерах всех поколений, конечно с детализацией способов реального воплощения указанных блоков и способов организации связей между ними.

3.3. Состав и назначение основных элементов персонального компьютера

Обобщающие термины

– *вычислительная техника*: совокупность устройств, предназначенных для автоматической или автоматизированной обработки данных;

– *вычислительная система*: набор взаимодействующих между собой устройств и программ, предназначенный для обслуживания одного рабочего участка;

– *архитектура ЭВМ*: описание принципов действия, информационных связей и взаимного соединения основных узлов компьютера.

– *конфигурация аппаратная / программная*: состав аппаратных / программных средств, входящих в вычислительную систему;

– *hardware*: совокупность аппаратных средств компьютера.

Виды внутренней памяти

– ***оперативная*** (ОЗУ): энерго**зависимая**. Тип памяти: RAM (Random Access Memory) – память прямого доступа. Это озна-чает, что время обращения к любой ячейке памяти одинаково;

– ***кэш-память***: энерго**зависимая**. Сверхоперативное ЗУ небольшого объѐма для обмена информацией между ОЗУ и микропроцессором. Компенсирует разницу в скорости работы этих устройств;

– ***постоянная память*** (ПЗУ): энерго**независимая**, предназначена для хранения BIOS (basic input/output system), программ запуска и остановки компьютера. Тип памяти ROM (Read Only Memory) – только для чтения;

– ***CMOS RAM***: энерго**независимая**, энергопотребление обеспечивается внутренней батарейкой. Используется для хранения информации о конфигурации и составе оборудования компьютера, а также о режимах его работы;

– ***flash memory***: энерго**независимая**, перепрограммируемая постоянная память;

– ***видеопамять*** (VRAM): энерго**зависимая**, предназначена для хранения закодированных изображений.

Виды внешней памяти (ВЗУ)

ВЗУ используют для длительного хранения программ и данных в двоичном коде. Все виды памяти этого назначения **энергонезависимые**.

– ***жѐсткий диск***, ***винчестер***, ***HDD***: Hard Disk Drive;

– ***гибкий диск***, ***Floppy Disk***: в настоящее время практически не используется;

– ***оптические диски***: CD-ROM (Compact Disk Read Only Memory) – одноразовая заводская запись, далее – только считывание. CD-R (Compact Disk Recordable) – одноразовая запись на компьютере, далее – только считывание. CD-RW (Compact Disk ReWriteable) – многократная перезапись на компьютере. Емкость от 700 Мб до 1,5 Гб;

– ***цифровые видеодиски***: DVD-ROM, DVD-R, DVD-RAM, DVD-RW. Емкость до 17 Гб;

– ***стримеры*** – накопители на магнитных лентах, используются для резервного копирования больших объѐмов информации.

Процессор

– ***центральный процессор*** содержит: УУ (устройство управления), АЛУ (арифметико-логическое устройство), регистры (элементы памяти), счѐтчик команд, кэш-память, математический сопроцессор. Крепится на материнской плате. Основные параметры: ***тактовая частота*** – влияет на скорость выполнения команд (от МГц до ГГц) и ***длина машинного слова*** – количество бит, которое может быть обработано за один такт (обычно кратно байтам)

– ***триггер***, ***разряд*** – электронная схема для хранения одной двоичной цифры;

– ***регистр*** – ячейка памяти процессора для кратковременного хранения данных или команды в процессе еѐ выполнения.

Системный блок

– ***материнская (системная) плата***: используется для крепления основных устройств компьютера (процессора, ОЗУ, ПЗУ, Кэш-память, интерфейсные схемы шин, гнѐзда расширений (слотов), обязательные системные средства ввода/вывода).

– ***шина*** (***магистраль***, ***системная шина***): включает в себя ***шину данных*** (разрядность 8, 16, 32, 64 бита), ***шину адресов*** (разрядность 16, 20, 24, 32, 36 битов) и ***шину управления***. Обеспечивает обмен информацией между процессором и устройствами компьютера.

Реализуется как пучок проводов, по каждому из которых передаѐтся 1 бит информации. Совокупность проводов, по которым передаются адреса ОЗУ, – ***адресная шина***, данных из этих адресов – ***шина данных***, сигналов, определяющих характер обмена информацией по магистрали (считывание, запись, синхронизация обмена информацией между устройствами и т.д.) – ***шина управления***.

– ***порты***: разъемы, с помощью которых к компьютеру подключаются внешние устройства. Они выведены на заднюю панель системного блока.

– ***последовательные порты*** (***COM*1**, ***COM*2**): передают электрические импульсы один за другим. К ним подключают мышь и внешний модем.

– ***параллельный порт*** (***LTP***): обеспечивает более высокую скорость передачи информации, чем последовательные, так как передает одновременно 8 электрических импульсов, к нему подключают принтер.

– ***USB-порты*** (Universal Serial BUS – универсальная последовательная шина): обеспечивают высокоскоростное подключение периферийных устройств: сканеров, цифровых камер.

– PS/2-порт: обычно подключается клавиатура.

Устройства ввода

Устройства ввода преобразуют информацию из естественной формы, доступной органам чувств человека, в двоичную форму:

– ***клавиатура***: для ввода текстовой информации;

– ***сканеры***: для текстовой и графической информации, зафиксированной на бумаге;

– ***дигитайзеры*** (графические планшеты): для ввода графической информации без промежуточной фиксации еѐ на бумаге (она рисуется на специальном планшете световым пером);

– ***манипулятор мышь***;

– ***манипулятор джойстик*** – наклоны ручки эквивалентны перемещению мыши, для игровых программ;

– ***манипулятор трекбол*** – встроенный в стационарный корпус шарик, прокрутка которого эквивалентна перемещению мыши. Используется в оптических мышках и ноутбуках;

– ***манипулятор пенмаус*** – похож на шариковую ручку, на конце которой находится узел, регистрирующий еѐ перемещения;

– ***цифровые видеокамеры и фотоаппараты***;

– ***микрофон***.

Устройства вывода

Устройства вывода преобразуют информацию из двоичной формы в естественную форму, доступную органам чувств человека:

– ***монитор***. Основные характеристики: тип экрана, размер экрана по диагонали (обычно в дюймах), разрешающая способность. Для жидкокристаллических мониторов – угол обзора.

– *адаптер*: устройство для соединения блоков компьютера с разными способами представления информации;

– ***контроллер***: устройство для сопряжения разных устройств и управления их работой.

– ***принтеры***: ***матричные***, ***струйные***, ***лазерные***. Различаются по способу печати;

– ***плоттеры*** (графопостроители): выводят документы больших размеров (чертежи, плакаты и т. п.);

– ***акустические стереоколонки***.

3.4. Вопросы для самопроверки по теме 3

Задание № 1. Укажите назначение системной шины (магистрали).
Задание № 2. Укажите устройства, размещаемые на материнской плате ПК.
Задание № 3. Укажите, к какому поколению относятся персональные компьютеры.
Задание № 4. Как называются электронные схемы для управления внешними устройствами?
Задание № 5. На каких видах внешней памяти невозможно случайно стереть информацию?
Задание № 6. По какому признаку классифицируются принтеры?
Задание № 7. Что обозначает термин «адаптер»?
Задание № 8. Укажите устройства, которые не входят в состав внутренней памяти современного компьютера.
Задание № 9. Укажите, какие из названий относятся к устройствам вывода данных:

1. плоттер;
2. процессор;
3. блок питания;
4. монитор;
5. сканер.

Задание № 10. Укажите устройства, которые входят в состав внутренней памяти современного компьютера.
Задание №11. Укажите, какие из названий относятся к устройствам ввода данных:

1. жѐсткий диск;
2. мышь;
3. привод CD-ROM;
4. джойстик;
5. регистры.

Задание № 12. Укажите устройство хранения данных, которое работает только при включѐнном питании:

1. ПЗУ;
2. гибкий магнитный диск;
3 ОЗУ;
4 жѐсткий диск.

Задание № 13. Укажите, какие из приведенных утверждений являются верными:

1. сетевая плата не является устройством приѐма-передачи данных
2. компакт-диск - это оперативная память
3. гибкий магнитный диск - это долговременная память
4. в мониторах на жидких кристаллах отсутствует электромагнитное излучение *Вопрос* № 14. Укажите наиболее важные характеристики жидкокри-сталлического монитора:

1. цвет фона окна;
2. физический размер экрана;
3. объѐм хранимых данных;
4. скорость обработки информации;
5. угол обзора.

4. ПРОГРАММНЫЕ СРЕДСТВА РЕАЛИЗАЦИИ ИНФОРМАЦИОННЫХ ПРОЦЕССОВ

Программное обеспечение ЭВМ (ПО) – это множество программ, которые используются или могут быть использованы на компьютере.

Совокупность всех программных средств компьютера и требующихся им данных обычно обозначают термином ***software*** . По типу функций, которые выполняют программы, выделяют три основных группы ПО: ***системное***, ***инструментальное***, ***прикладное***.

К нему также относят области деятельности, необходимые для проектирования и разработки программ разного типа:

– технологии проектирования программ (нисходящее, восходящее проектирование, структурное, объектно-ориентированное программирование и т. п.);

– методы отладки и тестирования программ;

– анализ качества работы программ;

– документирование программ и т. п.

4.1. Системное программное обеспечение ЭВМ

Системное ПО обеспечивает эффективную работу аппаратуры компьютера. Основные типы системных программ:

– ***операционная система*** (***ОС***): совокупность программ, тестирующих все устройства компьютера и управляющих их работой, а также организующих взаимодействие пользователя с компьютером (пользовательский интерфейс). Основные функции ОС:

– загрузка программ в оперативную память и управление ходом их выполнения;

– обмен данными между выполняющейся программой и внешними запоминающими устройствами;

– обслуживание нестандартных ситуаций в ходе выполнения программы;

– цикл работ по окончанию работы программы;

– организация хранения и поиска информации на внешних носителях;

– организация интерфейса пользователя;

– выполнение сервисных функций: форматирование дисков, копирование файлов и т. п.

Основные ОС: ***MS DOS*** – однопользовательская и однопрограммная ОС, в которой все команды по работе с файлами пользователь должен был набирать вручную в текстовом режиме; ***UNIX***, ***LINUX*** – используются на серверах сети Интернет; ***Windows NT/2000/XP/8*** – многопрограммные, многопользовательские, сетевые ОС; ***Android*** – портативная (сетевая) операционная система основанная на ядре Linux, применяемая в коммуникаторах, планшетных компьютерах, электронных книгах, цифровых проигрывателях, наручных часах, нетбуках и смартбуках;

– ***операционные оболочки***: улучшают пользовательский интерфейс, предусмотренный в ОС. Табличные оболочки (Norton Commander для MS DOS, Far Manager и Norton **Navigator** для Windows) представляют каталоги в виде текстовых таблиц и основные команды манипуляций с файлами можно не вводить вручную, а запускать функциональными клавишами. В графических оболочках для указания типа файла используются графические значки, для заказа типовых действий – специальные способы указания на них (щелчки мышью, горячие клавиши вместо ввода полного текста команд). Эти оболочки используются во всех версиях Windows;

– ***драйверы***: программы, которые расширяют возможности ОС по управлению работой устройств компьютера. Техническая реализация одного и того же типа устройства в разных моделях существенно различается. Для того, чтобы согласовать ее работу с остальными устройствами, для каждой модели одного и того же устройства разрабатывают специальную программу – драйвер. В процессе установки ОС определяет тип и конкретную модель каждого устройства компьютера и загружает соответствующий ему драйвер;

– ***утилиты***, (***сервисные программы***): небольшие программы, выполняющие действия обслуживающего характера и увеличивающие эффективность работы отдельных устройств, например, программы для диагностики различных устройств (Scandisk, Check Disk), программы оптимизации, «кэширования» и динамического сжатия дисков, и т. п.;

– ***архиваторы***: программы, которые переписывают файлы в сжимающем их коде, используются для подготовки файлов к пересылке по сети и для хранения дублирующих копий важной информации;

– ***антивирусные программы***: проверяют уже имеющиеся в компьютере и вновь поступающие в него программы на наличие действий, которые могут повредить ОС или файлы пользователя.

4.2. Файловая структура ОС. Операции с файлами

Файл – это именованная совокупность любых данных, размещѐнная на внешнем запоминающем устройстве и хранимая, обрабатываемая и перемещаемая как единое целое.

Совокупность правил и программ, по которым выполняются операции с файлами, называется ***файловой системой***. Основные из них: ***FAT*** (File Allocation Table) – используется в операционных системах для 16-и и 32-х разрядных процессоров, ***NTFS*** (New Technology File System) – используется, начиная с ОС Windows NT для 32-х и 64-х разрядных процессоров). В них пользователю предоставляются следующие возможности:

– создание папок;

– копирование, перемещение, переименование и удаление файлов и папок;

– навигация по файловой структуре

– запуск программ и открытие документов;

– создание ярлыков;

– стандарты для обозначения пути к файлу: абсолютный и относительный адрес файла. ***Абсолютный адрес*** начинается указанием диска, на котором расположен файл, и далее последовательно через символ «\» (обратный слеш) перечисляются все папки, которые следует открыть, чтобы найти нужный файл. В ***относительном адресе*** через символ «\» перечисление папок ведется, начиная от той, которая активна в данный момент. Для перехода в папку нижнего уровня следует указать имя этой папки. Переход в папку верхнего уровня вне зависимости от ее реального названия обозначается символом «..» (две точки).

Файл характеризуется свойствами и атрибутами. Их значения указываются в диалоговом окне «Свойства», которое открывается по команде «Свойства» в меню «Файл» и в контекстном меню. Основные из них:

– *имя файла* может содержать до 256 символов и состоит из двух частей, которые разделяются точкой. Первая часть – имя, которое назначает пользователь для того, чтобы в дальнейшем было по-

нятно, что за информация хранится в этом файле. Обычно имя пользователя состоит из букв русского и латинского алфавита, цифр, пробелов, дефиса. Остальные символы использовать не рекомендуется, хотя некоторые – допускается. Вторая часть – расширение. Оно указывает на тип файла или программу, которая в дальнейшем должна работать с ним. В нем разрешается использовать не более 4-х символов. Пояснения к стандартным расширениям, с которыми часто встречаются пользователи, приведены в табл.4.1.

– *длина файла* в байтах (длина занимаемого участка на диске);

– *время и дата* создания файла (для опознания последних по времени вариантов файла).

