Правила техники безопасности при работе в компьютерном классе.

При работе на ЭВМ требуется соблюдать ряд правил в целях избежания поражения электрическим током или выхода из строя самого компьютера.

1. Будьте внимательны, дисциплинированны, осторожны, точно выполняйте указания учителя.
2. Не трогайте провода и розетки.
3. Не прикасайтесь к экрану монитора, задним частям системного блока и монитора.
4. Не включайте ЭВМ без разрешения учителя.
5. Запрещается приносить с собой или находиться в верхней одежде.
6. Запрещается работать за ЭВМ влажными или грязными руками.
7. Не держите на рабочем месте предметы, не требующиеся при выполнении задания.
8. При работе с ЭВМ необходимо, чтобы экран находился на расстоянии 55 - 65 см от глаз, перпендикулярно линии взора.
9. Во время работы с ЭВМ запрещается перемещение по классу. Избегайте резких движений.
10. Нажимать на клавиши и кнопки разрешается только пальцами.
11. Запрещается класть любые предметы на различные части ЭВМ.
12. Запрещается использовать при работе на ЭВМ дискеты без разрешения учителя.
13. Запрещается держать при себе во время проведения занятий мобильные телефоны и другие электронные устройства, за исключением ЭВМ на стационарном рабочем месте учащегося.
14. При обнаружении неисправностей немедленно сообщить об этом учителю.

Действия при неправильной работе программ или “ зависании” компьютера

- При работе на ЭВМ могут возникнуть непредвиденные ситуации - блокирование клавиатуры, неверные действия программы - иными словами “зависание”. В таких случаях действуют следующим образом: нажать CTRL + C для аварийного завершения программы в ОС Linux или CTRL + ALT + DEL в ОС Windows, чтобы выгрузить "зависнувшую" программу из памяти и продолжить работу;
- если это не помогает, то надо вызвать перезагрузку ОС - нажать CTRL + ALT + DEL.
- если и это не помогло - нажать кнопку RESET на системном блоке компьютера.
- при отсутствии кнопки RESET выключить, а затем включить компьютер.

1. Информация и информационные процессы.

Информация - это сведения, передаваемые в форме знаков или сигналов.

Примеры информации:	Форма
Сведения из газет, книг, журналов	Текстовая
Величины (рост, вес, скорость, даты и т.д.)	Числовая
Рисунки, фотографии, картины, чертежи	Графическая
Сведения при разговоре, музыкальный трек	Звуковая
Данные на компьютере, телефоне, планшете	Электронная (закодированная)
Информация, воспринимаемая человеком	Зрительная, слуховая, тактильная, вкусовая, обонятельная

Свойства информации.

Свойство	соблюдение	Не соблюдение
Объективность	На улице +25°	На улице тепло
Понятность	Привет	你好
Полезность	Ваше расписание уроков	Расписание из другой школы
Достоверность	2*2=4	Столица России - город Владимир
Актуальность	Сегодня вечером возможен дождь	Месяц назад была ясная погода
Полной	Сеанс начинается 20.09.16 в 18.30	Адрес: г. Владимир

Информационные процессы.

Воздействие на информацию называют информационным процессом.

Информационные процессы - процессы приема, передачи, хранения и обработки информации в общении людей, в живых организмах, технических устройствах.

Передача - отправить, рассказать, жестикулировать и т.д.

Получение - услышать, увидеть, почувствовать, загрузить и т.д.

Хранение - записи в тетради, память мозга и т.д.

Обработка - решение задачи, поиск сведений, создание нового рисунка, конструирование и т.д.

Информатика - это наука, изучающая организацию информационных процессов при помощи компьютеров.

Компьютер - является универсальной (многофункциональной) машиной для обработки информации.

ЭВМ действует по заложенной в нее программе. Поэтому создание новых программ расширяет возможности компьютера без технического вмешательства. Такие качества компьютеров быстро привели к проникновению ЭВМ в различные сферы человеческой деятельности. Это, в свою очередь, вызвало бурный рост новых технологий в таких областях как:

- космонавтика
- ядерная энергетика
- промышленность
- медицина
- сервисное обслуживание
- телекоммуникации
- бизнес
- кинематография
- образование и т.д.

Вывод:

Таким образом, каждый грамотный человек должен понимать сущность обработки информации на ЭВМ и уметь выполнять эту обработку для достижения конкретной цели.

Вопросы

1. Примеры и формы информации.
2. Что такое информационные процессы? Привести примеры. Дать определение науки информатики.
3. Объяснить причины появления и бурного развития ЭВМ.

"История появления электронных вычислительных машин"

Слово компьютер означает “вычислитель”, т.е. устройство для вычислений. Потребность в вычислительной технике возникла очень давно. Еще в V в. до н.э. греки и египтяне использовали абак - устройство похожее на русские счеты.

В 1642 г. Блез Паскаль изобрел устройство, механически умеющее складывать, а в 1673г. Готфрид Вильгельм Лейбниц сконструировал арифмометр, механически выполняющий четыре арифметических действия. Начиная с XIX в. арифмометры получили очень широкое применение. Существовала специальная профессия - счетчик -

человек быстро и точно соблюдавший определенную последовательность инструкций для арифмометра.

Наряду с вычислительными устройствами развивались механизмы для автоматической (без участия человека) работы по заданной программе. В шарманку помещали диск с штырьками - в зависимости от положения штырьков звучала та или иная мелодия. В ткацком станке Жаккарда узор ткани задавался с помощью перфокарт (картонный лист с пробитыми дырочками). Для смены узора достаточно было поставить перфокарту с другим расположением дырочек .

В п.п. XIX в. англичанин Чарльз Бэббидж решает соединить идею механической вычислительной машины Лейбница с идеей автоматической работы по программе. Он разрабатывает проект своей машины: память для хранения 1000 чисел по 50 десятичных знаков; арифметические операции выполняются в соответствии с программой, записанной на жаккардовых перфокартах. В программе можно было задать повторение группы арифметических операций, а также выполнение при определенном условии. Беббидж не смог довести до конца свою работу - машина оказалась сложна для развития промышленности того времени. Однако идея была верна и в 1943 г. Американец Говард Эйкен на идеях работы Беббиджа и электромеханических реле смог построить машину под названием “Марк-1” .Аналогичную машину в 1941 г. построил Немец Конрад Цузе.

К тому времени потребность в автоматических вычислениях (в баллистике, криптографии) значительно возросла. Группа американцев под руководством Джона Мочли и Проспера Эккерта начала конструировать машину, подобную машине Эйкена и Цузе, но на основе электронных ламп. Их машина, названная ENIAC работала в 1000 раз быстрее, чем «Марк-1», однако для задания программы работы требовалось несколько часов или даже дней нужным образом соединять провода. Требовалось упростить задание программы. В 1945 году к работе был привлечен знаменитый математик Джон Фон Нейман. Он просто и четко формулирует основные принципы работы универсальных вычислительных машин, т.е. компьютеров. По этому принципу первый компьютер был построен в 1949 г. английским исследователем Морисом Уилксом.

Принципиальная схема Д.Ф.Неймана

Для того чтобы устройство было универсальным и эффективным, Нейман предложил использовать следующие устройства:

- арифметическо-логическое устройство, выполняющее арифметические и логические операции;
- устройство управления, которое организует процесс выполнения программ;
- запоминающее устройство, или память для хранения программ и данных;
- внешние устройства для ввода-вывода информации.

Связи между устройствами компьютера показаны на схеме:

- управляющие связи
- информационные связи

Принцип действия:

1. Вначале с помощью какого - либо внешнего устройства в память компьютера вводится программа.
2. Устройство управления считывает содержимое ячейки памяти, где находится первая команда программы, и организует ее выполнение.
3. Результаты записываются в память или на внешнее устройство.
4. Устройство управления считывает содержимое следующей ячейки памяти и т.д.

Основной принцип Неймана -

работу ЭВМ определяет программа, находящаяся в ее электронной памяти.

Именно этот принцип предопределил универсальность ЭВМ. Нет необходимости для решения новой задачи создавать новое устройство - достаточно поместить в оперативную память соответствующую программу. Причем загрузка программ будет происходить быстро в отличие от механических способов задания программ первых вычислительных машин типа ENIAC .

Современные ЭВМ используют в своей основе схему Неймана с некоторыми отличиями - арифметическо-логическое устройство и устройство управления объединены в единое устройство - центральный процессор; выполнение программ может прерываться при поступлении неотложных сигналов от внешних устройств – это нужно для обработки действий пользователя (н-р: ввод с клавиатуры и работа мышкой), вывода информации на принтер и т.п..

Вопросы

1. Назовите основные вехи в истории появления вычислительных машин.
2. Что такое перфокарта?
3. Изобразите и расскажите принцип действия схемы Д.Ф.Неймана.
4. Какие особенности имеют сегодняшние компьютеры в сравнении со схемой Неймана?

Представление информации в компьютере.

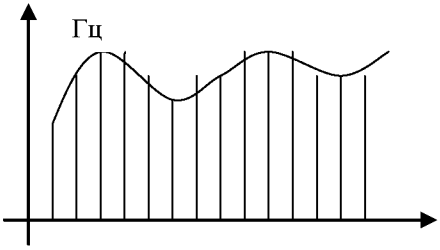
Существует два типа информации – аналоговая и дискретная. Аналоговая информация передается с помощью непрерывного сигнала. К таким сигналам можно отнести звук, изображения, запахи. Дискретная информация представляет собой набор отдельных знаков и сигналов: числа, текст, азбука Морзе.

Компьютер может обрабатывать только числовую информацию, поэтому аналоговая информация должна приводится к дискретному виду, а затем кодироваться.

Примеры:

Как же кодируются звуковая, текстовая и графическая информация. Вот основные принципы представления информации.

Звуковая информация. Непрерывная звуковая волна разбивается на одинаковые временные промежутки (преобразуется к дискретному виду) и для каждого участка измеряется интенсивность звука на определенных частотах, представляя результаты в числовой форме. С помощью программ можно выполнить преобразование полученной информации (н-р наложить звуки разных источников). Можно осуществить и обратную операцию.



К **текстовой информации** (изначально являющейся дискретной) применяется следующий подход - каждому символу (буквы алфавита, знаки препинания, знаки математических операции и др.) ставят в соответствие порядковый номер. А номер это есть число. При вводе символа в ЭВМ он кодируется своим числом, при выводе декодируется в изображение символа.

Графическая информация: изображение можно разбить на квадраты небольшого размера (дискретизация), каждый из которых может быть закрашен только одним цветом из палитры с конкретным количеством цветов. Такой отдельный квадрат называется **пиксель**. Каждый цвет в палитре пронумеровывается. В итоге на ЭВМ вместо изображения сохраняются номера (кода) цветов каждого пикселя последовательно по строкам. При выводе на экран пиксели окрашиваются в соответствии с кодами своих цветов. В силу невозможности человеческого глаза воспринимать очень мелкие детали, изображение, состоящее из точек, будет восприниматься как единое целое.

Каким же образом представляется информация в ЭВМ? Учитывая, что компьютер работает от электрического тока, кодировать информацию технически проще в виде двух различных сигналов: включено - выключено, намагничено - не намагничено, высокое - низкое напряжение и т.д. Принято обозначать один сигнал цифрой 0, а другой цифрой 1. Таким образом, вся информация в компьютере кодируется последовательностью единиц и нулей или числами в **двоичной системе счисления**. Любые системы счисления равноправны по своему смыслу - это записи каких-либо величин - количества денег, длина пути, и т.д. Люди используют десятичную систему счисления в силу исторически сложившихся обстоятельств - десять пальцев на руках. В древности существовала также шестидесятеричная система - ей мы обязана измерением времени, а на ЭВМ используется двоичная система в силу конструктивных особенностей.

Число в любой системе можно разложить следующим образом:

$47092 = 2 + 9\cdot10 + 0\cdot100 + 7\cdot1000 + 4\cdot10000 = 2\cdot10^0 + 9\cdot10^1 + 0\cdot10^2 + 7\cdot10^3 + 4\cdot10^4$ из разложения можно видеть, что каждая цифра в числе в десятичной системе - это коэффициент при основании системы (десять) в соответствующей степени. Зададимся вопросом, что за число 11011 в системе счисления с основанием 2 , 3 и 4)? Будем действовать аналогично предыдущему случаю.

$1011_2 = 1\cdot2^0 + 1\cdot2^1 + 0\cdot2^2 + 1\cdot2^3 = 1 + 2 + 0 + 8 = 11_{10}$

$1011_3 = 1\cdot3^0 + 1\cdot3^1 + 0\cdot3^2 + 1\cdot3^3 = 1 + 3 + 0 + 27 = 31_{10}$

$1011_4 = 1\cdot4^0 + 1\cdot4^1 + 0\cdot4^2 + 1\cdot4^3 = 1 + 4 + 0 + 64 = 69_{10}$

Для того, чтобы осуществить обратный перевод (из десятичной системы в другую), надо разделить исходное число на основание другой

системы счисления нацело с остатком, затем получившееся частное вновь делим нацело на основание и получаем частное и остаток и т.д., пока в очередном не получим значение 0 . Если записать остатки по порядку , начиная с последнего, то получим запись числа в новой системе счисления. Переведем число 238 из десятичной системы в системы с основанием 2 , 5 , 8:

238 : 2 = 119 остаток 0
119 : 2 = 59 остаток 1
59 : 2 = 24 остаток 1
24 : 2 = 12 остаток 0
12 : 2 = 6 остаток 0
6 : 2 = 3 остаток 0
3 : 2 = 1 остаток 1
1 : 2 = 0 остаток 1

238 : 5 = 47 остаток 3
47 : 5 = 9 остаток 2
9 : 5 = 1 остаток 4
1 : 5 = 0 остаток 1

238 : 8 = 29 остаток 6
29 : 8 = 3 остаток 5
3 : 8 = 0 остаток 3

10	2	$16=2^4$
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	Д
14	1110	E
15	1111	F

$238_{10} = 11000110_2$ $\qquad$ $23810 = 1423_5$
$23810 = 356_8$

Сложение также просто **1 + 0 = 1** , **0 + 1 = 1** ,**1 + 1 = 10** (т.к. $2_{10} = 10_2$) .Например:

```
  x 101101          101101₂ = 45₁₀
      1001            1001₂ = 9₁₀
    101101          45₁₀ • 9₁₀ = 405₁₀
+   000000
   000000
  101101
  110010101         110010101₂ = 405₁₀
```

Вычитание: **1 – 0 = 1, 1 – 1 = 0, 10 – 1 = 1.** Например:

```
  101101        1001101:101=1001 или 1001000:11=11000
- 1011
  100010
```

Основной недостаток - более длинная запись чисел в двоичной системе. Поэтому часто используют для обозначения кодированной информации шестнадцатеричную систему счисления. В этой системе для записи чисел используются все десять цифр и несколько латинских букв:

0 , 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , A , B , C , D , E , F

Цифра A имеет значение 10 , B - 11 , C - 12, D - 13 , E - 14 , F - 15 . Например , $251_{10}=FB_{16}$.

10	2	$8=2^3$
0	000	0
1	001	1
2	010	2
3	011	3
4	100	4
5	101	5
6	110	6
7	111	7

Принцип перевода из 16-ричной системы счисления в 10-тичную и наоборот аналогичен переводу в 2-чной системе счисления.

Для более быстрого перевода из 16-ричной в 2-чную и обратно, минуя 10-чную, используется так называемая таблица взаимосвязи 10-чной системы счисления с системами счисления, основание которых является степенью 2.

Например: $0101.1000.1110.0011_2=58E3_{16}$ или $B7AF12_{16}=101101111010111100010010_2$.

Аналогично рассматривается и 8-ричная система счисления.

Вопросы

1. Назовите примеры различного представления информации из жизни.
2. Почему на ЭВМ информация кодируется двоичными числами?
3. Перевести числа из одной системы счисления в другую: 49_{10}, 76_{10}- $?_2$; 10101011_2, 1111111_2 -$?_{10}$
4. Составить таблицу умножения в шестнадцатеричной системе счисления
5. Написать, в чем состоит принцип кодирования текстовой информации.
6. Написать, в чем состоит принцип кодирования графической информации.
7. Написать, в чем состоит принцип кодирования звуковой информации.

Единицы измерение информации

Бит - минимальная единица измерения информации. Бит может принимать только два значения - 1 и 0. Таким образом, **бит** – это минимальная ячейка памяти, которая разбивается на 1 и 0 (техническое понятие бита).

Но компьютер не определяет смысловую ценность информации. Что это значит? Когда мы бросаем монету, мы можем определить два равновероятных возможных события: «орел» или «решка». Но реализуется только одно. Когда монета брошена, то остается одно выпавшее событие, т.е. наше незнание уменьшилось в два раза. Таким образом, можно дать смысловое понятие бита. За единицу количества информации принято такое количество информации, которое содержит сообщение, уменьшающее неопределенность знания в два раза. Такая единица названа **бит**. Если мы обозначим количество вариантов или возможных событий – n, количество информации – i, формула Шеннона будет выглядеть следующим образом $2^i=n$.

Как правило, ЭВМ работает не с отдельными битами, а с группами бит: **8 бит = 1 байт (b)**

В одном байте можно получить $\mathbf{2^8 = 256}$ различных комбинаций битов. Этого количества достаточно для кодировки всех используемых символов на ЭВМ. Поэтому 1 символ занимает в памяти 1 байт (можно

одинаково сказать “объем сообщения 150 байт” или “ объем сообщения 150 символов”).

Существуют и более крупные единицы информации:

1 килобайт (Kb) = 2^{10} b = 1024 b

1 мегабайт (Mb) = 2^{20} b = 1048576 b

1 гигабайт (Gb) = 2^{30} b ≈ 1 млрд. b

1 террабайт(Tb)=2^{40}b

Для сравнения 1 Mb ≈ 400 страниц текста.

Единицы измерения скорости передачи информации.

1 бод=1бит/сек – такое количество информации, передаваемое за 1 секунду.

Скорость передачи достаточно важно знать для решения задач в реальном времени. Скорость загрузки программ определяется скоростью чтения информации с носителей. Иногда кажется, что компьютер «задумывается», это значит, что у него недостаточно большая скорость передачи информации.

Скорость важна и для коммуникаций (телефонные линии отличного качества – 50кбит/сек, в старых – 30-35 кбит/сек).сейчас очень быстро развивается оптоволоконное производство (200 Мбит/сек), в нем достаточно малые потери, очень гибкие, проводит сотни потоков (в телефонном кабеле – 1 поток).

"Устройства, входящие в состав персональной ЭВМ"

Основные блоки.

I. СИСТЕМНЫЙ БЛОК.

В системном блоке находятся внутренние устройства компьютера. Состав:

1. **Основная электронная плата (материнская плата).** Обеспечивает подключение и взаимодействие различных внутренних и внешних устройств.