Кроме перечисленных выше атрибутов каждому файлу сопоставляются атрибуты, определяющие допустимые действия с ним. Например:

– *только для чтения* (исправления, сделанные во время просмотра, не будут сохраняться после закрытия файла;

– *скрытый* (не будет высвечиваться в каталоге), но открыть его можно, введя имя вручную;

– *архивный* (для автоматического обновления изменѐнных версий в архивах);

– *системный*.

Таблица 4.1.

Основные стандартные расширения файлов:

Расширение	Формат/назначение программы
Системные программы/программы в машинных кодах	
*.com *.pcx	системные программы в машинных кодах
*.sys	программа драйвера
*.bak	автоматически создаваемая резервная копия файла
*.tmp	временная копия с промежуточной информацией, автоматически уничтожаемая после закрытия программы
*.hlp	файл встроенной справочной системы
*.exe	прикладная программа в машинных кодах
	Графические форматы
*.bmp	формат растровой графики, в котором задан цвет каждого пиксе-ля
* .jpg	форматы сжатой растровой графики, удобный для хранения от-сканированных фотографий и иллюстраций. Используется в Ин-тернете
*.gif *.tif *.png	Формат для диаграмм, создаваемых программным путѐм, рисун-ков типа аппликации с ограниченным набором цветов, несложной анимации. Используется в Интернете
*.psd, *.pdd	формат растровых графических файлов в PhotoShop
*.cdr	формат векторных графических файлов в CorelDRow
*.eps	формат векторных графических файлов для подготовки афиш, объявлений, плакатов, который рекомендуется для создания ил-люстраций в настольных издательских системах.
Текстовые форматы	
*.txt	формат текстовых файлов без форматирования
*.doc, *.docx	формат текстовых файлов программ Word до 2003 и после 2003 года
*.rtf	универсальный формат форматированных текстовых файлов, понятный всем версиям программы Word
*.pdf	Комбинация текста с растровой графикой. Используется в Ин-тернете для публикации документов.
Расширения основных программ Microsoft office	
*.xls, *.xlsx	форматы файлов программы Excel до 2003 и после 2003 года
*.ppt, *.pptx	формат файлов электронных презентаций программы PowerPoint до 2003 и после 2003 года
*.mdb, *.mdbx	формат файлов программы Access до 2003 и после 2003 года
*.htm, *.html	формат файлов для Web-страниц

Для того, чтобы применить нужную команду не к одному файлу, а к нескольким, в именах которых есть общие для них фрагменты, вводится понятие ***групповое имя файла (маска имени)***. В маске в явном виде указываются символы, которые обязательно должны присутствовать в нужных файлах, остальные символы заменяются шаблонными:

– * означает, что на этом месте может оказаться любое количество любых символов, в частности, ничего;

– ? означает, что на этом обязательно должен быть какой-нибудь символ, но только один;

– | означает, что требуется обработать одной командой все файлы, указанные в предыдущей маске, кроме файлов, подходящих под маску, указанную после него.

Примеры:

. – под эту маску подходят все файлы (любое имя и любое расширение);

*.jpg – маска указывает на все файлы рисунков с расширением —jpg‖;

экзам.* – маска указывает на файлы с любым расширением, у которых в имени пользователя есть указанный фрагмент текста;

.|*.bak‖ – означает, что требуется обработать одной командой все файлы, кроме файлов с расширением . —bak‖, какими бы ни были их имена.

4.3. Инструментальное программное обеспечение ЭВМ

Инструментальное ПО служит для разработки программ, применяемых в самых разных областях деятельности человека.

В истории программирования можно выделить следующие этапы по способам написания программ:

– программирование в машинных кодах;

– использование машинно-ориентированных языков низкого уровня;

– использование алгоритмических языков высокого уровня.

Для того чтобы ЭВМ могла выполнять программу, она должна быть записана по строгим правилам в виде, доступном процессору, т. е. представлять собой последовательность двоичных чисел и кодов. Такие коды называются ***машинными кодами, машинными командами***. Программа, написанная таким образом, – ***программой на ма-***

шинном языке или ***исполняемым модулем***.

Первоначально, когда появились ЭВМ, программисты для разработки программ использовали машинный язык. Это было очень трудно и неудобно, так как приходилось самому распределять память под команды программы и данные, держать в памяти массу абстрактных двоичных кодов, обозначающих адреса данных и команд, которые их обрабатывают.

На втором этапе для облегчения программирования была создана программа, которая автоматически заменяла в программах удобные человеку названия переменных и операций на машинные команды. Эта процедура была обозначена термином ***трансляция***, а программа, выполняющая ее, ***транслятором***. На этом этапе трансляторы могли переводить в машинные коды конструкции, которые заменялись одной или несколькими (условно не больше пяти) машинными командами. Для совокупности правил, по которым следует записывать программу, чтобы ее мог обработать такой транслятор, используют термины ***ассемблер***, ***макроассемблер***.

Ассемблер – это машинно-ориентированный язык низкого уровня. Алгоритм, по которому строится программа, приходится дробить на очень мелкие шаги, которые умеет выполнять процессор. При появлении новых моделей процессоров с иной структурой команд и встроенных операций приходится создавать новые версии ассемблера, трансляторов с него и переписывать под них пользовательские программы.

Следующий этап развития инструментального ПО связан с появлением ***алгоритмических языков***, каждый из которых содержит конструкции, удобные для записи алгоритмов в той или иной сфере человеческой деятельности. Алгоритмические языки относятся к классу языков высокого уровня. Они позволяют при составлении программы дробить алгоритм на более крупные смысловые блоки, не связанные со структурой и типом команд ЭВМ, на которой будет работать программа. При изменении модели процессора переписывается только транслятор с алгоритмического языка, а пользовательские программы остаются прежними.

4.4. Основные понятия алгоритмических языков. Алфавит. Синтаксис. Семантика

Обычный разговорный язык состоит из четырех основных

элементов: символов, слов, словосочетаний и предложений. Алгоритмический язык содержит подобные элементы, только слова называют *элементарными конструкциями*, словосочетания – *выражениями*, предложения – *операторами*. Алгоритмический язык (как и любой другой язык), образуют три его составляющие: алфавит, синтаксис и семантика.

Алфавит – фиксированный для данного языка набор символов (букв, цифр, специальных знаков и т.д.), которые могут быть исполь-зованы при написании программы.

Синтаксис – правила использования символов алфавита в специальных конструкциях, с помощью которых составляется алгоритм.

Семантика – система правил толкования конструкций языка.

Таким образом, программа составляется с помощью соединения символов алфавита в соответствии с синтаксическими правилами и с учетом правил семантики.

Программа, написанная на алгоритмическом языке, называется ***исходным модулем***.

Для ее трансляции можно использовать два типа технологий: ***интерпретацию*** или ***компиляцию***. Трансляторы, реализующие эти технологии, называются соответственно ***интерпретаторами*** и ***компиляторами***.

При интерпретации транслятор переводит в машинные коды очередную строку исходного модуля, сразу же выполняет еѐ и переходит к обработке следующей строки, не создавая законченного исполняемого модуля всей программы в целом. Такой тип трансляции удобен на этапе отладки или тестирования программы.

При компиляции трансляция разбивается на два этапа (рис. 4.1). На первом компилятор создает ***объектный модуль*** – промежуточную программу, в которой конструкции исходного модуля переведены в машинные коды, но вместо реальных адресов данных и работающих с нею подпрограмм, используются относительные адреса и не добавлены служебные подпрограммы, необходимые для еѐ работы.

На втором этапе программа-сборщик заменяет в объектном модуле относительные адреса на абсолютные и подключает к нему необходимые для работы стандартные программы.

В результате получается ***исполняемый модуль*** – программа, написанная в машинных кодах и полностью готовая к работе.

Рис. 4.1. Схема трансляции в режиме компиляции

Этот режим трансляции удобен для расчѐтов по уже отлаженной и протестированной программе, так как исполняемый модуль можно сохранить в памяти и при необходимости провести расчеты с другим набором исходных данных, не повторяя трансляцию.

В настоящее время для большинства алгоритмических языков разработаны ***интегрированные системы программирования*** (Бейсик, Паскаль, Си и т. п.). Они включают в себя следующие компоненты:

– текстовый редактор для написания программы;
– библиотеки стандартных программ;
– средства отладки;
– справочную службу;
–компилятор и/или интерпретатор;
– диалоговый интерфейс.

4.5. Основные алгоритмические языки высокого уровня

Один из первых языков программирования высокого уровня, ***Фортран*** (**For**mula **Tran**slation), был создан в середине 50-х годов. Он используется ***для инженерных и научных расчетов***, для решения задач физики и других наук с развитым математическим аппаратом. Благодаря своей простоте и тому, что на этом языке накоплены

большие библиотеки программ, Фортран и в наши дни остается одним из самых распространенных алгоритмических языков.

Для решения экономических задач был создан язык программирования ***Кобол***.

Расширение областей применения ЭВМ влечет за собой создание языков, ориентированных на новые сферы применения: ***Снобол*** – алгоритмический язык ***для обработки текстовой информации. Лисп*** – алгоритмический язык для обработки символов. Он находит широкое применение в исследованиях ***по созданию искусственного интеллекта***.

В 1968 г. был объявлен конкурс на лучший язык программирования для обучения студентов. Для этого конкурса ***Никлаус Вирт создал язык Паскаль***, достаточно простой, удобный, с наличием мощных средств структурирования данных. Хотя Паскаль был разработан как язык для обучения программированию, он впоследствии получил широкое развитие и в настоящее время считается одним из самых используемых языков.

Для обучения младших школьников Самуэлем Пайпертом был разработан ***язык Лого***. Он отличается простотой и богатыми возможностями.

Широкое распространение в школах в качестве обучающего языка получил язык ***Бейсик***, позволяющий взаимодействовать с ЭВМ в режиме непосредственного диалога. Спустя много лет после изобретения Бейсика, он и сегодня самый простой для освоения из десятков языков ***общецелевого программирования***.

Необходимость разработки больших программ, управляющих работой ЭВМ, потребовала создания специального языка программирования. В начале 70-х г. был разработан язык ***СИ***, который широко используется как ***инструментальный язык для разработки операционных систем, трансляторов, баз данных и других системных и прикладных программ***. Это язык программирования общего назначения. Во многих случаях программы, написанные на Си, сравнимы по скорости с программами, написанными на языке Ассемблера. При этом они имеют лучшую наглядность и их более просто сопровождать. В отличие от Паскаля, в нем заложены возможности непосредственного обращения к некоторым машинным командам и к определенным участкам памяти компьютера.

Появление функционального программирования привело к

созданию языка ***Пролог*** (ПРОграммы ЛОГические)**.** Этот язык программирования разрабатывался ***для задач анализа и понимания естественных языков на основе языка формальной логики и методов автоматического доказательства теорем***.

В 80-х г. 20 века был создан язык ***Ада***. В дополнение к классическим свойствам, он ***обеспечивает программирование задач реального времени и моделирования параллельного решения задач.***

К языкам ***сверхвысокого уровня*** можно отнести ***Алгол-68*** **и** ***APL.*** Повышение уровня этих языков произошло за счет введения сверхмощных операций и операторов.

В современной информатике можно выделить два основных направления развития языков программирования: процедурное и непроцедурное.

В ***процедурных*** языках программа явно описывает определенную последовательность действий, которые необходимо выполнить, чтобы получить результат.

Среди процедурных языков выделяют в свою очередь ***структурные*** и ***операционные языки***. В структурных языках одним оператором записываются целые алгоритмические структуры: ветвления, циклы и т.д. В операционных языках для этого используются несколько операций. Широко распространены следующие структурные языки: ***Паскаль***, ***Си***, ***Ада***, ***ПЛ/1***. Среди операционных известны ***Фортран***, ***Бейсик***, ***Фокал***.

Непроцедурное (***декларативное***) программирование появилось в начале 70-х годов 20 века, но стремительное его развитие началось в 80-е годы, когда был разработан японский проект создания ЭВМ пятого поколения, целью которого явилась подготовка почвы для создания интеллектуальных машин. К непроцедурному программированию относятся ***функциональные*** и ***логические*** языки.

В ***функциональных языках*** программа описывает вычисление некоторой функции. Обычно эта функция задается как композиция других, более простых, те в свою очередь разлагаются на еще более простые и т.д. Один из основных элементов в функциональных языках – ***рекурсия***, то есть вычисление значения функции через значение этой же функции от других элементов. Присваивания и циклов в классических функциональных языках нет.

В ***логических языках*** программа вообще не описывает действий. Она задает данные и соотношения между ними. После этого сис-

теме можно задавать вопросы. Машина перебирает известные и заданные в программе данные и находит ответ на вопрос. Порядок перебора не описывается в программе, а неявно задается самим языком. Классическим языком логического программирования считается ***Пролог***. Построение логической программы вообще не требует алгоритмического мышления, программа описывает статические отношения объектов, а динамика находится в механизме перебора и скрыта от программиста.

Можно выделить еще один класс языков программирования – ***объектно-ориентированные языки высокого уровня.*** На таких языках не описывают подробной последовательности действий для решения задачи, хотя они содержат элементы процедурного программирования. Объектно-ориентированные языки, благодаря богатому пользовательскому интерфейсу, предлагают человеку решить задачу в удобной для него форме. Примером такого языка может служить язык программирования визуального общения ***Object Pascal***.

Языки описания сценариев, такие как ***Perl***, ***Python***, ***Rexx***, ***Tcl*** и языки оболочек ***UNIX***, предназначаются не для написания приложения с нуля, а для комбинирования компонентов, набор которых создается заранее при помощи других языков. Развитие и рост популярности Internet также способствовали распространению языков описания сценариев. Так, для написания сценариев широко употребляется язык Perl, а среди разработчиков Web-страниц популярен ***JavaScript***.

4.6. Прикладное программное обеспечение ЭВМ

Прикладное ПО – это программы, предназначенные для решения индивидуальных задач пользователя или классов задач в конкретной области применения информационных технологий (***проблемной области***). Программы этого типа можно разбить на три группы.

Индивидуальное прикладное ПО:

– игровые и развлекательные пакеты;

Программы, разрабатываемые для отдельного пользователя или организации.