На ней расположены **системные магистрали передачи данных** (шины), которые осуществляет обмен данными с другими устройствами, и слоты, в которые устройства полключаются. На скорость передачи данных влияет разрядность шины и тактовая частота. Разрядность шины - количество бит информации передаваемой за 1 операцию. Разрядность влияет на пропускную способность шины, тем самым на скорость передачи информации. Бывают 16-и, 32-х, 64-х разрядные. Тактовая частота – это количество элементарных операций, производимых за 1 с, измеряется в мегагерцах. Обычно материнская плата работает на частотах **33 Мгц , 66 Мгц , 100 Мгц , 200 Мгц и т.д.**

2. **центральный микропроцессор** (“мозг компьютера”) - осуществляет **выполнение программ** и управление работой остальных устройств компьютера.

Скорость его работы определяет быстродействие компьютера. Микропроцессоры отличаются друг от друга моделью (поколением), тактовой частотой и разрядностью. Модели Intel - совместимых процессоров называются **86 , 286 , 386 ,486 , Pentium , Pentium II, Pentium III, Pentium IV, Core**. Тактовая частота измеряется в мегагерцах и получается умножением частоты материнской платы на какое-нибудь число (**66 Мгц , 200 Мгц, 300 Мгц, 1000 Мгц, 1700 Мгц, 2000 Мгц (2 Ггц), 3Ггц и т.д.).** Большая тактовая частота обеспечивает большую производительность процессора. Также на скорость процессора влияет его **разрядность** - количество бит информации, обрабатываемое процессором за одну операцию. Персональные ЭВМ обычно оснащены **8 , 16, 32 и 64-х разрядными** процессорами. Процессоры имеют собственную небольшую сверхбыструю память - **кэш-память**. Она содержит информацию из наиболее часто используемых участков оперативной памяти. Поэтому процессор может не обращаться к ОЗУ, а работать с данными кэш-памяти, что в несколько раз быстрее. Тем самым общее быстродействие ЭВМ повышается. Обычный размер кэш-памяти 128 Kb, 256 Kb или 512 Kb. Наличие большой кэш-памяти в процессоре увеличивает его скорость работы более, чем в 2 раза при одинаковой тактовой частоте.

3. **оперативная память** (оперативное запоминающее устройство - **ОЗУ**) - хранит выполняемые в данный момент программы, их данные и полученные результаты.

Называется “оперативная” потому, что работает очень быстро (≈8 наносекунд ($8 \cdot 10^{-9}$) на одну операцию , процессору не приходиться долго ждать чтения или записи данных. При выключении ЭВМ вся информация в оперативной памяти стирается. Объем оперативной памяти влияет на скорость работы программ на компьютере. Так, один и тот же компьютер с 128 Mb ОЗУ будет выполнять программы медленнее, чем с 256 или 512 Mb оперативной памяти. Это обусловлено тем, что если часто используемые данные хранятся в процессе работы программы в ОЗУ, а не на диске, то скорость обращения к ним намного быстрее. В настоящее время распространены следующие объемы памяти – 1024 Mb или 1 Gb, 2 Gb, 4 Gb .

4. **постоянная память** (**ПЗУ**) - содержит программы для подготовки ЭВМ к работе.

Эта память вступает в работу сразу при включении компьютера. Процессор считывает из нее программы тестирования устройств и передает управление загрузчику операционной системы. Эти данные заносятся в ПЗУ заводом - изготовителем и не могут быть изменены.

5. **Жесткий** (несъемный) **магнитный диск** (винчестер). Обеспечивает хранение, запись и чтение больших объемов информации.

Информация сохраняется и при выключенном компьютере. Отличается от дискет и компакт-дисков надежностью хранения информации, значительно большей емкостью (80 Gb, 160 Gb , 250 Gb , 1000 Gb и т.д.) и высокой скоростью чтения и записи информации – зависит от скорости вращения (5400, 7200, 10000 об/мин).

6. **Дисководы** для чтения и записи информации при работе различными дисками:

дискеты (1,44 Mb), CD(700 Mb) или DVD (4,5 -7,6 Gb).

7. **Блок питания** - обеспечивает преобразование переменного тока сети в постоянный ток низкого напряжения (н-р 5 V) . От блока питания берут энергию все остальные внутренние устройства.
8. **Динамик** - обеспечивает вывод звуковой информации (моно).
9. **Адаптеры** и **контроллеры** - обеспечивают подключение к компьютеру различных устройств.

Адаптеры вставляются в разъемы (слоты) шины, а контроллеры впаяны в материнскую плату. К их разъемам подсоединяются различные внутренние и внешние устройства.

- **Видеоадаптер** - микросхема, обеспечивающая хранение переданной процессором цифровой "картинки" в специальной памяти - называется **видеопамять**, и передачу ее содержимого на монитор 60-100 раз за

секунду без участия процессора. Объем видеопамяти (128 Mb , 512 Mb, 1024 Mb и т.д.) определят цветовую и качественную характеристики изображения. Чем больше объем, тем больше цветов и большую четкость можно будет получить на экране монитора. В настоящее время распространены так называемые ускорители, которые имеют свой собственный процессор, который берет на себя необходимые вычисления и вывод графики на экран монитора, тем самым разгружая центральный процессор.

- **стереозвуковой контроллер** - для подключения колонок, микрофона и синтезатора и джойстика.
- **сетевой контроллер** - обеспечивает передачу данных между компьютерами по специальному кабелю.
- **контроллер клавиатуры** - обеспечивает передачу сигналов от нажатых клавиш в шину данных.
- **контроллеры дисков** - подключение жесткого диска, дисководов для дискет и лазерных компакт-дисков.
- **контроллеры ввода-вывода.** Имеют порты (разъемы), к которым присоединяются кабели дополнительных устройств. Порты бывают следующих типов: **параллельные (LPT1-LPT4)** - к ним подключаются принтеры; **последовательные (COM1 - COM4)** - к ним подключаются манипулятор “мышь”, модем; **игровой порт** - к нему подключается джойстик. Некоторые устройства могут подключаться и к параллельным, и к последовательным портам. Параллельные порты работают быстрее.

10. **Вентилятор и радиатор** – радиатор забирает тепло у нагревающихся устройств ЭВМ (процессор, видеокарта, блок питания, жесткий диск и т.д.), а вентилятор охлаждает. Некоторые процессоры объединены с вентилятором и называется – бокс.

К системному блоку подсоединяются все внешние устройства

II. КЛАВИАТУРА

Она предназначена для ввода символьной информации и управляющих действий. Классическая клавиатура имеет 100-102 клавиши.

В центре клавиатуры расположены алфавитные клавиши. Рядом находятся управляющие клавиши. Значения управляющих клавиш:

- **Enter** - вводит строку на обработку ЭВМ. Строкой может являться название загружаемой программы, выбранный пункт меню или запрос
- **Tab** - перемещает курсор на фиксированное число позиций влево или переход курсора между экранными окнами или вариантами ответов на запрос.

- **Esc** - позволяет отменить какое - либо последнее действие или завершить выполнение программы.
- **F1-F12** - функциональные клавиши. В разных программах могут иметь разные значения. Некоторые распространенные значения: **F1** - вызов справки по действующей в данный момент программе, **F2** - запись данных в файл, **F3** - чтение данных из файла, **F10** - вызов меню или завершение программы .
- **Ctrl, Alt** - позволяют изменять значения других клавиш. Для этого эти клавиши прижимают и нажимают другие (обычно обозначается **Ctrl + O , Alt + F4**) .
- **Pause** - приостанавливает действие программы. Возобновить работу можно нажатием, например, пробела.
- **Break** - при нажатой клавише **Ctrl** позволяет аварийно завершать некоторые программы. Расположена эта клавиша совместно с **Pause**.
- **Print Screen** – делает «фотографию» содержимого экрана в буфер обмена.
- **Num Lock** - переключает режимы работу цифровой панели клавиатуры (находится справа). Если одноименный индикатор включен - режим цифр, выключен - режим управляющих действий.

КЛАВИШИ ПРИ РАБОТЕ С ТЕКСТОМ

- **Enter** - При наборе текста переводит курсор в начало следующей строки, в некоторых случаях разбивает строки на две, вставляет пустую строку в тексте.
- **Caps Lock** – переключает режим ввода строчных или прописных символов, т.е. нижний или верхний регистры. Определить регистр можно по **индикатору Caps Lock**. Если он горит – верхний регистр (заглавные буквы), не горит – нижний (строчные буквы).
- **Shift** - позволяет временно переключить регистр ввода букв, а также вводить символы, расположенные в верхних углах клавиш. Для этого **Shift** надо прижать и нажать нужную клавишу.
- **Delete** - удаляет символ в позиции курсора. В некоторых случаях может соединять две строки в одну, при этом курсор должен стоять в конце строки.
- **Backspace (бэкспейс)** - удаляет символ слева от курсора, при этом курсор сдвигается в позицию удаленного символа. В некоторых случаях может соединять строки в одну, при этом курсор должен стоять в начале той строки, которую необходимо поднять. Находится **Backspace** над клавишей Enter. Часто вместо названия есть только стрелка **←.**
- **Insert** - переключает режимы ввода текста ВСТАВКА/ЗАМЕНА. В режиме вставка вводимые символы вставляются в позиции курсора в

тексте, а в режиме замены вводимые символы «затирают» текст в позиции курсора.

- Клавиши ← ↓ ↑ → перемещают курсор на одну позицию в соответственном направлении.
- **Home** - перемещает курсор в начало строки.
- **End** - перемещает курсор в конец строки.
- **Page Up** - перемещает курсор на страницу вверх по тексту.
- **Page Down** - перемещает курсор на страницу вниз по тексту.

III. Монитор.

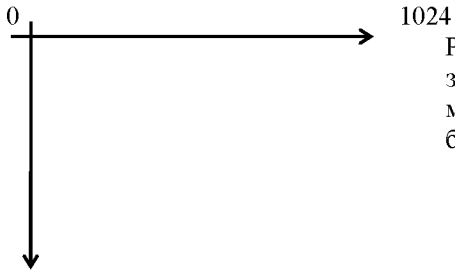
Он предназначен для вывода текстовой и графической информации полученной от видеоадаптера. Может работать в двух режимах - текстовом и графическом.

Текстовый режим. В этом режиме экран монитора условно разбивается на отдельные участки - клетки (по аналогии с тетрадным листом), где в каждой клетке может находиться один символ из кодировочной таблицы (см. приложение). Сочетание символов на экране и дает текстовое изображение. Обычно для символов имеется 25 строк по 80 “клеток” (столбцов):

столбцы ⇒

строки ⇓	**1**	**2**	**3**	**4**	**5**	**6**	**...**	**80**
1								
2								
3								
...								
25								

Графический режим. Предназначен для вывода на экран произвольной информации: графиков, рисунков, диаграмм и т.д. Текстовая информация в этом режиме может иметь различный вид (шрифты, размер и т.п.). В этом режиме монитора экран состоит из отдельных точек, каждая из которых может быть какого-то одного цвета из нескольких возможных. Такая точка называется **пиксель**. Количество пикселов по горизонтали и вертикали называется **разрешающей способностью** монитора. Например, 640 x 480 (VGA-мониторы), 800 x 600 , 1024 x 768 (Super-VGA мониторы).



Разрешающая способность не зависит от размера самого монитора, подобно тому, как большие и маленькие телевизоры

имеют на экране 625 строк развертки. Для максимального использования разрешающей способности необходимо иметь видеоадаптер с достаточным количеством видеопамяти.

768

Существенное влияние на качество изображения влияет размер пикселя. Чем меньше размер пикселя, тем четче изображение. Для мониторов с диагональю экрана 14 дюймов (35,5 см) для разрешающей способности 640 x 480 размер пикселя должен быть не больше 0,39 мм, а лучше 0,31. Для разрешения 800 x 600 - 0,31 мм , а для 1024 x 768 - 0.28 мм или 0.25 мм. На качество изображения так же влияет и частота кадров. Например: у телевизора 60 Гц, а у монитора от 85 до 120 Гц.

Дополнительные устройства

1.Манипулятор "мышь".

Мышка предназначена для ввода управляющих действий в ЭВМ: движение курсора, нажатия на экранные «кнопки», выбор объектов и тд .

2. Дискеты.

Гибкие магнитные диски (дискеты) - предназначены:

- для хранения информации при выключенной ЭВМ;
- для переноса информации с одного компьютера на другой.

дискета имеет размер 3 1/2 дюйма, а объем памяти - 1,44 Mb. На сегодняшний день дискета считается устаревшими носителем.

3. CD и DVD

Компакт-диски обладают значительным преимуществом перед дискетами. Во-первых, они имеют большую емкость (650-800 Mb), во-вторых, более долговечны, т.к. для чтения или записи не требуется механических воздействий на поверхность диска, и в-третьих, скорость работы намного выше. DVD (цифровые видео-диски) изначально разрабатывались как носители видеоинформации, но учитывая цифровую природу хранимой на них информации, в настоящее время широко используются на компьютерах как еще более (в сравнении с компакт-диском) объемное хранилище информации (4,7 Гб), скоростной и надежный носитель.

4.Принтер.

Принтер (печатающее устройство) - предназначен для вывода текстовой и графической информации из компьютера на бумагу.

Обычно с помощью компьютера и принтера набираются и выводятся на бумаге все виды текстов, отчеты, документы и т.д.

Основные виды принтеров:

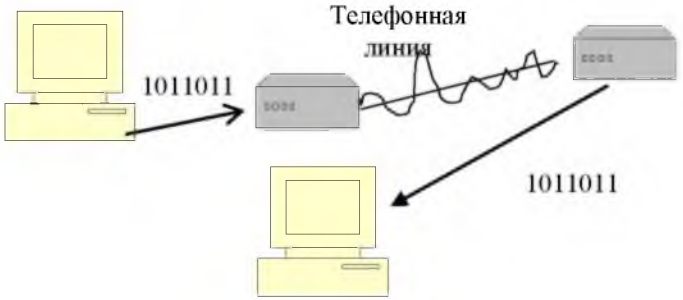
- **матричный**. Печатающая головка содержит вертикальный ряд иголок (9, 24 или 48). Головка движется вдоль печатаемой строки, а иглы по командам ЭВМ ударяют по бумаге через красящую ленту. Из получаемых точек формируется изображение букв и др. Скорость работы таких принтеров 10-60 сек/стр. Для повышения качества можно задать режим многократного прохода печатающей головки по одной строки. В таких случаях страница может выводиться более 5 мин. Минусы: медленная скорость печати, низкое качество печати. Плюсы: дешевый расходный материал.
- **струйный.** Изображение формируется микрокаплями специальных чернил, выдуваемых на бумагу с помощью сопел. Получаемое качество близко к типографскому. Удобен для цветной печати. Скорость 15 - 100 сек/стр. Дорогой расходный материал, заправки одного картриджа хватает примерно на 300 копий. Минусы: дорогой расходный материал, небольшое количество копий при одной заправки картриджа, засыхание чернил при не постоянной печати. Плюсы: цветная печать, большая скорость печати.
- **лазерный**. Обеспечивают наилучшее качество печати. Используют для печати принцип ксерографии - изображение наносится на бумагу со специального барабана, к которому электрически притягиваются частички специального порошка, нагретого до температуры 200 градусов. В отличие от ксерокса барабан электризуется лучом лазера по командам компьютера. Скорость 5-15 сек/стр. Дешевый расходный материал, заправки одного картриджа хватает примерно на 3000 копий. Высокое качество печати. Минусом является высокая стоимость самого принтера.

5.Сканер.

Он предназначен для ввода текстовой или графической информации в ЭВМ. Сканеры могут вводить в компьютер рисунки. Специальные программы могут распознавать на введенном рисунке текст.

6.Модем

Устройство передачи информации по телефонному медному проводу или оптоволоконному. Модем может выступать в двух видах: передающий модем кодирует информацию в волновой сигнал. Принимающий модем

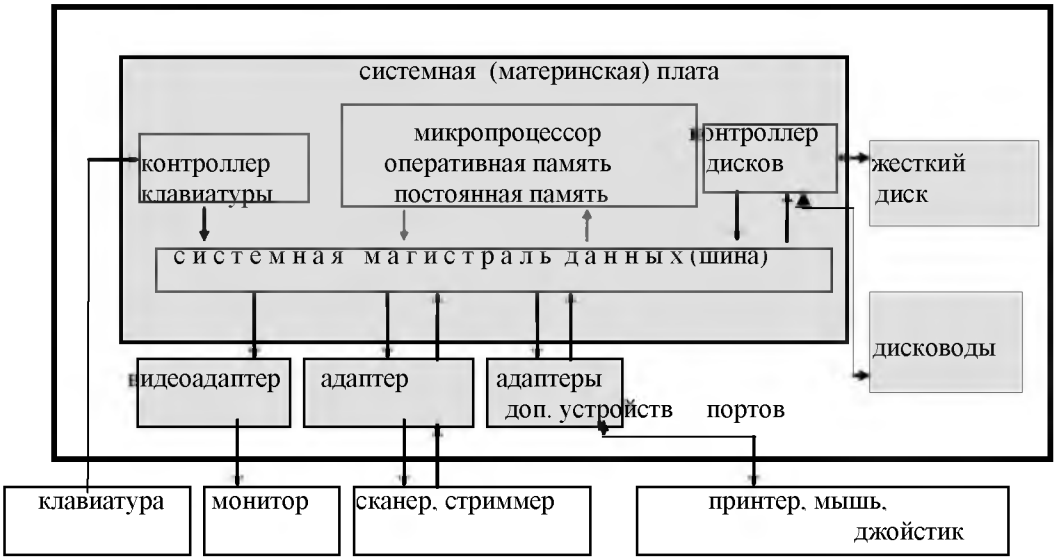


декодирует волновой сигнал в компьютерную информацию. Один моделирует, а другой демоделирует. Отсюда получается название – модем.

Таким образом, информация может быть передана от одного ЭВМ ко второму, потом к третьему и т.д. На этом основаны глобальные компьютерные сети (Интернет).

“Магистрально-модульный принцип взаимодействия устройств ЭВМ”

СИСТЕМНЫЙ БЛОК КОМПЬЮТЕРА



Ввод-вывод информации между микропроцессором и устройствами происходит посредством адаптеров и шины:

- для каждого внешнего устройства в ЭВМ имеется электронная схема, которая им управляет. Такая схема называется адаптер или контроллер.
- все адаптеры и контроллеры взаимодействуют с микропроцессором и оперативной памятью через системную магистраль передачи данных, которая обычно называется шина.

Таким образом, чтобы подсоединить к ЭВМ новое внешнее устройство, нет необходимости создавать новый компьютер, достаточно вставить адаптер этого устройства в свободный разъем шины, а к разъему самого адаптера подключить устройство. Чтобы заменить одно устройство (например, устаревший адаптер монитора) другим, надо просто вынуть старый адаптер из разъема и вставить новый. Можно заменить и материнскую плату, чтобы модернизировать компьютер. Такая

возможность изменения конфигурации компьютера стала одной из причин успеха распространения IBM - совместимых ЭВМ (90 % всех персональных компьютеров в мире).