Стандартное прикладное ПО:

– текстовые редакторы и процессоры;

– графические редакторы;

– программы электронных презентаций;

– электронные таблицы;
– системы управления базами данных;
– бухгалтерские и финансовые пакеты;
– системы автоматизированного проектирования;
– издательские системы;
– системы документооборота;
– программы-переводчики;
–поддержка электронной почты;
– образовательные, обучающие программы, мультимедийные энциклопедии;
– мультимедийные программы для воспроизводства, создания и редактирования звуко- и видеозаписей;
–ит.п.

Интегрированные пакеты программ:

– MS Office, Open Office, Corel Word Perfect Office, Star Office и т. п. Такие пакеты представляют собой совокупность разных стандартных прикладных программ, охватывающих все типы деятельности в той предметной области, для которой он создан, обладающих однотипным интерфейсом и средствами передачи информации между различными компонентами пакета. Пример: интегрированный пакет «издательская система» должен содержать:
– текстовый редактор;
– орфографический корректор;
– программу слияния текстов;
– программу формирования оглавлений и составления указателей;
– автоматический поиск и замену слов и фраз;
– средства телекоммуникаций;
– электронную таблицу;
– СУБД;
– модули графического оформления;
– графический редактор;
– набор разных шрифтов;
–ит.п.

4.7. Общие сведения о графических редакторах

В настоящее время существует несколько технологий создания рисунка в ЭВМ:

– растровая;

– векторная;

– фрактальная;

– flash-графика.

Характеристики растровой графики. Изображение передаèтся последовательной записью цвета всех пикселей рисунка. Растровыми являются все сканированные изображения и цифровая фотография. *Основные редакторы*: Paint, Adobe Photoshop, FotoEditor. *Основные форматы*: bmp, gif, jpg, png, tif, pdf – описаны в п. 3.2 табл. 3.1.

Характерные особенности растровой графики:

– самое точное воспроизведение цвета;

– в пределах одного замкнутого контура можно создать только один тип заливки, но в дальнейшем его можно корректировать;

– при увеличении изображения появляется лестничный эффект;

– при уменьшении изображения может теряться чѐткость и количество цветов;

– можно корректировать часть контура, создавать любые композиции из шаблонных и индивидуальных контуров, которые после их закрепления уже нельзя разложить на составляющие;

– возможна трансформация (растянуть, сжать, отразить, увеличить, уменьшить, сдвинуть, скрутить и т. д.) только прямоугольного выделенного фрагмента изображения;

– можно использовать фильтры для изменения тональности изображения и достижения различных спецэффектов (туман, морская рябь, просмотр через мокрое стекло и т. п.);

– можно вводить в рисунки тексты.

Характеристики векторной графики. Изображение составляется, как мозаика, из отдельных ***графических примитивов*** (***автофигур***), которые строятся на указанном месте с помощью формул. Такая графика удобна для создания диаграмм, блок-схем, *Основные редакторы*: графика MS Word, Adobe Illustrator, Corel Draw. 3DStudioMax, 3D Canvas, Lightwave, Maya Большинство векторных редакторов могут работать и с растровой графикой. *Основные форматы*: wmf, cdr, ps, eps – описаны в п. 3.2 табл. 3.1.

Характерные особенности векторной графики:

– файлы рисунков имеют существенно меньший объем по сравнению с аналогичными файлами растровой графики;

– допускается масштабирование рисунка без искажений;

– объединение и разделение ранее объединѐнных фигур на исходные графические примитивы;

– деформирование, перемещение как отдельных графических примитивов, так и их объединѐнных блоков;

– команды форматирования действуют на фигуру в целом, а не на отдельные еѐ части;

– – в пределах одного замкнутого контура можно создать только один тип заливки, но в дальнейшем его можно корректировать;

– нельзя корректировать часть графического примитива;

– можно использовать эффекты тени для любых и объѐма – для замкнутых контуров;

– можно вводить в рисунки тексты.

Разновидность векторной графики – трѐхмерная графика.

Характеристики фрактальной графики. Удобна для визуализации моделей всевозможных трѐхмерных объектов, природных ландшафтов и т. п. Формирование изображений целиком основано на формулах и уравнениях, описывающих эти объекты. *Основной редактор*: Brass.

Характеристики flash-графики. Используется для высококачественных анимационных изображений на Web-страницах и электронной рекламы. Позволяет кодировать качественную анимацию в небольших по размерам файлах. *Основной редактор*: Macromedia Flash.

Помимо редакторов, обеспечивающих полный спектр работ с графикой, широко используются ***программы-вьюверы*** (***просмотрщики***),которые предназначены только для просмотра и печати ранее созданных изображений. К ним относятся, например, Adob Acrobat Reader (просмотр графики и текстов в формате PDF), GSview (просмотр графики и текстов в формате PS и EPS)

Вопросы для самопроверки по теме 4

Задание № 1. Укажите, какой из перечисленных терминов обозначает программы, обеспечивающие взаимодействие ОС с периферийными устройствами:

1. контроллер;
2. транслятор;
3. драйвер;
4. компилятор.

Задание № 2. Укажите, как называется именованная область внешней памяти произвольной длины с определѐнным количеством информации.

Задание № 3. Укажите, что не входит в основные функции операционной системы:

1. обеспечение диалога с пользователем;
2. разработка программ для ЭВМ;
3. управление ресурсами компьютера;
4. организация файловой структур.

Задание № 4. Размер кластера 512 байт, размер файла – 816 байт. Укажите, сколько места на диске займет этот файл.

Задание № 5. Укажите, что определяет расширение файла:

1. размер;
2. имя;
3. тип;
4. расположение.

Задание № 6. Укажите, что относится к основным компонентам системного программного обеспечения:

1. обрабатывающие программы и система автоматизации программирования
2. операционная система и система программирования
3. монитор и супервизор
4. пакеты прикладных программ

Задание №7. Укажите, какая группа файлов будет выделена по маске <*.*|*.bak>.

Задание № 8. Перечислите маски для текстовых файлов.

Задание № 9. Укажите обобщенное название программ для согласования работы внешних и внутренних устройств компьютера

Задание № 10. Укажите наиболее известные способы представления графической информации в компьютере.

Задание № 11. Укажите графический формат, приводящий при сохранении фотографий к наименьшему объѐму файла.

Задание № 12. Укажите этапы трансляции, при которой создается исполняемый файл.

Задание № 13. Укажите, к какому классу языков программирования относится Ассемблер

Задание № 14. Укажите язык наиболее удобный для системного программирования

Задание № 15. Укажите язык наиболее удобный для логического программирования

Задание № 16. Укажите неверные утверждения для растрового графического редактора:

1. можно рисовать с помощью манипулятора мышь линии произвольной формы;
2. нельзя сохранять рисунки на внешних носителях;
3. нельзя масштабировать фрагменты изображения;
4. возможна тональная коррекция изображения.

Задание № 17. Укажите неверные утверждения для векторного графического редактора:

1. можно применять разные варианты заливки;
2. при увеличении рисунка может появиться лестничный эффект;
3. можно сохранять рисунок в различных графических форматах;
4. можно использовать пересечение объектов как отдельный графический примитив.

Задание № 18. Укажите верные утверждения для векторного графического редактора:

1. можно формировать разную заливку одного объекта
2. можно объединять графические объекты
3. нельзя сохранять рисунок на внешнем носителе
4. возможно удаление части графического примитива

Задание № 19. Укажите системы программирования среди перечисленных названий:

1. Adobe PhotoShop
2. Visual FoxPro
3. Visual C++
4. Borland Delphi

Задание № 20. Укажите основные особенности трансляции в режиме интерпретации.

5. МОДЕЛИ РЕШЕНИЯ ФУНКЦИОНАЛЬНЫХ И ВЫЧИСЛИТЕЛЬНЫХ ЗАДАЧ

5.1. Основные понятия моделирования

Модель – упрощѐнное подобие реального объекта, процесса или явления, которое отражает его существенные особенности.

Сущность – обобщѐнное название объекта, явления или процесса, которое изучается с помощью моделирования.

Атрибуты (параметры) – характеристики сущности, которые учитываются в еѐ модели.

Моделирование – метод познания, состоящий в создании и исследовании моделей изучаемых сущностей.

Каждой сущности можно сопоставить несколько моделей в зависимости от того, для какой цели она создаѐтся. Пример – возможные модели человека. Для отдела кадров на работе – это его анкета или резюме, в которых учитываются атрибуты, необходимые в профессиональной деятельности; для поликлиники – медицинская карта, в которой учитываются атрибуты здоровья; для приятелей – черты характера и набор его хобби; для портного – геометрические размеры тела.

Необходимость создания моделей диктуется следующими факторами:

– исследования на оригинале может быть экономически невыгодным;

– изучение может приводить к разрушению сущности (моделирование взрывов, методики лечения, хранения продуктов и т. п.);

– оригинала нет в действительности (изучение сущностей прошлого или будущего);

– необходимо исследование только некоторых свойств оригинала.

5.2. Классификации моделей

В зависимости от того, какой фактор является наиболее важным при моделировании, для классификации моделей используют разные признаки:

– по области использования;

– по фактору времени;

– по отрасли знаний;

– по форме представления.

Классификация моделей по области использования

Учебные – используются при обучении.

Опытные – это уменьшенные или увеличенные копии проектируемого объекта. Используют для исследования и прогнозирования его будущих характеристик (аэродинамическая труба).

Научно-технические – для исследования процессов и явлений.

Игровые – репетиция поведения объекта в различных условиях.

Имитационные – отражение реальности в той или иной степени (это метод проб и ошибок).

Классификация моделей по фактору времени

Статические – описывают состояние системы в определенный момент времени (единовременный срез информации по данному объекту). Примеры: классификация животных, строение молекул, список посаженных деревьев, отчет об обследовании состояния зубов ит.д.

Динамические – описывают процессы изменения и развития системы (изменения объекта во времени). Примеры: моделирование движения тел, развития организмов, процесс химических реакций.

Классификация моделей по отрасли знаний – это классификация по отрасли деятельности человека (математические, биологические, химические, социальные, экономические, исторические и т. д.).

Классификация моделей по форме представления

– материальные (***предметные***, ***физические***) – это модели, которые имеют реальное воплощение и отражают внешние свойства или внутреннее устройство моделируемых сущностей, суть процессов и явлений в объекте-оригинале. Материальное моделирование – это экспериментальный метод познания окружающей среды. Примеры: детские игрушки, скелет человека, чучело, макет солнечной системы, школьные пособия, физические и химические опыты, авиамодель истребителя, полоса препятствий.

– информационные – это целенаправленно отобранная информация о моделируемой сущности, которая отражает ее свойства, наиболее существенные для исследователя. В информационных мо-

делях реальный объект или процесс заменяется его формальным описанием. Такая процедура называется ***формализацией.***

Например, информационной моделью движения поездов является расписание их движения, а материальной – макет железной дороги с движущимися паровозиками.

По уровню формализации различают:

– хорошо формализованные модели. Их можно решить средствами, принятыми в данной предметной области, не используя субъективные мнения экспертов;

– плохо формализованные модели. Их нельзя решить без привлечения эксперта в данной предметной области.

Типы информационных моделей

Абстрактные (***мысленные***) – при построении модели используются понятия, не существующие в реальной жизни. Примеры: модель идеального газа. Она представляет каждую молекулу как материальную точку, т. е. объект, который имеет массу, но не имеет размеров. В модели движения планет вокруг солнца каждая планета тоже представляется как материальная точка.

Вербальные – мысленные модели, выраженные в разговорной форме с помощью естественных языков. Пример: инструкция пилоту самолѐта – это вербальная неформализованная модель, так как она пишется на естественном языке.

знаковые (***формализованные)*** – выражены специальными символами, применяемыми в изучаемой предметной области. Например, компьютерная модель реализована средствами программной среды, математическая – формулами, которые описывают изучаемую сущность. Знаковая формализованная модель музыкального произведения – запись с помощью нот и т. д.

В знаковых информационных моделях выделяют класс ***образно-знаковых*** моделей. Например, к таким моделям относятся:

– ***Геометрические*** – рисунок, пиктограмма, чертеж, карта, план, объемное изображение;

– ***Структурные*** – таблица, граф, схема, диаграмма;

– ***Алгоритмические*** – нумерованный список действий, пошаговое перечисление, блок-схема.

По способу организации данных информационные модели делятся на:

– *реляционные* (***табличные***): перечень объектов и их свойств оформляется в виде связанных между собою таблиц. Каждая строка таблицы содержит информацию об одном экземпляре (***сущности***) предметной области, каждый столбец – значения одной и той же характеристики (***атрибута***) для разных сущностей. Пример: расписание движения поездов – это табличная информационная модель реального перемещения поездов по железной дороге;

– *иерархические*: объекты распределены по уровням. Каждый элемент высокого уровня состоит из элементов нижнего уровня, а элемент нижнего уровня может входить в состав только одного элемента более высокого уровня.

Такие модели представляются ориентированным графом («деревом»), у которого начальная вершина не подчинена никакой другой, а все остальные подчинены только одной, но могут иметь в своèм подчинении сколько угодно объектов нижнего уровня. Если из каждого узла выходит только два потомка, то такая структура называется бинарным деревом.

Примеры: файловая структура в компьютере (система каталогов), система доменных имèн в Интернете, структура почтовых адресов, классификации животных, растений. В иерархической модели две любые вершины могут быть соединены только одним путèм. Пример: относительный путь к файлу имеет только один вариант.

– *Сетевые*: между объектами моделируемой системы существуют множественные связи. Такие модели представляются графом, в котором имеются связи между вершинами, позволяющие создать разные пути перехода между ними.

Примеры: модель функционирования Интернет, где каждый сервер может связаться с любым другим сервером через цепочку промежуточных узлов, и эти цепочки могут быть разными; модель взаимодействия пациентов и врачей в больнице, где каждого больного обследует несколько врачей и в то же время каждый врач следит за здоровьем нескольких больных; модель взаимодействия студентов и преподавателей в процессе обучения.

5.3. Базы данных и базы знаний

В наше время любой специалист независимо от сферы деятельности в той или иной мере занимается сбором, накоплением, сортировкой разнообразных данных и прочими операциями их обработ-

ки. При структурировании данных по реляционному типу в состав базы обычно входят следующие средства: таблицы, запросы, формы, отчеты, макросы и модули.

Таблицы – в них накапливается информация, которая вводится с клавиатуры.

Запросы – программы, которые манипулируют данными из таблиц и в черновом виде подготавливают ответы на типовые вопросы пользователей.