Устройства по классам				
Обработка	Хранение	Ввод	Вывод	Связь
Центральный процессор Графический процессор	Оперативная память Жесткий диск Дискета CD, DVD	Клавиатура Мышь Сканер микрофон	Монитор Принтер Плоттер Колонки	Сетевой адаптер Модем

Контрольные вопросы.

1. Назначение микропроцессора, его характеристики.
2. Назначение клавиатуры.
3. Назначение адаптеров и контроллеров.
4. Назначение принтера, виды принтеров.
5. Назначение оперативной памяти, ее характеристики.
6. Назначение жесткого магнитного диска (винчестера).
7. Назначение монитора.
8. Назначение сканера.
9. Назначение постоянной памяти.
10. Назначение главной электронной (материнской) платы.
11. Назначение манипулятора "мышь".
12. Виды съемных дисков, их характеристики.
13. Назначение модема.
14. Назначение блока питания. Назначение дисководов, виды.

Программы для компьютеров

Компьютер - универсальное устройство для обработки информации. В отличие от телефона, магнитофона или телевизора, имеющих ограниченный и постоянный набор заложенных в них функций, компьютеры могут выполнять любые действия по обработке информации. Для этого необходимо составить для ЭВМ на понятном ей языке точную и подробную последовательность инструкций (называется **программа**), как надо обрабатывать информацию. Сам по себе компьютер не обладает никакими знаниями, все эти знания сосредоточены в выполняемых на компьютере программах. ри выполнении программ используются различные устройства ввода/вывода для получения или передачи информации. Именно программный принцип управления и обеспечивает универсальность ЭВМ.

Из всего этого следует, что для эффективного использования ЭВМ необходимо знать возможности и свойства основных используемых программ!

Все программы условно делят на три основные группы:

- **системные** программы - обслуживание устройств ЭВМ в процессе работы.
- **прикладные** программы - выполнение пользовательских задач.
- **инструментальные** системы - создание новых программ.

Системные программы

Основные разновидности системных программ:

- операционные системы.
- программы - оболочки операционной системы
- драйверы устройств.
- антивирусные программы.
- программы - упаковщики.
- вспомогательные программы (утилиты).

Прикладные программы

Основные разновидности системных программ:

- текстовые редакторы
- электронные таблицы
- системы управления базами данных
- обучающие программы
- игровые и другие.

Инструментальные программы (системы программирования)

Основные разновидности системных программ:

- Basic (Visual Basic)
- Pascal (Delphi)
- C
- Assembler

Операционная система

Операционная система (ОС) - это программа, которая загружается при включении компьютера.

Назначение: обеспечивает работу всех устройств компьютера и взаимодействие с пользователем.

Основная причина необходимости ОС в том, что при каждом сеансе работы на ЭВМ необходимо выполнять тысячи элементарных операций низкого уровня (в машинных кодах) для работы с устройствами и управления ресурсами ЭВМ. Например, дисковод “понимает” только элементарные операции, как включить-выключить двигатель, установить читающие головки на определенный цилиндр, прочесть информацию с

дорожки диска в компьютер. Поэтому для выполнения такого несложного действия, как загрузка программы с диска необходимо выполнить тысячи операций по запуску команд дисководов, проверки их выполнения и т.д. Пользователю вводить такие команды долго и неудобно, причем при новом включении ЭВМ процесс должен заново повторяться. Операционная система скрывает от пользователя эти сложные и ненужные подробности, предоставляя ему удобные “человеческие “ средства управления компьютером.

Разновидности ОС:

- дисковая ОС – ДОС;
- Windows;
- ОС/2;
- Linux;
- UNIX.

Организация хранения данных на внешних носителях в ОС.

Файловая система — это методы и структуры данных, которые используются операционной системой для хранения файлов на диске или в его разделе.

Файл - это область на диске, имеющая имя.

В файлах может храниться различная информация: программы, тексты, документы, изображения, ведомости, музыка и т.д. Различают две категории файлов - текстовые и двоичные. Текстовые файлы предназначены для чтения и записи в них информации человеком. Двоичные файлы, как правило, содержат программы.

Каждый файл имеет обозначение, состоящее из двух частей: имя и расширение, разделенные точкой.

Примеры обозначений файлов: primer.$$$; prog.exe; plan.pc; readme; самостоятельная работа.doc.

Смысл имени файла аналогичен смыслу названия книги. Расширение, как правило, дополняет информацию о содержимом файла.

Все файлы условно делят на две группы – исполняемые и информационные.

Исполняемые файлы загружаются в оперативную память для выполнения компьютером

Расширения исполняемых (загрузочных, программных) файлов в OS Windows **exe** или **com**.

В Linux расширение может быть любым (даже отсутствовать).

Информационные файлы содержат информацию, используемую программами или созданную пользователем.

Обычно имеют характерные расширения:

doc, txt - файл содержит текст, документацию;
hlp - файл со справочной информацией;
pcx , gif ,bmp, jpg - файл содержит изображение;
bas ,pas , c, for , asm - файл с текстом программы на одном из языков программирования(Basic, Pascal, C, Fortran, Assembler);
mp3, wav, cda – музыкальные;
avi, mpg, vob, mpeg – видео-файлы;
bak, tmp - копия файла, сделанная перед его изменением и др.
rar, zip, cab – архивные файлы
sys – системный файл
html – файл с гипертекстовой информацией (Интернет-

При необходимости используют и любые другие расширения:

spisok.96
examen.11a
readme (без расширения)

Допустимо назначать неисполняемым файлам ничего не значащие расширения. Однако это зачастую осложняет ориентирование в информации на дисках. Дисковая операционная система не различает строчные и прописные буквы, т.е. обозначение **Doclad.Bio** , **doclad.bio** , **dOcLaD.BIO** подразумевает один и тот же файл, а, например, в OS Linux это три разных файла.

На одном диске могут храниться десятки тысяч файлов самых разных назначений. Поэтому для более легкой ориентации среди файлов в ОС используют систему каталогов (директорий, папок).

<u>Каталог (папка)</u> - это общее имя группы файлов и других каталогов.

Аналогом каталогов в повседневной жизни служат способы хранения книг в библиотеке, т.е. если просто хранить все книги в общей куче, то очень сложно при большом количестве книг находить нужную. Поэтому книги хранят группами под каким-либо общим названием (художественная литература, детективы, справочники и т.д.). При чем внутри раздела могут существовать подразделы (художественная литература - мировая и отечественная). Таким же образом разбивают на группы и файлы, которые имеют общее назначение - принадлежат одному программному комплексу, выполняют схожие действия, хранят информацию одинакового направления и т.д. В любом каталоге могут, кроме файлов находиться другие каталоги. Каталог, который находится в другом каталоге, называется **<u>подкаталог</u>**. В каждом подкаталоге могут быть другие файлы и подкаталоги и т.д.

Каталоги обозначаются также как и файлы, но расширение обычно не используют. На каждом диске существует главный, называют " **<u>корневой</u>** " каталог. Такой каталог имеет заранее определенное обозначение, («/» в

Linux (называется "слэш") или «\» в Windows (обратный "слэш"), которое не может быть изменено. Начиная с корневого каталога располагаются все остальные файлы и каталоги .

Каталог, с которым в данный момент работает пользователь, называется "рабочим" или «текущим». Часто схему каталогов на диске указывают схематично - в виде “дерева” - графического представления зависимости каталогов друг от друга. Каталоги, находящиеся в корневом каталоге («на стволе»), называются каталоги первого уровня, их подкаталоги – каталоги второго уровня и т.д. Каталоги одного уровня не могут иметь одинакового имени. Для разного уровня допускается. Цепочку связанных каталогов разного уровня называют "путь" или "адрес". Имена каталогов разделяются символом \ в Windows (например \ **SYSTEM \ ANTI_VIR \ ADINF**) или / в Linux (например **/USER / DOS / QBASIC**)

В Windows принято обозначение диски латинскими буквами с двоеточием

A: дисковод или дискета
B: дисковод или дискета
C: жесткий диск
D: жесткий диск или компакт-диск
E: жесткий диск или компакт-диск
F: жесткий сетевой диск и т.д.

Полное имя файла - это обозначение файла с указанием диска и пути каталогов к этому файлу. Пример:

H:\GAMES\POLE\ pole.exe

В Linux диски обозначаются также как файлы:

hda (или sda)	жесткий диск
cdrom	компакт-диск
hdc	флешка

В ОС используется принцип **“по умолчанию”**: если не указано имя диска - считается текущий диск; если не указан путь каталогов - считается текущий каталог.

ОС Linux.

Файл.

Концепция файла — ключевая в операционной системе Linux. Практически все моменты, связанные с данными, в том или ином виде представляются в виде файла или операций с файлами. Для Linux по большому счету все равно, с каким устройством или процессом взаимодействовать — система работает с файлом.

Файл содержит информацию в некотором формате. Для операционной системы - просто набор байтов.

Каталог.

Каталоги являются элементами иерархического дерева. Любой каталог может содержать файлы и подкаталоги. Каталог — это файл, содержащий список записей. Каждая запись состоит из номера индексного дескриптора (атрибута) и имени файла.

Файл устройства.

В операционной системе Linux доступ к устройствам осуществляется через специальные файлы. Такой файл является точкой доступа к драйверу устройства.

Канал.

Файлы этого типа используются для связи между процессами для передачи данных.