Формы – программы, размещающие на экране нужную пользователю информацию. Формы используются для вывода тех сведений, которые достаточно просмотреть с экрана и на принтер выводить не нужно.

Отчеты – программы, подготавливающие информацию для вывода на принтер.

Макросы и модули – программы, в которых производится такая обработка информации, которая недоступна каждому из перечислен-ных выше средств по отдельности.

Наряду с базами данных в настоящее время при моделировании плохо формализованных задач используют базы знаний. Базы знаний в отличие от баз данных содержат в себе не только фактическую информацию, которая является постоянной для данной предметной области, но и правила вывода, допускающие автоматические умозаключения о вновь вводимых фактах, то есть моделируют «осмысленную» обработку информации с помощью правил логики. Компьютерные модели таких задач называются ***интеллектуальными системами***. Математические методы, по которым работают интеллектуальные системы, называются ***методами искусственного интеллекта***. В их основе лежат эвристические приѐмы. ***Эвристика*** – это неформализованная процедура, сокращающая количество шагов поиска решения за счет некоторого способа направленного поиска решения, а не простого перебора всех возможных вариантов. Часто считается, что база знаний отличается от базы данных именно нали-чием механизма вывода новых знаний из старых. Еще одна особен-ность, которая закладывается в базы знаний, это выдача системой «объяснения» хода ее рассуждений при поиске ответа.

Наиболее известный класс интеллектуальных систем — это экспертные системы (ЭС). Эти системы рассматриваются как модели

поведения экспертов в определенной области знаний. База знаний ЭС создается при помощи трех групп специалистов:

– эксперты той проблемной области, к которой относятся задачи, решаемые ЭС;

– инженеры по знаниям, являющиеся специалистами по разработке интеллектуальной информационной системы (ИИС);

– программисты, осуществляющие реализацию ЭС.

ЭС может функционировать в 2-х режимах: ввод фактов по данной предметной области или проведение консультаций. В режиме ввода знаний эксперт с помощью инженера по знаниям вводит известные ему сведения о *предметной области* в базу знаний ЭС. В режиме консультации пользователь ведет диалог с ЭС, сообщая ей сведения о *текущей задаче* и получает рекомендации ЭС, сопровождающиеся «объяснением хода ее рассуждений». Например, в сфере тушения лесных пожаров на основе информации о пожаре (описание растительности в данной области, сводка погоды и т. п.), выдаѐтся некая рекомендация относительно стратегии борьбы с данным пожаром (оценка требуемых ресурсов, рекомендации по размещению техники и т. п.)

5.4. Этапы моделирования

Создание компьютерной модели разбивается на несколько этапов:

1. ***Постановка задачи***: описание сущности, подлежащей моделированию, цели исследования и той информации о моделируемом объекте, которую необходимо учитывать в связи с поставленной целью.

2. ***Разработка модели***: выбор типа создаваемой модели (предметная, компьютерная, математическая и т. п.) и ее реализация. Если выбран компьютерный тип модели, создается соответствующая программа.

3. ***Тестирование модели***: проведение компьютерного или натурного эксперимента на тестах. ***Тест*** – набор данных, для которых заранее известен результат.

Для компьютерных программ применяются разные технологии тестирования. Если тестируется программа, которая получена путем внесения изменений, улучшающих ранее работавшую программу, то, чтобы убедиться, что внесенные изменения не испортили ее, в пер-

вую очередь ее проверяют на тех тестах, по которым проверяли предыдущую программу, Такое тестирование называется ***регрессионным***. Тестирование новых сложных программ разбивается на два этапа: ***альфа-тестирование*** – это проверка программы на тестах, которые составил ее разработчик. ***Бета-тестирование*** – передача программы заказчикам в бесплатное пользование с условием, что при возникновении ошибок в работе разработчик будет уведомлен об этом.

4. ***Анализ результатов моделирования***: процесс проверки правильности модели на совокупности тестов, охватывающих все диапазоны моделируемых параметров. Если результаты соответствуют цели, построение модели закончено, и можно использовать модель для прогнозирования реальных результатов. В противном случае модель уточняется за счет введения в нее новых данных, не учтенных на первом этапе или изменяется ее реализация.

5.5. Вопросы для самопроверки по теме 5

Задание № 1. Укажите отличие модели от реального объекта, явления или процесса.
Задание № 2. Процесс описания объекта на искусственном языке называется ___.
Задание № 3. В основе методов искусственного интеллекта лежат ___.
Задание № 4. Отличительной чертой интеллектуальных систем является ___.
Задание № 5. Можно ли отнести к плохо формализованным задачам задачи, приводящие к трудоѐмким расчѐтам по известным формулам? *Задание* № 6. Какие методы используют для плохо формализованных задач?
Задание № 7. К какому классу моделей относится рецепт кушанья?
Задание № 8. Модель гравитационного взаимодействия двух тел, записанная в виде формул, это ___.
Задание № 9. Моделью Земли для определения законов еѐ движения вокруг Солнца является: ___.
Задание № 10.Укажите этап, на котором осуществляется определение целей моделирования.
Задание № 11. Термины иерархическая, сетевая, реляционная – это ___.

Задание № 12. Дана таблица моделирования:

1	моделируемый процесс	A	ракета
2	моделируемый объект	B	исследование траектории
3	цель моделирования	C	полѐт ракеты
4	моделируемые характеристики	D	координаты местоположения

Укажите правильный порядок установки соответствия в этой таблице.

Задание № 13. Дана таблица моделирования:

1	моделируемый процесс	A	человек
2	моделируемый объект	B	разработка метода лечения
3	цель моделирования	C	температура и давление
4	Моделируемыехарактеристики	D	влияние лекарства на состояние больного организма

Укажите правильный порядок установки соответствия в этой таблице.

Задание № 14. Укажите, какие модели относятся к классу предметных:

1. модель молекулы в виде кристаллической решѐтки;
2. алгоритм работы станка с числовым программным управлением;
3. макет нефтяной вышки;
4. электрическая схема радиоприемника.

Задание № 15. Укажите типы моделей в классификации по фактору времени.

Задание № 16. Эвристика – это ___.

Задание № 17. Аэродинамическая труба это ___.

Задание № 18. Укажите типы тестирования компьютерных моделей.

6. АЛГОРИТМИЗАЦИЯ И ПРОГРАММИРОВАНИЕ

6.1. Понятие алгоритма и его свойства

Алгоритм – это чѐтко определѐнная последовательность действий, описывающих процесс преобразования объекта из начального состояния в конечное с помощью понятной исполнителю последовательности команд.

Из этого определения следует, что правильно составленный алгоритм характеризуется следующими свойствами:

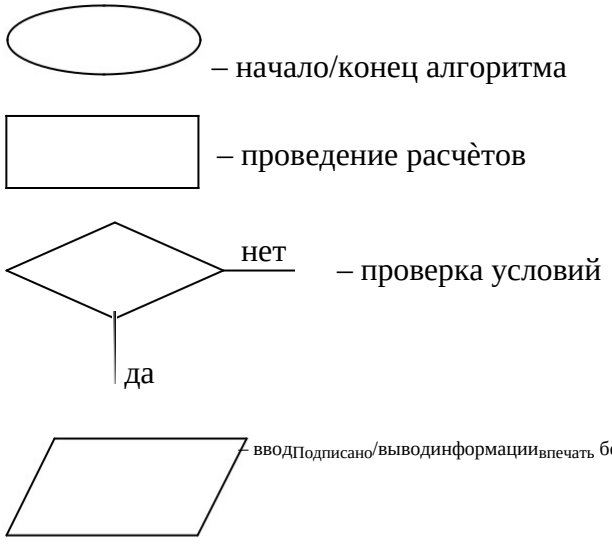
– ***дискретностью***, т. е. представлен в виде последовательности команд, которые исполнитель должен выполнять одну за другой;

– ***понятностью***, т. е. должен содержать только те команды, которые входят в систему выполняемых команд исполнителя;

– ***детерминированностью***, т. е. алгоритм должен быть представлен таким образом, чтобы, выполняя очередную команду, исполнитель точно знал, какую команду следует выполнять следующей;

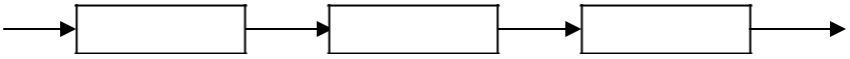
– ***результативностью***, т. е. алгоритм должен обеспечить преобразование от начальных данных к результату за конечное число команд.

Для графического представления алгоритмов компьютерных программ используют специальные обозначения. Основные из них:

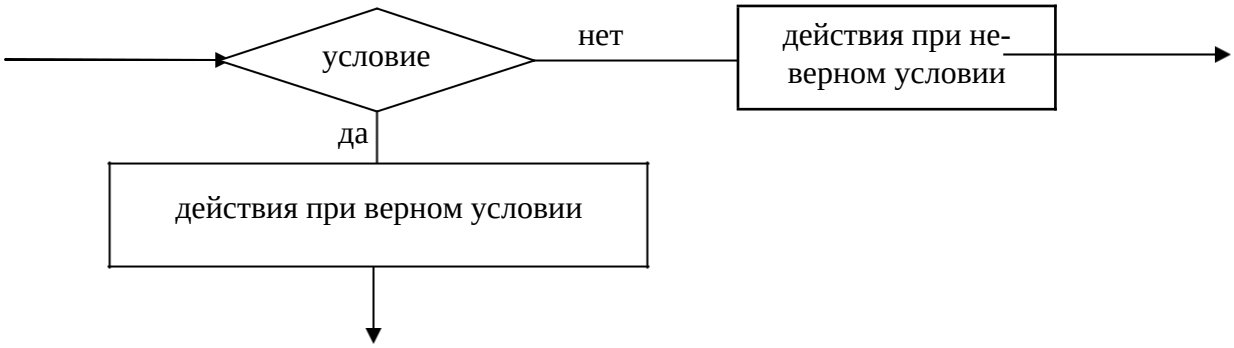


6.2. Основные типы алгоритмических структур и их блок-схемы

– ***линейная*** – каждое действие должно быть выполнено последовательно одно за другим:

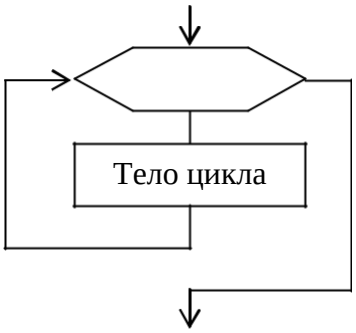


– ***ветвление*** – в зависимости от выполнения или невыполнения указанного условия реализуется та или иная последовательность команд:

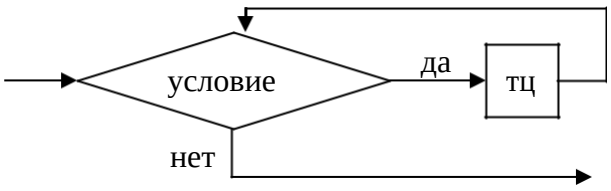


– ***цикл*** – серия команд (***тело цикла***) выполняется многократно. Разновидности циклов:

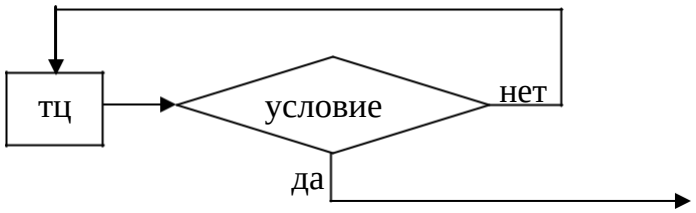
Цикл со счѐтчиком – выполняется заранее определѐнное количество раз:



Цикл с предусловием – проверяется условие и, если оно выполняется, то тело цикла (тц) повторяется, если нет – происходит переход к действию, следующему за телом цикла. Если условие не выполняется при первой проверке, то тело цикла не выполняется ни одного раза:



Цикл с постусловием – тело цикла (тц) выполняется, затем проверяется условие и, если оно не выполняется, то тело цикла повторяется, если выполняется, то происходит переход к действию, следующему за телом цикла. В этом варианте тело цикла выполняется хотя бы один раз всегда.

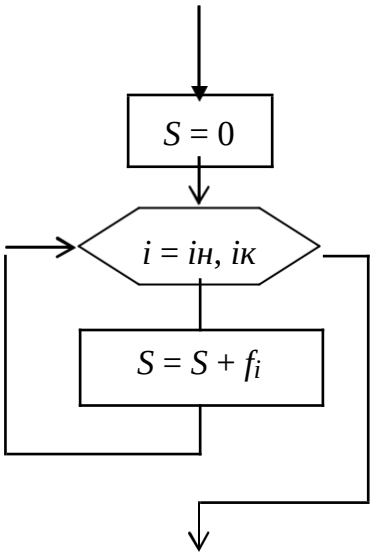


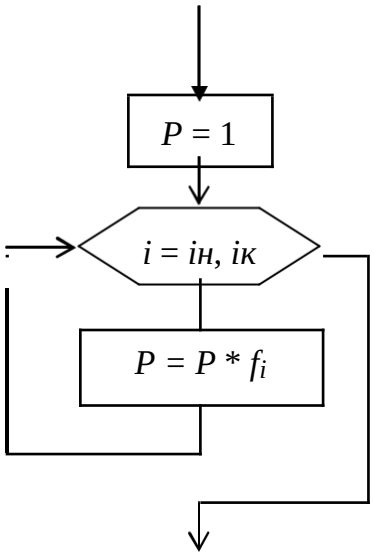
6.3. Примеры блок-схем алгоритмов

1. Суммирование/произведение заранее определѐнного количества слагаемых/сомножителей:

$$S = \sum_{i=iн}^{iк} f_i \,,\; P = \prod_{i=iн}^{iк} f_i \,,$$

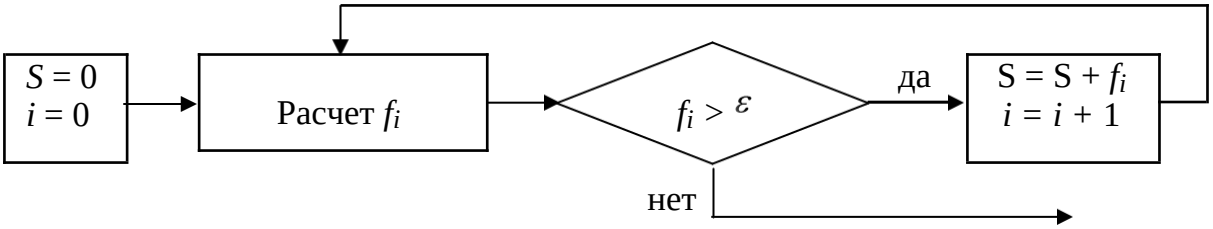
где f_i – алгоритм расчета i-го слагаемого/сомножителя, $iн$, $iк$ – границы изменения индекса i.



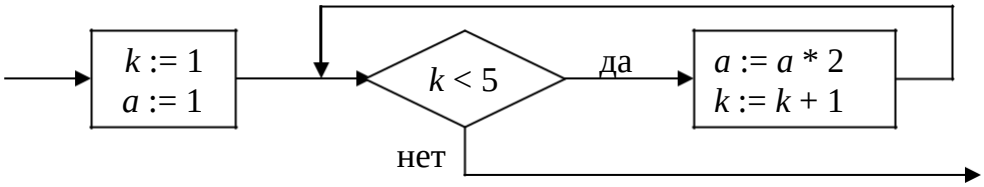


2. Суммирование слагаемых f_i до тех пор, пока выполняется условие

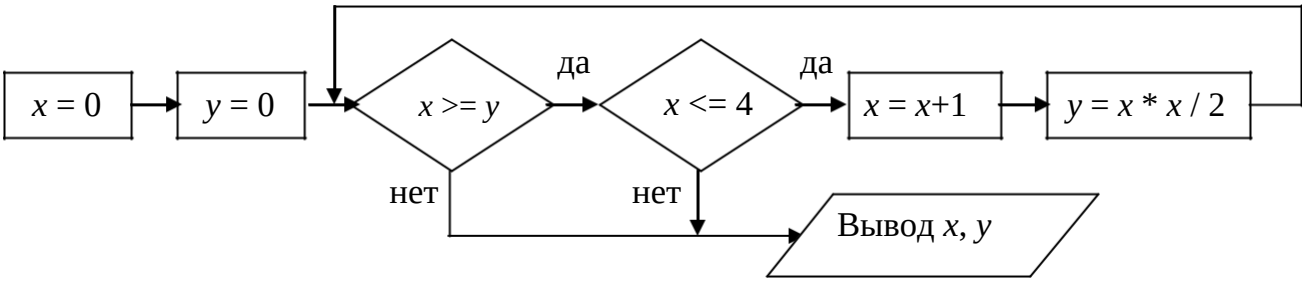
$f_i > \varepsilon$



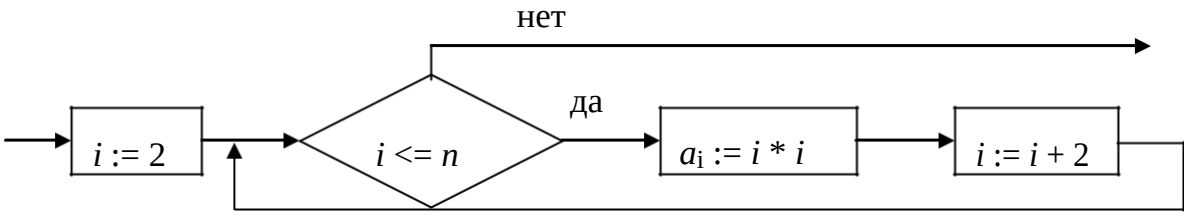
3. Алгоритм вычисляет $a = 2^4$.



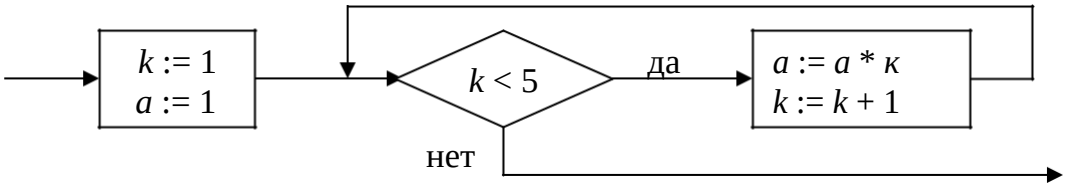
4. Алгоритм вычисляет $x = 3$; $y = 4{,}5$.



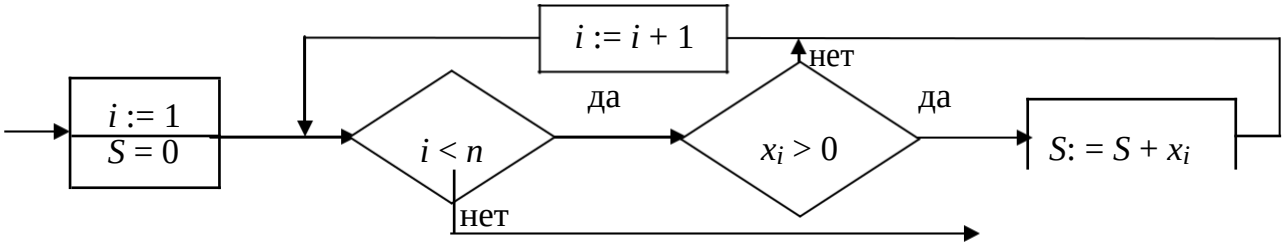
5. При $n = 8$ алгоритм вычисляет для элементов массива A значения 4, 16, 36, 64 соответственно.



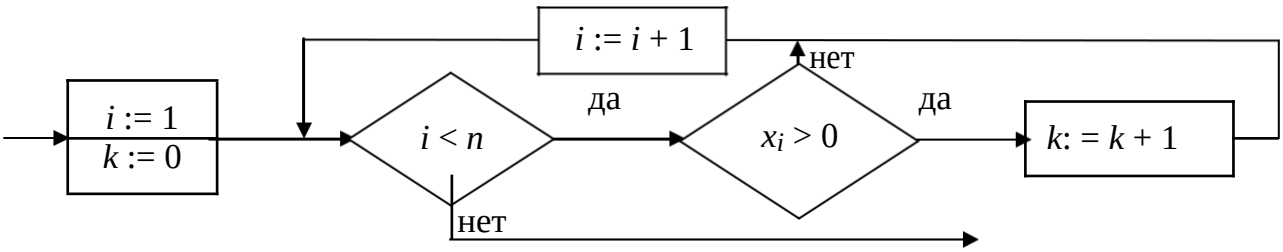
6. Алгоритм вычисляет $a = 1 * 2 * 3 * 4$.



7. Задан одномерный массив X: x_1, x_2, …, x_n . Приведѐнный фрагмент программы определяет сумму положительных элементов массива X.



8. Представленный фрагмент блок-схемы вычисляет количество нулевых элементов в массиве X, состоящем из n элементов.



6.4. Примеры алгоритмов, составленных в псевдокоде

Сложные алгоритмы удобнее записывать на ***псевдокоде,*** который включает фразы естественного языка и общепринятые математические обозначения, оформленные по стандартам, принятым в алгоритмических языках.

1. После выполнения представленного фрагмента программы переменная y приняла значение 10. Каким было значение x перед входом в этот фрагмент?

$$y := x - 1;\ x := y + 2;\ y := x + y\ ;\ \text{вывод}\ y$$

Ответ: $x = 5$.

2. Задан фрагмент алгоритма. Какими будут значения *d* и *c* на выходе, если на входе *а* = 8; *в* =3?

1) Если*а*<*в*,то*с*=*в*–*а*иначе*с*=2*(*а*–*в*)
2) *d* = 0
3) Пока *c* > А выполнять: *d* = *d* + 1;*c* = *c* – 1

Ответ. При на выходе из этого фрагмента переменные *d* и *c* примут значения: *d* = 2; *c*=8

3. Представлен алгоритм (div(*x*,*y*) - целочисленное деление *x* на *y*, mod(*x*,*y*) - целый остаток от деления):

```
k = 50
Выбор
  при div(k,12) = 4: d := k
  при mod(k,12) < 5: d := 2
  при mod(k,12) > 9: d := 3
  иначе d := 1
Всѐ
```

Ответ. Значение переменной *d* после выполнения алгоритма равно 2

4. Представлен алгоритм (mod(*x*,*y*) - целый остаток от деления):

```
k = 70
Выбор
  при mod(k,12) = 7: d := k
  при mod(k,12) < 5: d := 2
  при mod(k,12) > 9: d := 3
  иначе d := 1
Всѐ
```

Ответ. Значение переменной *d* после выполнения алгоритма равно 3

5. Представлен фрагмент программы:

x := 9; *y* := 7; *p* := *x*= *y*; *q* := *y* > *x*; *p* := *p* and *q*

Ответ. В результате выполнения фрагмента программы переменные *p* и *q* примут значения *p* = false; *q* = false

6.5. Вопросы для самопроверки по теме 6

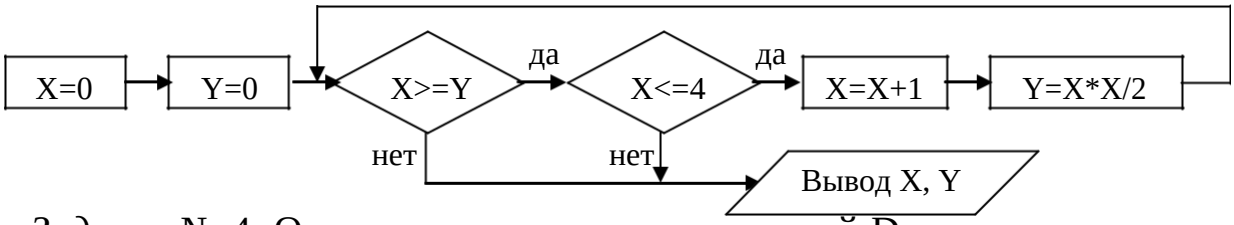
Задание № 1. Перечислите основные свойства алгоритма.

Задание № 2. Приведѐн алгоритм процедуры «ABCD» в псевдокоде:

Начать
Писать ("Введите "A,B,C,D")
Читать(A,B,C,D)
Если A=B то
Если C<B то
X:=1
Иначе
X:=2
Иначе
X:=3
Конец

Запишите этот алгоритм математическим языком на бумаге.

Задание № 3. Определите, какими будут значения X и Y на выходе из этого алгоритма:



Задание № 4. Определите значение переменной D после выполнения алгоритма (mod(x,y) – целый остаток от деления):

k=70
Выбор
при mod(k,12)=7: D:=k
при mod(k,12)<5: D:=2
при mod(k,12)>9: D:=3
иначе D:=1
Всѐ

Задание № 5. При какой организации цикла может случиться, что тело цикла не выполнится ни разу.

Задание № 6. Представлен фрагмент программы:

Y:=X+5; X:=Y; Y:=X+Y; вывод Y

Определите значение переменной X перед входом в этот фрагмент, если известно, что после выполнения этого фрагмента переменная Y приняла значение 14.

Задание № 6. Укажите вариант описания, соответствующий циклу с постусловием:

1. пока условие истинно, выполнять оператор;
2. выполнять оператор, пока условие ложно;
3. выполнять оператор заданное число раз;
4. если условие истинно, выполнять оператор, иначе – остановиться.

Задание № 8. Определите, сколько раз тело цикла выполняется в представленном фрагменте:

```
B:=10;D:=40
Начало цикла: пока D >=
B D:=D–B
Конец цикла.
```

Задание № 9. Определите вид вывода результатов в предложенном фрагменте программы:

```
X:=5
Z:=7
вывод ("X=",X," X=",Z,Z+X)
```

1. X=5 X=712
2. "X=",5," X=",7,12
3. X=5 X=7 12
4. "X=",X," X=",Z,Z+X

7. ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

7.1. Основные технологии программирования

При разработке компьютерных моделей для задач, решаемых по сложным и громоздким алгоритмам, получаются длинные программы, которые очень сложно отлаживать и тестировать. Для облегчения этих работ были разработаны разные технологии программирования, которые упрощают структуру программы и ее отладку. Основные из них:

– ***модульное программирование;***

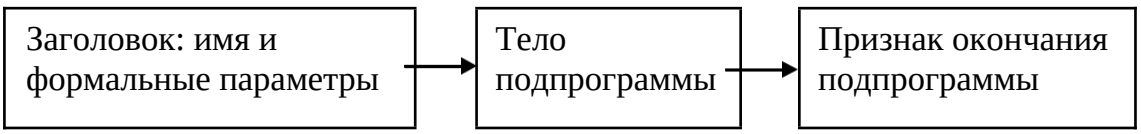
– программирование «***снизу вверх***» (***восходящее программирование***);

– программирование «сверху вниз» (нисходящее программирование).

Согласно этим технологиям в алгоритме задачи выделяются модули. ***Модуль*** – это либо логически законченный фрагмент общей задачи, либо часто повторяющийся блок расчѐтов.

Каждый модуль оформляется по определѐнным правилам в виде ***подпрограммы***, и в текст основного алгоритма вместо него вставляется короткая инструкция вызова подпрограммы. Когда выполнение программы доходит до этой инструкции, выполняется фрагмент, заложенный в тело подпрограммы, после чего управление передаѐтся на команду, следующую за инструкцией вызова подпрограммы.

Различают ***подпрограммы функции*** (используются как операнды в выражениях) и ***процедуры*** (используются как операторы). Общую структуру подпрограмм можно представить следующим об-



разом:

Тело подпрограммы – это операторы, которые программируют ту часть алгоритма, которая выделена в данный модуль. ***Формальные параметры*** – это те переменные в теле подпрограммы, значения которых надо будет перед началом работы этого модуля получить из основной программы.

В инструкции по вызову указывается имя подпрограммы и ***фактические параметры***, то есть значения формальных параметров, с которыми следует выполнить тело подпрограммы в данном месте.