Ссылки.

Индексный дескриптор может быть связан с несколькими именами файлов. Дескриптор содержит поле, хранящее число, с которым ассоциируется файл.

Сокет.

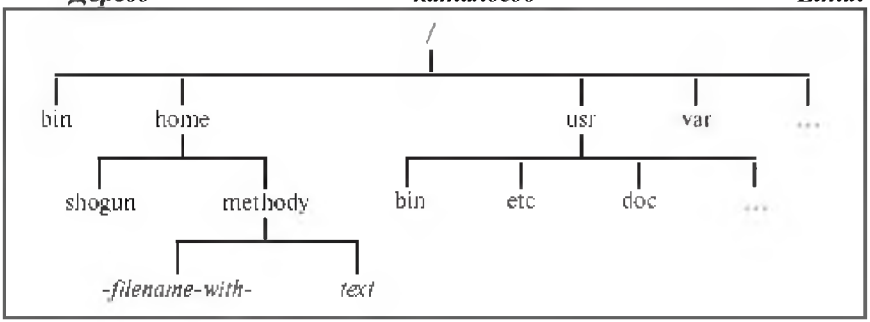
Сокеты предназначены для взаимодействия между процессами. Используются для доступа к сети TCP/IP.

Типы файловых систем:

Linux поддерживает большое количество типов файловых систем. Перечислим наиболее важные из них: Minix, Xia, Ext, Ext2, Ext3, Ext4, VFS, Proc, Sysfs, ReiserFS, Devfs.

Поскольку в Linux используется единое связанное дерево каталогов, то, в отличие от DOS/Windows, не существует такого понятия файловой системы, как диск. Все дисковые устройства (файловые системы) интегрируются в дереве каталогов в так называемые точки монтирования, в качестве которых выступают обычные каталоги.

Дерево каталогов Linux



Для того чтобы ориентироваться в Linux, необходимо хорошо представлять себе структуру и размещение каталогов и файлов.

Имя	Содержимое
/	Корневой (Root) каталог. Является родительским для всех остальных каталогов в системе
/bin	Содержит важные для функционирования системы файлы
/boot	Содержит файлы для загрузчика ядра
/dev	Хранит файлы устройств
/home	Пользовательские домашние каталоги
/lib	Важные разделяемые библиотеки и модули ядра
/media	Каталог для автоматически монтируемых устройств (дисковод, CD-ROM)
/opt	Дополнительные пакеты приложений
/proc	Точка монтирования псевдофайловой системы proc, которая является интерфейсом ядра операционной системы
/root	Домашний каталог для пользователя root
/sbin	Содержит важные системные исполняемые файлы
/tmp	Хранит временные файлы

Рассмотрим подробнее иерархию каталогов.

Корневой (Root) каталог. Точка монтирования всей файловой системы. Играет исключительно важную роль в процессе "жизнедеятельности" операционной системы. Для загрузки системы необходимо, чтобы в корневом разделе (корневой раздел в Linux - это аналог диска C: для DOS/Windows. Каталоги /usr, /opt, /var спроектированы так, что они могут размещаться на файловых системах, отличных от корневой.

Каталог /bin содержит важные исполняемые файлы, которые используются всеми пользователями, в том числе и администратором системы. Каталог /bin не должен содержать подкаталогов. Исполняемые файлы, от которых напрямую не зависит функционирование системы, рекомендуется размещать во вторичной иерархии — в каталоге /usr/bin.

Каталог /boot содержит все, что требуется для процесса загрузки, исключая файлы конфигурации.

Каталог /dev содержит файлы устройств или специальные файлы. Создание в каталоге /dev файлов устройств осуществляется с помощью предназначенной для этого утилиты makedev, находящейся в нем же.

Каталог /etc содержит конфигурационные файлы и каталоги, которые специфичны для данной системы.

Основные составляющие для загрузки компьютера.

1. **Базовая система ввода- вывода** (**Basic Input Output System - BIOS**). Находится в ПЗУ (постоянной памяти компьютера). Это обусловлено тем, что ЭВМ может работать только по программе, находящейся в ее памяти. В оперативной памяти в момент включения ничего нет, поэтому программа для подготовки компьютера к работе берется из BIOS .

Назначение:

- тестирование всех устройств при включении ЭВМ;
- вызов загрузчика;
- управление календарем;
- управление динамиком.

2. **Загрузчик** операционной системы - это небольшая программа, находящаяся в первом секторе системного (загрузочного) диска.

Назначение: загрузка с диска в оперативную память файлов ядра операционной системы.

3. Операционная система.

Назначение: обеспечивает работу всех устройств ЭВМ и диалог с пользователем.

Контрольные вопросы.

1. Прикладные программы: назначение и разновидности.
2. Дать определение файла.
3. Привести примеры расширений файлов пользователя (неисполняемых), описать их назначение.
4. Инструментальные программы: назначение и разновидности.
5. Дать определение каталога.
6. Привести пример пути к файлу;
7. Как называется и обозначается главный каталог на диске
8. Указать расширения программных (загрузочных, исполняемых) файлов, описать их назначение.
9. Системные программы: назначение и разновидности.
10. Дать определение, указать назначение и разновидности операционных систем.
11. Изобразить пример дерева каталогов.
12. Назначение базовой системы ввода-вывода (bios).

Популярные операционные системы

Дисковая операционная система (ДОС)

ДОС использовалась около 15 лет на нескольких поколениях персональных компьютеров. Основным достоинством этой системы была низкая требовательность к ресурсам компьютера, но пользователям нужно

было обладать определенной подготовкой. На современном этапе ДОС используется как средство восстановления работоспособности Windows.

Управление пользователем ДОС осуществляется через ввод команд (пользователь вводит команду - ДОС исполняет). Команда ДОС состоит из названия и, если нужно, параметров. Можно использовать как строчные, так и прописные символы (ДОС не различает). Ввод каждой команды завершается нажатием клавиши Enter. В одной из строк экрана находится **приглашение** ДОС к работе, за которым мигает курсор. Вид приглашения может задаваться пользователем. Строка, в которой находится приглашение, называется **командной строкой ДОС**. По "умолчанию" приглашение выглядит **диск>:A>**

Диск указывается текущий (рабочий).

Как правило, более удобно использовать приглашение вида **диск\путь>** с текущим диском и каталогом. Если текущий каталог является подкаталогом другого каталога, то указывается полный путь:

C:\WORD\TEXT> текущий каталог - подкаталог **TEXT** каталога **WORD**, расположенный на диске **C:**

H:\USERS\NOY\1995_96\DECTOBIN>

B:\>

A:\DOS> Такого вида приглашение точно указывает место на диске, с которым осуществляется работа.

Ввод команд.

- Для ввода команды следует набрать имя команды без ошибок на клавиатуре и нажать клавишу **Enter**.
- Клавиша **F3** вызывает в командную строку, ранее введенную команду.
- Клавиша **Esc** очищает командную строку.
- Клавиша управления курсора ➔ вызывает очередной символ ранее введенной команды.
- Команда называется **внешней**, если для ее выполнения должен быть загружен одноименный исполняемый файл. При вводе внешней команды производится поиск файла с таким же именем и расширением **COM, EXE, BAT** (поиск производится в указанном порядке в текущем каталоге и установленных каталогах поиска). Например, если пользователь ввел **TRK**, то ДОС будет искать файлы **TRK.COM**, или **TRK.EXE**, или **TRK.BAT** .

Принцип работы ДОС при неполном вводе команд (**принцип по "умолчанию"**):

- нет имени диска - считается текущий диск;
- нет имени каталога - считается текущий каталог;
- нет расширения при загрузке файла - считается **com** или **exe** или **bat**.

Команды ДОС.

1. изменить текущий диск:

Формат ввода: >имя диска:↵

Не следует вызывать диски A: и B: , если:

- в соответствующем дисководе нет дискеты
- дискета не читается
- дискета не отформатирована.

2. вызов содержимого текущего диска, каталога:

Формат ввода: >dir↵

Пример экрана после ввода команды DIR (в английском и русском варианте):

Volume in drive H is VOL1 Том в устройстве H имеет метку VOL1
Volume Serial Number is 0108-FC67 Серийный номер тома 0108-FC67
Directory of H:\CLIPBOAR\TMP Содержимое каталога H:\CLIPBOAR\TMP

.	<КАТАЛОГ>		
..	<КАТАЛОГ>		
CONFIG	<КАТАЛОГ>	21.10.98	23:26
W311	<КАТАЛОГ>	21.10.98	23:26
AUTOEXEC BAT	841	02.12.97	13:13
CONFIG SYS	1 022	02.12.97	13:14

4 файл(ов) 2279 байт 516 390 912 байт свободно

Обычно эту команду применяют после ввода других, чтобы увидеть изменения.

Если список достаточно длинный, то команда вводится следующим образом: >dir /P↵ (ставится пауза).

3. **открыть каталог:** >CD имя каталога↵ или перейти в корневой каталог, т.е.сделать его текущим. >CD \ ↵

4. **закрыть каталог**: >CD..↵

5. **загрузка исполняемого файла**: >имя файла.exe↵

6. **создание каталога**: >MD имя каталога↵ - имя не должно совпадать ни с одним имеющийся каталог.

7. **создание текстового файла:** >copy con имя файла↵

а) курсор перейдет на новую строку под приглашение

б) набрать текст (переход на другую строку только Enter, вернуться нельзя, нельзя перемещать курсор по тексту)

в) **F6** (вставка символа, обозначающего конец файла, это условное обозначение нечитаемый символ ^Z) появится в конце строки

г) **Enter**. Если все будет выполнено правильно, то в результате будет выдана запись «1 file(s) copied» или «1 файл скопировано». Чтобы убедиться, что файл создан, необходимо применить команду dir.

8. **просмотр файла**: >type имя файла↵ - содержимое файла будет выдано на экран.

9. **копирование файлов**: >copy имя файла-оригинала имя файла-копии↵ - копия будет иметь тоже самое содержимое, дату и время создания.

Особенности.

Пример 1. >copy myfile.txt double.doc↵ - в текущем каталоге будет получена копия файла myfile.txt с названием double/doc.

Пример 2. >copy myfile.txt a:\↵ - в корневом каталоге диска a: будет получена копия файла myfile.txt с тем же названием.

Пример 3. >copy myfile.txt TEXT↵ - в этом примере предполагается, что в текущем каталоге имеется подкаталог TEXT. В этом подкаталоге будет получена копия файла myfile.txt с тем же названием, если такого каталога нет, смотри пример 1, т.е. будет получена копия файла myfile.txt с новым названием TEXT.

Суть примеров: мы можем при копировании указать новое имя копии или диск, на который надо получить копию с тем же названием или подкаталог, в котором необходимо получить копию с тем же именем. Возможны комбинации данных примеров.

Пример 4. >copy myfile.txt d:\users\trk\double.doc↵ - указывается диск, путь каталогов и новое имя копии.

10. **переименование файла:** >ren имя файла новое имя файла↵ - пример: >ren myfile.txt new.txt↵. При вводе этой команды надо учитывать, что новое имя не должно совпадать с другими именами файлов и каталогов, которые находятся в том же каталоге, что и переименованный файл.

11. **удаление файла**: >del имя файла↵ - при удалении нескольких файлов выдается запрос на подтверждение операции. После ввода подобных команд выдается запрос на подтверждение, т.к. в результате действия этой команды может быть потеряна нужная информация. Поэтому нужно быть уверенным, что нужно удалить именно все файлы.

12. **удаление каталога**: >rd имя каталога↵ - нельзя удалить "не пустой" (т.е. содержащий файлы и/или подкаталоги) каталог. Если каталог не удаляется, а команда DIR не показывает наличие файлов, следует применить команду DIR /A:H для обнаружения и удаления скрытых файлов. Затем снова повторить команду удаления каталога. Также нельзя удалить текущий каталог. При удалении каталога, предварительно его необходимо закрыть.

Использование шаблонов в именах файлов.

Некоторые из выше рассмотренных операций не удобно применять на большие группы файлов. Например, скопировать 78 файлов на дискету или удалить все файлы из подкаталога.

Для задания шаблона применяют два специальных символа: * и ?.

1. ***** - заменяет любой набор символов.

Примеры:

- *.txt – все файлы с расширением txt
- trk.* - все файлы с именем trk
- *.* - все файлы

2. **?** – заменяет любой символ.

Примеры: doclad?.doc – файлы, у которых первые 6 символов doclad, седьмая произвольная, а расширение doc.

Специальные символы можно комбинировать: *.? – все файлы с расширением из одного символа.

Шаблоны можно применять в командах просмотра, копирования, удаления и распечатки.

Пример 1: >type *.txt↵ - просмотреть все файлы с расширением txt.

Пример 2: >copy *.* a:\↵ -скопировать все файлы текущего каталога в корневой каталог диска a:.

Пример 3: >del *.bak↵ - удалить все файлы с расширением bak.

Пример 4: >copy *.doc prn↵ - распечатать все файлы с расширением doc.

Решить задачи.

а) В некотором каталоге хранился файл **Задача5**. После того, как в этом каталоге создали подкаталог и переместили в созданный подкаталог файл **Задача5**, полное имя файла стало **E:\Класс9\Физика\Задачник\Задача5**. Каково было полное имя этого файла до перемещения? 1) E:\Физика\Задачник\Задача5; 2) E:\Физика\Задача5; 3) E:\Класс9\Задачник\Задача5; 4) E:\Класс9\Физика\Задача5.

б) В некотором каталоге хранился файл **Дневник.txt**. После того, как в этом каталоге создали подкаталог и переместили в созданный подкаталог файл **Дневник.txt**, полное имя файла стало **A:\SCHOOL\USER\TXT\MAY\Дневник.txt**. Каково полное имя каталога, в котором хранился файл до перемещения? 1) MAY; 2) A:\SCHOOL\USER\TXT; 3) TXT; 4) A:\SCHOOL\USER\TXT\MAY.

в) В некотором каталоге хранился файл photo.bmp. После того, как пользователь поднялся в этом каталоге на два уровня выше, создал там подкаталог Фотографии и скопировал туда файл photo.bmp, полное имя файла стало D:\Мои рисунки\Фотографии\photo.bmp. В каком каталоге первоначально находился этот файл? 1) D:\Мои рисунки\Избранное; 2) D:\Мои рисунки\Избранное\Графика; 3) D:\Фотографии \Избранное; 4) D:\Мои рисунки.

г) В некотором каталоге храниться файл **Задачи_по_программированию.txt.** В этом каталоге создали подкаталог и переместили в него файл **Задачи_по_программированию.txt.** После чего полное имя файла стало **D:\INFORM\LESSONS\10_CLASS**

Задачи_по_программированию.txt какого полное имя каталога, в котором хранился файл до перемещения? 1) **D:\INFORM** 2) **10_CLASS** 3) **D:\INFORM\LESSONS** 4) **LESSONS\10_CLASS**

д) Учитель работал в каталоге **D:\ Материалы к урокам \ 10 класс \ Практические работы.** Затем перешёл в дерево каталогов на уровень выше, спустился в подкаталог **Лекции** и удалил из него файл **Введение**. Каково полное имя файла, которое удалил преподаватель? 1) D:\ Материалы к урокам \ 10 класс \ Введение 2) D:\ Материалы к урокам \ 10 класс \ Лекции \ Введение 3) D:\ Материалы к урокам \ Лекции \ Введение 4) D:\ Материалы к урокам \ Лекции \ Введение.

4. Определите, какое из указанных имен файлов удовлетворяет маске:

а) d?*r.c?* 1) dark.cpp 2) dirty.c 3) dress.cpp 4) order.cpp

б) *lo?.c?* 1) ol.c; 2) sol.cpp; 3) solution.c; 4) solution.cpp.

ОС Windows.

Впервые появилась в 1985 году. Популярность завоевала в третьей версии (версия 3.11) в 1993 г. Начиная с четвертой версии, стала называться по году создания: 1995г. – Windows 95, 1998г. - Windows 98, 2000 г. Windows Millennium.

5 версия 2000г. – Windows 2000, 2001 г. - Windows XP,

6 версия 2006 г. – Windows Vista,

7 версия 2009 г. Windows 7.

8 версия 2012 г. Windows 8

9 версия - не существует

10 версия 2014 г. Windows 10

Достоинства Windows:

1. удобный интерфейс (внешний вид взаимодействия, интуитивно понятный пользователю)
2. автоматическая настройка большинства устройств компьютера и наличие драйверов для остальных .
3. использование всех ресурсов компьютера
4. легкость в обучении
5. поддержка мультимедиа
6. распространенность (90% компьютерного парка)

Недостатки:

1. значительные требования к компьютеру (каждая следующая версия Windows предъявляет все большие требования к процессору, оперативной памяти и жесткому диску).
2. некоторые программы под управлением Windows работают медленно или не работают совсем (до появления Windows было написано много хороших программ)

3. невысокая степень надежности, как самой ОС, так и защиты от хакерских атак.
4. Требуется оплатить лицензию как на саму Windows, так и на многие программы, работающие под ее управлением (например, MS Office)

OS Linux

Самая распространенная бесплатная ОС с открытым исходным кодом. При ее разработке из мира семейства UNIX старались взять все лучшее.

На сегодняшний день существует основное ядро Linux, разработку, которого координирует его создатель Линус

Достоинства:

1. Безопасность

Чтобы не быть многословным, достаточно сказать что, большинство из тех, кто устанавливают на своих компьютерах Linux, забывают о существовании антивирусных программ и о существовании самих компьютерных вирусов. Это не значит, что не существует вирусов, нацеленных на Linux. Они существуют. Но структура системы (разграничение прав и пр.) позволяет забыть об их существовании, если пользователь соблюдает простые правила элементарной "гигиены".

2. Доступность

Под доступностью здесь понимается, что Linux - бесплатная операционная система и бесплатно подавляющее большинство программ. При наличии соединения с интернетом установить Linux и любые программы вы можете прямо сейчас.

Недостатки:

1. Требовательность к подготовке пользователя

Предоставляя пользователю большие возможности в плане управления системой, Linux и требует от пользователя чуть больше. Там, где пользователю Windows достаточно кликнуть некоторое количество раз мышкой, пользователю Linux возможно придётся набрать с клавиатуры специальную команду со многими параметрами (аналогично, как в DOS). Подобная требовательность облегчается обилием информации в Internet, рассчитанной на пользователей разного уровня подготовленности.

2. Несовместимость с некоторыми устройствами

. Обычно, первыми появляются драйверы для наиболее популярной на данный момент операционной системы Windows, а драйверы для Linux зачастую "запаздывают". Поэтому, если в состав вашего компьютера входят устройства, первые образцы которых появились на рынке в последние пару месяцев, то Linux может их ещё не поддерживать.

3. Некоторое специализированное программное обеспечение создается только для Windows, поэтому в Linux его использовать затруднительно.

Прикладные программы

Текстовый редактор.