Пример. Дан массив целых чисел $\{a_i\}$,i=1, 2, ..., 15. Программа должна вычислять произведение двух сумм некоторых элементов массива $\{a_i\}$. Так как операции, которые следует выполнить для суммирования, не зависят от конкретных значений используемых чисел, алгоритм суммирования, чтобы не повторять его дважды, можно оформить в виде подпрограммы. Поскольку по условию задачи конечным результатом должно быть произведение сумм, а не каждая из них по отдельности, подпрограмму можно оформить по типу подпрограммы-функции. Алгоритм, по которому будет составляться программа, представляем в псевдокоде:

Функция СУММА(*i*1,*i*2)
начало: *s* = 0
начало цикла для *i* = *i*1 до *i*2
  s = *s* + *a*(*i*)
конец цикла
СУММА = *s*
Конец функции
Начало программы
вывести на экран("введите значения массива
A") начало цикла для *j* = 1 до 15
  ввести с клавиатуры
a(*j*) конец цикла
вывести на экран ("введите границы индексов первой суммы")
ввести с клавиатуры *g*, *w*
вывести на экран ("введите границы индексов второй суммы")
ввести с клавиатуры *t*, *l*
p:= СУММА(*g*,*w*) * СУММА(*t*,*l*)
вывести на экран ("произведение равно", *p*)
Конец программы

В программу введены константы: *g* = 1; *w* = 12; *t* = 8; *l* = 15. Это – фактические параметры, которые надо использовать вместо формальных параметров *i*1 и *i*2 при вычислении значения *p*. В результате переменная *p* примет значение произведения сумм элементов с 1 по 12 и с 8 по 15 из массива $\{a_i\}$.

При использовании технологии «сверху вниз» разработка программы начинается с последовательной детализации алгоритма на

всѐ более мелкие части до тех пор, пока получатся такие модули, для которых можно написать конкретные команды. Затем составляется текст основной программы, в которой вместо фрагментов, выделенных в подпрограммы, ставят «***заглушки***». Это подпрограммы, в которых вместо реально нужных операторов ставят сигнальные печати или ввод результатов, которые должна была сосчитать эта подпрограмма. Таким образом проверяют и отлаживают последовательность действий в основном алгоритме. Затем подпрограммы-заглушки по очереди заменяют на соответствующие алгоритму подпрограммы, отлаживают и тестируют их. Такая технология облегчает создание программы, уменьшает количество ошибок и облегчает нахождение допущенных ошибок. Считается, что программа оптимального по размерам модуля целиком должна помещаться на экране дисплея.

При использовании технологии «снизу вверх» в первую очередь определяются и разрабатываются вспомогательные модули, которые потребуются для проектируемой программы. После того, как все модули отлажены, из них, как из кубиков, в соответствии с начальным алгоритмом собирается основная программа.

7.2. Основные принципы структурного программирования (программирование без GO TO)

Структурное программирование – методология и технология разработки программных комплексов, основанная на принципах модульного программирования и программирования "сверху-вниз".

Алгоритм задачи представляется как композиция *только трѐх базовых типов алгоритмов: линейных, ветвлений и циклов.* Эти конструкции могут быть соединены или вложены друг в друга произвольным образом, но никаких других способов управления последовательностью выполнения операций не используется.

Основные языки программирования, использующие структурную технологию:

– Ада, Си – языки общего назначения;

– Бейсик (до Visual Basic);

– КОБОЛ – для экономических задач (много операторов, облегчающих манипуляции с файлами);

– Фортран, Паскаль, ПЛ/1 – для вычислительных задач (удобные средства для записи формул).

7.3. Основные понятия объектно-ориентированного программирования

Объектно-ориентированное программирование (ООП) применяют при программировании разных манипуляций над объектами. Например, при составлении программ управления размерами и положением окон Windows, листами книги Excel, файлами и т. п.

Основные термины и понятия объектно-ориентированного программирования:

– моделируемая система состоит из ***объектов***. Объекты могут быть вложены друг в друга, например, объект «лист Excel» – это часть объекта «книга Excel»;

– объекты каким-то образом взаимодействуют между собой;

– каждый объект характеризуется своим состоянием и поведением. Состояние объекта задаѐтся значением некоторых его ***свойств***. Например, объекты типа «книга Excel» имеют свойства: имя, размер, открыта/закрыта и т. п. Действия, которые можно выполнять над объектом или которые он сам может выполнять, называются ***методами***. Например, объект типа «книга Excel» можно открыть, закрыть, переименовать, перенести в другую папку и т. п. После каждого действия изменяются какие-то свойства объекта.

Класс объектов – шаблон, определяющий основные свойства, методы и события группы объектов, объединяемых в класс. Это же можно сформулировать другими словами: это множество объектов, имеющих общее поведение и общую структуру.

События – ситуации, в которых надо программировать какой-то отклик объекта. Например, что делать, когда над гиперссылкой или кнопкой расположен курсор, когда щѐлкает курсор, когда происходит двойной щелчок.

Наследование – порождает иерархию объектов. Оно может быть смоделировано с помощью таксонометрической классификационной схемы (иерархии). В основном классе (***родителе***) можно выделять подклассы (***потомки***). Они состоят из объектов, входящих в класс родителя и обладают наряду со всеми его характеристиками дополнительной группой свойств, которых у других объектов класса-родителя нет. Пример: класс-родитель – окна Windows, подклассы – диалоговые окна, окна документов, окна папок. Подклассы окон документов – окна документов Word, окна документов Excel, окна документов PowerPoint и т. п.

Инкапсуляция – сокрытие деталей программ, создающих и манипулирующих объектами. Создание объектов, манипулирование ими осуществляется программами языка ООП. Программист указывает в своей программе только то, что и с каким объектом нужно сделать, или какой результат нужно получить. То есть объекты рассматриваются как «чѐрные ящики». Такой способ упрощает разработку программы и еѐ модификацию.

Полиморфизм –.для выполнения одного и того же типа действий в разных подклассах одного класса можно использовать одно и то же имя, хотя это действие реализуется разными внутренними методами (программами). Например, действие «трансформация объекта», которое программист обозначает этим термином для фигур, входящих в разные подклассы класса «геометрические фигуры», выполняется по разным формулам, следовательно, его выполняют разные программы (методы). Другими словами это означает способность объектов выбирать внутренний метод самостоятельно, исходя из типа их данных.

Основные языки ООП:

– C++ – для системного программирования;

– Java, JavaScript, PHP, Perl – для разработки сценариев в динамических Veb-страницах;

– Simula – первый язык, построенный по принципам ООП;

– Delphi (Object Pascal) – удобен для программирования баз данных.

7.4. Этапы решения задач на компьютере

1. Постановка задачи:

– сбор информации о задаче;

– описание исходных данных и конечных целей;

– определение формы выдачи результатов.

2. Анализ и исследование модели задачи:

– анализ существующих аналогов;

– анализ технических и программных средств;

– разработка математической модели;

– разработка структур данных.

3. Разработка алгоритма:

– выбор метода проектирования алгоритма;

– выбор формы записи алгоритма (блок-схема, псевдокод и т. п.

– выбор тестов и метода тестирования;

– проектирование алгоритма.

4. Программирование:

– выбор языка программирования;

– уточнение способа организации данных;

– запись алгоритма на выбранном языке.

5. *Отладка и тестирование:*

– ***синтаксическая*** отладка: исправление ошибок в форме записи конструкций;

– отладка семантики и логической структуры: ***семантика*** – система правил истолкования отдельных конструкций языка, например проверка правильности организации циклов, ветвлений и т. п., соответствия типов переменных в выражениях, ***логическая структура*** – правильная последовательность обработки данных;

– тестовые расчѐты и анализ результатов тестирования;

– совершенствование программы.

Деятельность, направленная на исправление ошибок в программной системе, называется ***отладкой***. ***Тестирование*** – прогон отлаженной программы на эталонных вариантах исходных данных, для которых заранее известны результаты (см. п.5.3.).

6. *Анализ результатов тестирования* и, если нужно, уточнение модели и повторение п. п. 2 – 5.

7. *Сопровождение программы:* составление документации по математической модели, алгоритму, программе, набору тестов, использованию готовой программы и т. п.

7.5. Вопросы для самопроверки по теме 7

Задание № 1. Выберите правильные варианты: при проектировании программного обеспечения используют подходы:

1. сверху-вниз
2. снизу-вверх
3. слева-направо
4. справа-налево

Задание № 2. Ошибка в форме записи программы – это ____ ошибка.

Задание № 3. Ошибка в последовательности обработки данных, организации циклов, ветвлений, соответствий типов переменных, используемых в выражениях, – это ____ ошибка.
Задание № 4. Укажите типовые алгоритмические структуры, используемые при структурном программировании.
Задание № 5. Укажите различие между формальными и фактическими параметрами подпрограмм.
Задание № 6. Укажите, какие из приведенных терминов являются базовыми понятиями в объектно-ориентированном подходе к программированию:

1. объект
2. свойство
3. метод обработки
4. инструкции
5. данные
6. модель
7. событие
8. класс объектов

Задание № 7. Какой методологии программирования присуще понятие "Иерархия классов?
Задание № 8. . Набор операторов, выполняющих заданное действие и не зависящих от других частей исходного кода, – это ____.
Задание № 9. Выделите принципы, которые можно использовать при разработке программ методом структурного программирования:

1. принцип модульной разработки сложных программ;
2. использование композиции трѐх базовых структур при записи алгоритма - линейных, ветвлений и циклов;
3. использование композиции двух базовых структур при записи алгоритма - ветвлений и циклов;
4. использование большого количества подпрограмм.

Задание № 10. Объясните смысл понятий, используемых в методологии объектно-ориентированного программирования: объект, класс, ме-тод, наследование, полиморфизм, инкапсуляция.

8. ЛОКАЛЬНЫЕ И ГЛОБАЛЬНЫЕ СЕТИ ЭВМ МЕТОДЫ ЗАЩИТЫ ИНФОРМАЦИИ

8.1. Классификация вычислительных сетей

Компьютерная сеть – это два или больше компьютеров, связанных каналами передачи информации. Цель создания сетей – обеспечение совместного доступа к ***сетевым ресурсам***.

Виды сетей по географическому признаку и размерам:

– *локальная*: сеть предприятия или учреждения, в которой рабочие станции распределены на расстоянии не более 300-500м. Обозначается аббревиатурами ЛВС или LAN (Local Areal Net);

– *региональные и корпоративные*: объединяют компьютеры большого региона или филиалы учреждений, разбросанные в пределах города, большого географического региона, страны. Обозначается аббревиатурой MAN. (Metropolitan Areal Net);

– *глобальные*: объединяют всех абонентов (LAN и MAN) вне зависимости от места их расположения: страны, континента, всей земли, например, Интернет. Обозначается аббревиатурой WAN. (World Areal Net).

Виды сетей по возможностям передачи сигналов:

– ***узкополосные***: канал связи может передавать только один сигнал в любой момент времени (телефонная линия);

– ***широкополосные***: одновременно можно передавать несколько сигналов, используя для каждого свою частоту передачи (кабельное телевидение).

8.2. Виды сетевых ресурсов

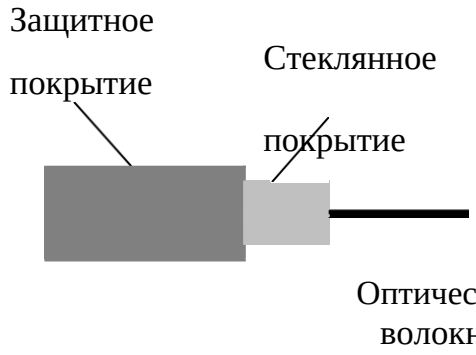
– ***аппаратные***: общий принтер, общий жѐсткий диск для хранения программ и данных отдельных пользователей;

– ***программные***: для выполнения сложных и длительных расчѐтов можно подключиться к мощной ЭВМ, послать на неѐ задание на расчѐты и исходные данные, и, по окончании расчѐтов просто получить готовые результаты.

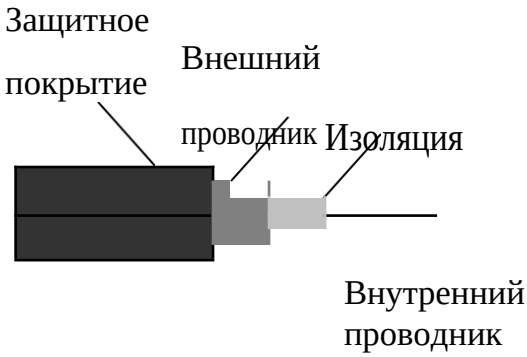
– *информационные*: всевозможные справочные данные, архивы научных работ, книг и т. п.

Виды каналов передачи информации:

– ***специальные кабели***: коаксиальный кабель, оптоволоконный кабель (самая надѐжная и быстрая связь), витая пара – телефонный



Оптоволоконный кабель



Коаксиальный кабель

кабель;

– ***электромагнитные волны разных частот***: спутниковая связь, подключение ноутбуков, мобильных телефонов;

Важнейшая характеристика канала – ***скорость передачи информации***. В ЛВС обычно от 10 до 100 Мбит/сек., в крупных сетях – _____**

На скорость передачи информации влияет также тип ее кодировки: чем короче кодировка символов, тем меньше бит она содержит и, следовательно, тем быстрее она передается по тому же самому каналу.

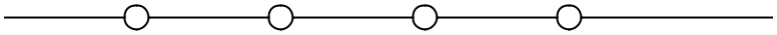
8.3. Топология и архитектура вычислительных сетей

Топология сети – это логический и физический способ соединения компьютеров. Различают следующие базовые варианты топологий:

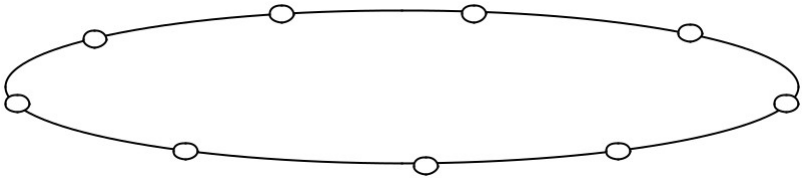
– ***одноранговая***: все компьютеры имеют одинаковые права (обычно до 10 рабочих станций);

– ***сети на основе сервера***: с***ервер*** – это компьютер, на котором хранится информация, необходимая разным пользователям или же выполняющий работы по поддержанию связи между компьютерами. Другое значение этого термина – программа, выполняющая запросы с рабочих станций на доступ к ресурсам. ***Рабочая станция (клиент)*** – это внутренний компьютер ЛВС. Другое значение этого термина – программа, создающая запрос серверу на какие-либо ресурсы;

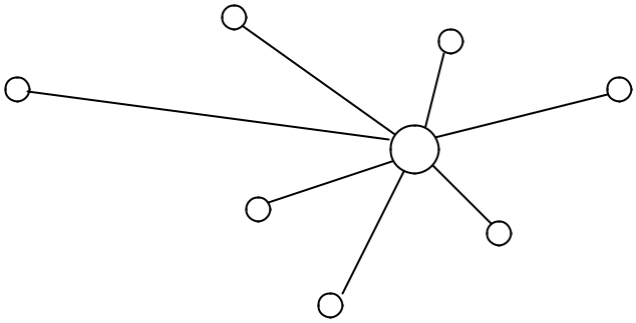
– ***линейная*** (***шинная***) ***топология***: соединяющий кабель последовательно проходит от одного компьютера к другому:



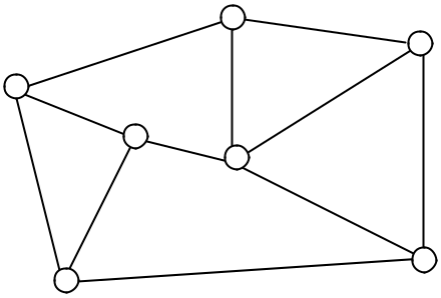
– ***кольцевая топология***:



– ***звездообразная топология***:



– ***полносвязанная*** (***сетевая топология***): имеется много путей, по которым можно переслать сообщение с одного компьютера на другой.



Архитектура сети – это понятие, которое включает в себя топологию сети, состав ее устройств, правила их взаимодействия, кодирование, адресацию и передачу информации, управление потоком сообщений, контроль ошибок, анализ работы в аварийных ситуациях.

Основные виды архитектур (*физический уровень*):

– ***Ethernet***: широковещательная сеть, т.е. все рабочие станции могут принимать все сообщения. Топология линейная или звездообразная.

– ***Token Ring***: топология кольцевая. Каждый узел ожидает своей очереди на посылку сообщения.

– ***FDDI***: высокоскоростная передача данных по оптоволоконным линиям. Топология смешанная (кольцевая + древовидная)

– ***ATM***: передача цифровых данных, видеоинформации и голоса по одним и тем же линиям.

– ***Wi-Fi***, ***IrDa***: беспроводная (радиорелейная) связь.

8.4. Программное обеспечение вычислительных сетей

Сетевые протоколы – это наборы правил для обмена информацией в сети и разработки сетевого оборудования. Более подробно их можно определить как стандарты, которые определяют формы представления и способы пересылки разных типов сообщений между компьютерами, а также правила совместной работы различного оборудования в сетях, необходимые при создании этого оборудования.

Совокупность протоколов, которые используют компьютеры при работе в сети, обозначается термином ***стек***. Традиционно стек делится на 7 уровней, функции которых определяются ***эталонной моделью взаимодействия открытых систем*** (сетевая модель ***OSI*** – Open System Interconnection):

– ***Физический уровень*** (**1**): процедуры управления аппаратурой передачи данных и подключѐнным к ней каналам связи. Примеры протоколов: Ethernet, Arc net, Token Ring.

– ***Канальный уровень*** (**2**): отвечает за передачу информации по логическому каналу, установленному между двумя ЭВМ, соединѐнными физическим каналом. На этом уровне обнаруживаются ошибки передачи пакетов, реализуется алгоритм восстановления информации в случае обнаружения сбоев или потери данных. Детализируются способы осуществления непосредственной связи объектов сетевого уровня. Примеры протоколов: LAP-B, SNAP, HDLC

– ***Сетевой уровень*** (**3**): отвечает за маршрутизацию пакетов и связь между разными сетями, т.е. устанавливает логические каналы между ***объектами*** для реализации протоколов транспортного уровня. Примеры протоколов: ***IP*** (***Internet Protocol***), IPX,IDN,X.25

– ***Транспортный уровень*** (**4**): определяет интерфейс между процессами и сетью, т.е. устанавливает логические каналы между ***процессами*** и обеспечивает передачу информационных пакетов, сформированных по правилам сетевого уровня. Примеры

протоколов: ***TCP*** (***Transmission Control Protocol***), UDP, NSP, X.224, NetBIOS.

– ***Сеансовый уровень*** (**5**): осуществляет установку и поддержку сеанса связи между двумя абонентами. Определяются средства, необходимые для синхронизации и управления обменом данных между сетевыми объектами. Примеры протоколов: RPC, X.225

– ***Представительский уровень*** (**6**): определяет синтаксис данных, т.е. коды и форматы данных, посылаемых в сеть. Каждая прикладная программа, каждая операционная система имеет свой способ кодировки данных и команд. На этом уровне устанавливаются стандартные способы кодировки для информации, которая выходит из локальной сети. Аппаратное обеспечение, обеспечивающее перекодировку информации при выходе из локальной сети, обозначается термином ***шлюз***. Примеры протоколов: X.226

– ***Прикладной уровень*** (**7**): совокупность правил для разработки программ-приложений, которые делают запросы к ресурсам, расположенным в сети. В него входит регламентация всех работ, которые связаны с запуском программ пользователя и их выполнением. Примеры протоколов этого уровня: HTTP, SMTP, FTP, DNS, POP3. В рамках этих протоколов пользователь делает запросы на использование сетевых ресурсов.

Основной протокол, под управлением которого работает ***Интернет***, – это ***TCP/IP***. Другими словами совокупность протоколов TCP/IP является основой построения и функционирования сети Интернет.

8.5. Протоколы электронной почты

Серверная часть пакета программ, обслуживающих электронную почту, состоит из трѐх основных подсистем: подсистемы хранения сообщений, транспортной подсистемы, службы каталогов.

Подсистема хранения обеспечивает хранение и разделение пришедшей почты с помощью учѐтных записей. Учѐтная запись содержит учѐтное имя пользователя (логин: название почтового ящика) и ряд другой информации.

Транспортная подсистема обеспечивает пересылку исходящих сообщений от своих клиентов и транзитных сообщений.

Служба каталогов обеспечивает хранение и корректировку учѐтных записей, направление сообщения именно тому, кому оно направлено.

Основные протоколы:

SMTP (Simple Mail Transfer Protocol) – простой протокол передачи исходящих сообщений. Он регламентирует передачу только символьной информации;

POP (Post Office Protocol) – используется для работы с входящей почтой

MIME (Multipurpose Internet Mail Extension) – многоцелевое расширение почты Интернета. Этот протокол регламентирует пересылку писем с присоединѐнными файлами музыки или изображений.

8.6. Коммуникационное оборудование

Повторитель (***репитер***) – передаѐт электрические сигналы от одного участка кабеля к другому, предварительно усиливая их и восстанавливая их форму. Используется в локальных сетях для увеличения их протяжѐнности. В терминологии OSI функционирует *на физическом уровне.*

Коммутаторы – многопортовые повторители, которые считывают адрес назначения каждого входящего пакета и передают его только через тот порт, который соединѐн с компьютером-получателем. Могут функционировать *на разных* уровнях OSI.

Концентратор (hub)– многопортовое устройство для усиления сигналов при передаче данных. Используется для добавления в сеть рабочих станций или для увеличения расстояния между сервером и рабочей станцией. Работает как коммутатор, но вдобавок может усиливать сигнал.

Мультиплексор (устройство или программа) – позволяет передавать по одной коммуникационной линии одновременно несколько различных сигналов.

Шлюз – подготавливает данные к передаче между сетями или прикладными программами, использующими разные протоколы (способы кодировки, физические среды для передачи данных). Например, при подключении локальной сети к глобальной. Функционирует на *прикладном* уровне.

Мост – соединяет две сети с одинаковыми протоколами, усиливает сигнал и пропускает только те сигналы, которые адресованы

компьютеру, находящемуся по другую сторону моста. Мостом также называют компьютер с двумя сетевыми картами, предназначенный для соединения сетей.

Маршрутизатор – соединяет разные ЛВС, как и мост, пропускает только ту информацию, которая предназначена для сегмента, с которым он соединѐн. Отвечает за выбор маршрута передачи пакетов между узлами. Выбор маршрута осуществляется на основе протокола маршрутизации, содержащего информацию о топологии сети, и специального алгоритма маршрутизации. Функционирует на *сетевом* уровне OSI.

8.7. Основные понятия криптографии

Криптография (***шифрование***) – это кодирование данных, посылаемых в сеть, так, чтобы их могли прочитать только стороны, участвующие в конкретной операции. Надѐжность защиты информации зависит от алгоритма шифрования и длины ключа в битах.

Метод шифрования – это алгоритм, описывающий порядок преобразования исходного сообщения в результирующее.

Ключ шифрования – это набор параметров, необходимых для применения метода шифрования. При компьютерном шифровании ключ представляется как последовательность символов, сохранѐнных на жѐстком или съѐмном диске. Различают ***статические*** варианты ключей – они не меняются при работе с разными сообщениями и д***инамические ключи*** – они изменяются для каждого сообщения.

Типы методов шифрования:

– ***симметричные***: один и тот же ключ используется и для шифровки, и для дешифровки. Такие методы неудобны в электронной коммерции, так как у продавца и покупателя должны быть разные права к доступу информации. Продавец посылает всем покупателям одни и те же каталоги, но покупатели возвращают ему конфиденциальную информацию о своих кредитных картах, и нельзя смешивать заказы и их оплату для разных покупателей;

– ***асимметричные*** (***несимметричные***): основываются на специальных математических методах, которые создают пару ключей так, что то, что зашифровано одним ключом, может быть дешифровано только другим, и наоборот. Один из ключей называется ***открытым***, его может получить каждый желающий. Второй ключ разработчик ключа оставляет себе, он называется ***закрытым (секрет-***

ным). Например, в электронной коммерции продавец создает пару ключей. Закрытый ключ он оставляет себе, открытый – посылает покупателю. Если клиент получил файл, к которому не подходит его ключ, значит, его послала не та фирма, с которой он ведет деловую переписку. Покупатели шифруют свои заказы, договоры своим открытым ключом и посылают их продавцу. Их дешифровку может сделать только владелец закрытого ключа, т. е. продавец.

Защищѐнный канал – способ передачи сообщений, при котором обе стороны используют один и тот же метод шифрования, известный только им.

Электронно-цифровая подпись – код специальной структуры, который позволяет однозначно связать содержание документа, пересылаемого по сети, и его автора (***аутентифицировать*** документ).

Хэш-функции – это функции, которые позволяют из одной последовательности чисел получить другую последовательность таким образом, что обратное преобразование невозможно.

Хэширование – обработка некоего сообщения хэш-функцией, при которой двоичные коды сообщения воспринимаются как коды двоичных чисел. В результате создается уникальная последовательность символов фиксированной длины, которая однозначно соответствует содержанию исходного сообщения. Эту последовательность обозначают термином ***хэш-код***. Смысл хеширования можно проиллюстрировать бытовым примером: для того, чтобы не забыть какой-нибудь тюк багажа в пути, мы запоминаем количество мест багажа, и, чтобы проверить, не забыли ли мы что-то, просто пересчитываем тюки. Но, зная только количество тюков, мы не сможем получить информацию о том, что в них содержалось.

Дайджест сообщения, ***электронная печать***, ***сводка сообщения*** – уникальная последовательность символов фиксированной длины, которая создаѐтся на основе содержания электронного документа. Дайджест получается путѐм хеширования. Он уникален для сообщения, как отпечатки пальцев для человека. Если изменить хотя бы один символ в документе, хэш-код станет другим.

8.8. Электронно-цифровая подпись (ЭЦП)

Технически шифрование/дешифровка с помощью хэш-функций состоят из следующих этапов:

1) сообщение дополняется сведениями об авторе и обрабатывается хэш-функцией. Это – ЭЦП;

2) ЭЦП добавляется к документу, который содержит передаваемую информацию и сведения об авторе;

3) полученный файл шифруется ключом ассиметричного метода и посылается принимающей стороне;

4) принимающая сторона расшифровывает полученное сообщение с помощью своего ключа, отделяет от него ЭЦП и обрабатывает той же хэш-функцией, что и отправитель;

5) Затем сравнивают полученный хэш-код с тем, который пришѐл в сообщении. Если они совпали, значит, сообщение не подверглось изменениям в пути и сведения об авторе правильны.

Абсолютной гарантии в невозможности подделать ЭЦП нет, так же как и для обычной подписи под документом, но подделать ЭЦП или вскрыть ключ гораздо сложнее и в интеллектуальном, и в экономическом плане. Технология взлома – это метод простого перебора вариантов следования символов, из которых можно создать ключ дешифровки. Время взлома определяется производительностью вычислительной техники, используемыми алгоритмами шифрования и длиной ключа (бит). Пример: при симметричном шифровании с ключом в 40 бит надо перебрать 2^{40} вариантов. При нынешней технике это займѐт меньше суток. При длине ключа в 128 бит – необходимое время взлома больше, чем возраст Вселенной.

8.9. Компьютерные вирусы

Компьютерные вирусы – это программы, которые могут самокопироваться, скрытно внедрять свои копии в файлы, в загрузочные секторы дисков, искажать документы, работу операционной системы и прикладных программ.

Каналы распространения вирусов:

– электронная почта;

– интернет-сайты;

– рассылки графических и аудио-файлов;

– непроверенные съѐмные диски.

По «*среде обитания*» вирусы можно классифицировать как:

– ***файловые***: внедряются в программы и обычно активизируются при их запуске. После запуска программы вирус находится в оперативной памяти и остается активным до перезагрузки ОС;

– ***загрузочные***: внедряются в загрузочный сектор диска и активизируются при загрузке ОС;

– ***макровирусы***: заражают файлы документов, оформляются в виде макросов, которые искажают содержание документа;

– ***сетевые***: распространяются по сети, внедряясь в файлы, пересылаемые по электронной почте, или заражая файлы, скачиваемые с серверов файловых архивов.

К основным разновидностям сетевых вирусов относятся:

– с***етевые черви***: распространяются по сетям, внедряясь в файлы вложений электронных писем. Способны самокопироваться и рассылать свои копии по другим сетевым адресам, имеющимся в компьютере. Парализуют работу сети и портят информацию в компьютере. Активизируются либо автоматически в момент открытия сеанса связи с сетью, либо при просмотре электронной почты.

– ***троянские программы*** (***трояны***): вирусы, производящие несанкционированные действия с информацией и маскирующиеся под привлекательные для пользователя программы. Обычно создают ярлык с ложным названием типа Game.exe. Другими словами их можно охарактеризовать как программы, которые не создают собственных копий, но преодолевают систему защиты компьютера и оказывают вредоносное воздействие на еѐ файловую систему;

– ***скрипт-вирусы***: программы, написанные на языках JavaScript или VBScript, которые активизируются при загрузке Web-страниц и выполняют разрушительные действия в программной системе компьютера.

8.10. Классификация антивирусных программ

В настоящее время существует множество программ, выполняющих антивирусную защиту. Их можно классифицировать по следующим признакам:

По принципу работы

– ***мониторы*** – являются частью ОС, автоматически проверяют на вирусы все поступающие файлы, все открываемые и закрываемые файлы, если нужно – попутно лечат. К программам такого типа относятся NOD32, AVP (Antiviral Toolkit Pro – лаборатория Касперского);

– *сканеры* – проверяют файлы, находящиеся в ОЗУ, выдают сообщения о подозрительных файлах, но не лечат их.

– *брандмауэр (firewall)* – программа, которая является фильтром при обмене информацией между внешней и локальной сетью. Она разрешает доступ к информации сети только уполномоченным лицам и задерживает файлы тех типов, которые указаны администратором сети (вирусы, баннеры, файлы запрещенной тематики).

По принципу обнаружения вирусов:

– ***полифаги***: просматривают коды программ на наличие в них известных вирусных фрагментов или фрагментов кода, типичных для вирусных действий, если можно – лечат. Качество работы сильно зависит от даты обновления вирусной базы. Наиболее популярные программы этого типа: NOD32, AVP, Doctor Web, Norton Antivirus,

– *ревизоры*: только диагностируют, заражѐн ли файл. Для этого они сверяют эталонные контрольные суммы всех неизменяемых файлов в компьютере с контрольными суммами на момент проверки. Те файлы, у которых они не совпали – заражены. Качество работы зависит от того, был ли заражѐн компьютер в момент создания базы эталонных контрольных сумм при загрузке ревизора на компьютер. Примером программы такого типа является Aids test,

– ***эвристические анализаторы***: реагируют на фрагменты кода, которые выполняют действия, похожие на вирусные, но не зарегистрированы в антивирусных базах.

Другие типы антивирусных программ: ***фаги***, ***дезинфекторы***, ***вирус-фильтры***, ***иммунизаторы***, ***детекторы***, ***вакцины***.

Помимо применения антивирусных программ для уменьшения вероятности потери и порчи информации из-за вирусов рекомендуется периодически архивировать важную информацию и хранить ее на съемных носителях. Это позволяет при обнаружении испорченных вирусом файлов восстановить их предыдущие версии, а не создавать их заново. Основные архиваторы, которые используются в настоящее время, это: 7–Zip, ARJ, WinRAR, WinZip. Основные расширения архивов:*.ZIP, *.RAR, *.ARJ.

8.11. Облачная антивирусная защита

Активное развитие телекоммуникационных систем в настоящее время сделало возможным предоставление конечному пользователю доступа к услугам, приложениям и вычислительным ресурсам различной мощности, находящимся на разных серверах. Такие виды услуг называются интернет-облаками или облачными вычислениями.

Одна из услуг, относящаяся к облачным вычислениям, – это антивирусные облака.

Производители антивирусных программ имеют в своем распоряжении гораздо более мощное оборудование и программное обеспечение для распознавания вирусных сигнатур (участков кода, которые могут нанести вред ПО компьютера, файлов в критических местах ЭВМ), чем то, которое имеет конечный пользователь. При работе в облаке антивирусные программы на компьютерах пользователей все подозрительные файлы, которые они не могут однозначно определить как вирусы, отправляют на сервер своей фирмы-производителя для анализа в экспертной системе. Сервер производителя, со своей стороны, выставляет по выполняемым операциям баллы опасности присланного приложения и расширяет, если нужно, базу вирусных сигнатур. Так как на сервер стекается информация от многих пользователей, такая технология ускоряет обнаружение новых вариантов вирусов и время реакции на них, а также обеспечивает обновление антивирусных баз всех своих пользователей частыми и маленькими порциями.

8.12. Вопросы для самопроверки по теме 8

Задание № 1. Расшифруйте аббревиатуры LAN, MAN, WAN. *Задание* № 2. Как называется иерархическая система назначения уни-кальных текстовых имѐн каждому компьютеру, находящемуся в се-ти?

Задание № 3. Укажите, какая часть электронного адреса ресурса описывает путь к файлу, расположенному на сервере:

http://wwwю.google.com/inf02000/01-02/det123.html

Задание № 4. Укажите тип связи, который на сегодня является наиболее защищѐнным от несанкционированного доступа: оптоволоконный кабель, телефонный кабель (витая пара), электромагнитные волны.

Задание № 5. Укажите, какие термины не относятся к базовым топо-логиям сетей: линейная (шинная), кольцевая, снежинка, полносвя-занная (сетевая).

Задание № 6. Укажите среди приведенных терминов, те, которые обозначают беспроводную связь: Ethernet, Wi-Fi, IrDa, FDDI.

Задание № 7. Укажите, что в сетевой терминологии обозначают термины «сервер», «клиент».

Задание № 8. Укажите, что в сетевой терминологии обозначает термин «протокол».

Задание № 9. Укажите, какая часть электронного адреса ресурса описывает протокол передачи информации:

http://www.google.com/inf02000/01-02/det123.html

Задание № 10. На сколько уровней разбиты протоколы сетевой модели OSI?

Задание № 11. Укажите протоколы сетевой модели OSI, относящиеся к прикладному уровню, и расшифруйте их назначение.

Задание № 12. Укажите, на каком уровне сетевой модели OSI устанавливаются правила маршрутизации.

Задание № 13. Как называется компьютер с двумя сетевыми картами, предназначенный для соединения сетей.

Задание № 14. Выберите правильный вариант назначения электронно-цифровой подписи:

1. удостоверение истинности отправителя и целостности сообщения;
2. восстановление повреждѐнного сообщения;
3. шифрование сообщение для сохранения его секретности;
4. пересылка сообщения по секретному каналу.

Задание № 15. Укажите, по какому признаку различаются симметричные и асимметричные методы шифрования. *Задание* № 16. Укажите типы вирусных программ.

Задание № 17. Укажите классификацию антивирусных программ по принципу работы.

ОТВЕТЫ НА ВОПРОСЫ ДЛЯ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

1. Основные понятия теории информации

Задание № 1. 1 бит.

Задание № 2. 1 бит, если есть только один рейс, большее количество – если их несколько.

Задание № 3. Ответ №3.

Задание № 4. Ответ №1.

Задание № 5. Ответ №1.

Задание № 6. Ответ №2.

Задание № 7. Ответ №2.

Задание № 8. Ответ №2.

Задание № 9. Ответ №1.

Задание № 10. Ответ №4.
Задание № 11. Ответ №2.
Задание № 12. 111111111_2.
Задание № 13. 1Кб, 1Гб, 1Мб, 1Тб.
Задание № 14. 5.
Задание № 15. 1 бит.
Задание № 16. 32 байта.
Задание № 17. В.
Задание № 18. 10а; 10б; 11а; 8а; 8б.
Задание № 19. 20 бит, 2 байта, 10 бит.
Задание № 20. 48 байт.
Задание № 21. 14_{10}.
Задание № 22. 1101_2.
Задание № 23. 10010001_2.
Задание № 24. 55_7, 55_8, 55_{16}.
Задание № 25. 20 бит, 2 байта, 10 бит.
Задание № 26. Вербальная.
Задание № 27. 15 байт

2. Основы логики и логические основы компьютера

Задание № 1. Ответ №2.
Задание № 2 Дизъюнкция.
Задание № 3. Простое ложное.
Задание № 4. Ответ №3.
Задание № 5. А=0; В=1.
Задание № 6. Дизъюнкция.
Задание № 7. A B или A & B или A^B или A• B.
Задание № 8. А = «Завтра будет дождик». В = «Я возьму зонтик». С = «Я никуда не пойду». А В+ С.
Задание № 9. Вербальная
Задание № 10. Ответ №2.
Задание № 11. Инверсия.

3. Технические средства реализации информационных процессов

Задание № 1. Интерфейс объединения функциональных устройств в вычислительную систему.

Задание № 2. Процессор, ОЗУ, ПЗУ, Кэш-память, интерфейсные схемы шин, гнѐзда расширений (слотов), обязательные системные средства ввода/вывода.
Задание № 3. К классу машин четвѐртого поколения.
Задание № 4. Контроллеры.
Задание № 5. CD-ROM, CD-R, DVD-ROMDVD-R, DVD-RAM.
Задание № 6. По механизма выполнения печати.
Задание № 7. Устройство для соединения блоков компьютера с разными способами представления информации.
Задание № 8. Жѐсткий диск, гибкий диск, оптические диски, цифро-вые видеодиски, магнитооптические диски, стримеры.
Задание № 9. Плоттер, монитор.
Задание № 10. ОЗУ, кэш-память, ПЗУ, CMOS RAM, flash memory, видеопамять.
Задание № 11. Мышь, джойстик.
Задание № 12. ОЗУ.
Задание № 13. Ответы 3 и 4.
Задание № 14. Физический размер экрана и угол обзора.

4. Программные средства реализации информационных процессов

Задание № 1. Ответ 3.
Задание № 2. Файл.
Задание № 3. Ответ 2.
Задание № 4. 2 кластера.
Задание № 5. Ответ 3.
Задание № 6. Ответ 2.
Задание № 7. Все файлы, кроме файлов с расширением "bak".
Задание № 8. *.txt, *.doc, *.docx.
Задание № 9. Драйверы.
Задание № 10. Векторный и растровый.
Задание № 11. JPG.
Задание № 12. Компиляция, компоновка.
Задание № 13. Язык программирования низкого уровня.
Задание № 14. Семейство языков «Си».
Задание № 15. Prolog.
Задание № 16. Ответы 2 и 3.
Задание № 17. Ответы 2 и 4.
Задание № 18. Ответ 2.

Задание № 19. Ответы 2, 3, 4.
Задание № 20. Интерпретатор воспринимает исходную программу на исходном языке и выполняет еѐ построчно, не создавая исполняемого модуля.

5. Модели решения функциональных и вычислительных задач

Задание № 1. В модели учитываются только существенные стороны изучаемого объекта, явления или процесса. *Задание* № 2. Формализация.
Задание № 3. Эвристические приѐмы.
Задание № 4. Использование модели знаний для решения задачи из конкретной проблемной области.
Задание № 5. Нельзя.
Задание № 6. Методы искусственного интеллекта.
Задание № 7. Вербальная информационная модель.
Задание № 8. Формальная математическая модель.
Задание № 9. Материальная точка.
Задание № 10. Постановка задачи.
Задание № 11. Модели данных.
Задание № 12. 1C; 2A; 3B; 4D.
Задание № 13. 1D; 2A; 3B; 4C.
Задание № 14. Ответы 1 и 3.
Задание № 15. Статические и динамические.
Задание № 16. неформализованная процедура, сокращающая количе-ство шагов поиска решения.
Задание № 17. Материальная опытная модель.
Задание № 18. Регрессионное, альфа-тестирование, бета-тестирование.

6. Алгоритмизация и программирование

Задание № 1. Дискретность, понятность, детерминированность, результативность.
Задание № 2. X=1, если A=B и C<D;
X=2, если A=B и C>=D;
x=3, если A<>B.
Задание № 3. X=4; Y=8.
Задание № 4. D = 3.
Задание № 5. Цикл с предусловием.

Задание № 6. X = 2.
Задание № 7. Ответ 2.
Задание № 8. 4 раза.
Задание № 9. X=5 X=712.

7. Технологии программирования

Задание № 1. Ответы 1 и 2.
Задание № 2. Синтаксическая ошибка.
Задание № 3. Семантическая ошибка.
Задание № 4. Линейные, циклические,
ветвления *Задание* № 5. См. п.7.1.
Задание № 6. Ответы 1, 2, 3, 7, 8.
Задание № 7. Объектно-ориентированной.
Задание № 8. Тело подпрограммы.
Задание № 9. Принципы 1 и 2.
Задание № 10. См. п. 7.3.

8. Локальные и глобальные сети ЭВМ. Методы защиты информации

Задание № 1. Local Areal Net, Metropolitan Areal Net, World Areal Net.
Задание № 2. Доменная система имѐн.
Задание № 3. inf02000/01-02.
Задание № 4. Оптоволоконный кабель.
Задание № 5. Снежинка.
Задание № 6. Wi-Fi, IrDa.
Задание № 7. См. п. 8.3..
Задание № 8. Набор правил для обмена информацией в сети.
Задание № 9. http://.
Задание № 10. 7 уровней.
Задание № 11. HTTP, SMTP, FTP, DNS, POP3.
Задание № 12. На сетевом..
Задание № 13. Мост
Задание № 14. Удостоверение истинности отправителя и целостности сообщения.
Задание № 15. См. п. 8.7.
Задание № 16. См. п. 8.9.
Задание № 17. См. п. 8.10

СПИСОК ЛИТЕРАТУРЫ

1. **Макарова Н.В., Волков В.Б.** Информатика: Учеб. для вузов. – СПб.: Питер, 2013. – 576 с. (для бакалавров)

2. Информатика: Учеб. пособие / Под ред. Б.Е. Одинцова, А. Н. Романова. 2-е изд., перераб. и доп. – М.: Вузовский учебник; ИНФРА–М, 2012. – 410 с.

3. **Гуриков С.Р.** Информатика: Учеб. – М.: Форум; ИНФРА–М, 2014. – 464 с. – (Высшее образование. Бакалавриат)

4. **Сергеева И.И.**, **Музалевская А.А.**, **Тарасова Н.В.** Информатика: Учеб. 2-е изд., перераб. и доп. – М.: Форум; ИНФРА–М, 2012. –384 с. – (Профессиональное образование)

Интернет-источники

5. Википедия: свободная энциклопедия: http://ru.wikipedia.org/

6. Система дистанционного обучения НИУ ИТМО: http: // de.ifmo.ru/

7. Электронный учебник: http://256bit.ru/informat/

8. Электронный учебник: http://book.kbsu.ru/

9. Информация и методы ее измерения, формулы и т. п.: http://marklv.narod.ru/inf/izminf.htm

10. Разные варианты классификации моделей:

11. Основные понятия информации и других разделов дисциплины «Информатика»: http://sesia5.ru/blok/1/11-1.htm

12. Краткое изложение основных разделов информатики: http://informatika.sch880.ru/p26aa1.html

13. Разъяснение основных терминов информатики: http://informatique.org.ru/svoystva-informacii.php

СОДЕРЖАНИЕ