Текстовый редактор – прикладная программа, позволяющая создавать текстовые документы на магнитном диске, редактировать их, просматривать содержимое документа на экране, распечатывать документ, изменять формат документа и т.д.

Примеры текстовых редакторов: Блокнот, MS Word (Windows), Writer (Linux) и др.

Текстовый файл – простейшая форма хранения текстовой информации. С помощью компьютера можно создавать текстовые документы, в которых используются разнообразные шрифты, нестандартные символы, рисуются таблицы, схемы, включая графические изображения и т.д. Текстовые редакторы с такими возможностями часто называют текстовыми процессорами.

Символ – это наименьшая единица символьной информации.

Слово – это набор символов, ограниченный пробелами или знаками препинания.

Абзац – это группа смежных строк файла, первая из которых может начинаться отступом влево по отношению к другим строкам абзаца, нулевым отступом или отступом вправо (красная строка). Абзац всегда начинается с новой строки. Для каждого абзаца устанавливается левая и правая границы и отступ в первой строке. Для перехода к новому абзацу требуется нажать клавишу <Enter>.

Строка – это произвольная последовательность символов между левой и правой границами абзаца.

Редактирование – внесение изменений в набранный текст.

Набираемый пользователем на клавиатуре текст отражается в рабочем поле редактора экрана. Строка состояния и панель инструментов содержат информацию о текущем состоянии текстового редактора. Как правило, в них указываются координаты курсора (номер текущей строки и позиции в строке, столбец), номер страницы, формат текста, текущей шрифт и так далее.

Меню команд управления редактора включает команды изменения режимов работы, файловых операций, печати, форматирования текста, обращения за справкой и др. Если в пункте меню существует ярлык (пиктограмма), то значит, что в панели инструментов он продублирован.

Форматирование теста.

Под форматированием печатного текста подразумевается оформление по каким-либо правилам различных участков текста. При форматировании

меняется не сам текст, а его внешний вид (его границы, абзацные отступы, межстрочные интервалы, выравнивание текста, размеры страницы и т.п.). Установка параметров формата может производиться как до набора текста, так и после. Если установка производилась до набора, то в дальнейшем при вводе текста все заданные параметры автоматически выдерживаются текстовым редактором. Если текст уже был набран, и возникла необходимость изменить параметры форматирования, то перед этим текст нужно выделить.

Электронные таблицы.
Структура электронной таблицы. Формулы.

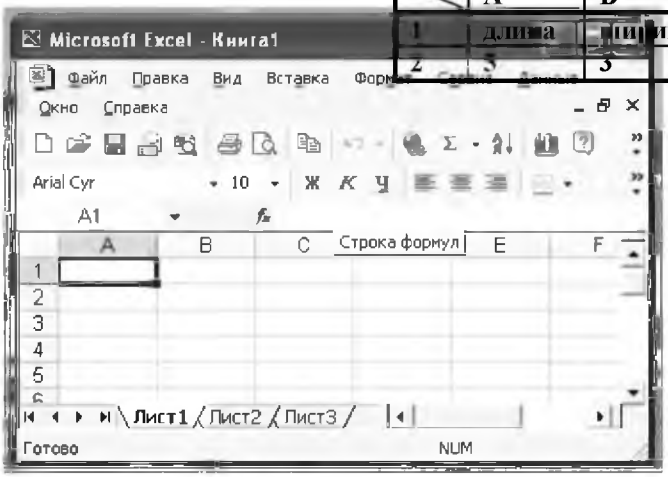
Электронная таблица (ЭТ)- инструмент для табличных расчетов на ЭВМ, т.е для того, чтобы обеспечить автоматическое вычисление табличных данных. Прикладные программы, позволяющие пользователю работать с электронными таблицами, называются табличными процессорами (ТП). Пример: Excel (Windows), Calc (Linux)

Электронная таблица состоит из прямоугольных клеток- *ячеек.* Горизонтальные ряды клеток образуют *строки,* вертикальные ряды- *столбцы.* Подобно шахматной доске строки имеют числовую нумерацию, а столбцы имеют буквенные обозначения (имена). В некоторых ТП допускаются числовые обозначения и для столбцов.

Для именования столбцов используются буквы латинского алфавита: A, B, C…. После столбца с именем Z следуют столбцы: AA, AB, AC, …, AZ, BA, BB и т. д. в алфавитном порядке. Каждая ячейка ЭТ имеет имя, составленное из имени столбца и номера строки, к которым она принадлежит. Примеры имен ячеек: A1, D5, M237, CA12. Имя ячейки определяет ее *адрес в таблице*, поскольку связано с местом расположения. В каждой ячейке может помещаться текст или формула.

Текст- это последовательность любых символов из компьютерного алфавита. Тексты используются для надписей, заголовков, пояснений, оформления таблицы и т. п.

	A	B	C
1	длина	ширина	площадь
2	3	3	=A2*B2



Формула- это выражение, определяющее

вычислительные действия ТП. Чаще всего формулы определяют математические вычисления. Вот пример заполненной электронной таблицы:

Таблица на рисунке выше находится в режиме отражения формул. Та же самая таблица в режиме отражения значений примет вид:

	A	**B**	**C**
1	**длина**	**ширина**	**площадь**
2	**5**	**3**	**15**

Все данные хранятся в ячейках. Их тип можно явно указывать. Для этого навести на ячейку мышку, ПКМ – формат – ячейки или пункт меню «Формат» - «Ячейки». Здесь можно выбрать тип данных: денежный, общий, числовой (с количеством знаков после запятой), процентный и т.д. Например: если ввели 5%, то при расчете будет умножаться на 0,05. в качестве разделителя между целой и дробной частью числа можно использовать запятую или точку (по умолчанию запятая).

Работа в ячейках аналогична работе с текстом в текстовом редакторе: форматирование, шрифты и т.д. но изменить размеры ячейки можно только на ее границах, которые обозначены на линейке, в самой таблице размеры изменить нельзя.

Блоком (фрагментом, диапазоном) таблицы называется любая прямоугольная часть таблицы. Блок обозначается именами диагонально-противоположных ячеек, B2:D3, вначале левый верхний угол, потом правый нижний. Блок может состоять только из одного столбца: например A1:A5, или из одной строки (B2:B10), или из одной ячейки (C3:C3).

Редактирование ячеек:

- замена содержимого – навести курсор на ячейку, набрать нужную информацию, Enter, при этом старые данные исчезают;
- внести изменения – а) навести курсор на ячейку, F_2, внести изменения, Enter; б) двойным щелчком ЛКМ по необходимой ячейке; в) щелкнуть ЛКМ в верхней строке формул (рис. см. выше)

Основное свойство электронной таблицы: *изменение числового значения в ячейке приводит к мгновенному пересчету формул, содержащих имя этой ячейки.*

Правила записи формул для различных ТП во многом схожи. Сформулируем эти правила:

- формулы содержат числа, имена ячеек, знаки операций, круглые скобки, имена функций;
- арифметические операции и их знаки: сложение (+), вычитание (-), умножение (*), деление (/), возведение в степень (∧);
- формула пишется в строку, символы последовательно выстраиваются друг за другом, проставляются все знаки операций; используются круглые скобки.

Автоматизацию вычисления обеспечивают формулы. В формулах разрешается использовать адреса других ячеек, при изменении данных той ячейки, от которой зависят другие, ЭТ автоматически пересчитывает значение. Любая формула начинается со знака «=», в конце Enter, в результате будет выдан ответ. При этом, если мы поставим курсор в данную ячейку, то в ней высветится формула, она же появится и в строке формул. Используются также различные функции: математические, статистические, финансовые и др.

	A	B	C
1	x	y	
2	5	=A2*A2	
3			

Иногда адреса ячеек, участвующих в формуле удобно не вводить вручную, а указывать мышкой те ячейки, которые участвуют в формуле.

Часто таблицы содержат похожие по смыслу расчетные формулы. Чтобы их постоянно не набирать, в ЭТ заложена возможность автоизменения формулы при копировании. Для этого необходимо знать 2 класса адресов в формулах.

Принцип относительной адресации. Относительный адрес обозначает следующее: *адреса ячеек, используемые в формулах, определены не абсолютно, а относительно места расположения формулы.* Он указывается обычным образом.

Копировать можно всеми основными способами текстового редактора, но это не всегда удобно. Например: если копии необходимо расположить в большом количестве подряд идущих ячейках. Поэтому существует свой способ: необходимо ухватиться ЛКМ за нижний правый угол ячейки или блока, который нужно скопировать, при этом мышка изменит свой внешний вид и станет ✚, а внизу у ячейки будет - ■, и перетащить копии в нужные ячейки.

Excel учитывает закономерность числовых данных в соседних ячейках при относительном копировании, при этом перед копированием необходимо выделить в блок те ячейки, которые показывают данную закономерность.

Например:

	-1						воскресенье		
	0		0,1	0,5	0,9	1,3	Понедельник		
	1						вторник		
1	2	3	4		Понедельник	вторник	среда	Четверг	пятница
	3						Четверг		
	4						пятница		

При копировании формул происходит изменение адресов ячеек. Если копирование в строке, то меняется буква столбца в зависимости алфавита, если в столбце, то в зависимости нумерации строк.

	A	B	C	D	E
1		2			
2		5			
3	9	4	=B1+B2		
4	11	2	=B2+B3		
5		=A3+A4	=B3+B4	=C3+C4	=D3+D4
6			=B4+B5		
7			=B5+B6		

Перемещение формул происходит при разнообразных манипуляциях фрагментами таблицы (копировании, вставках, удалении, переносе).

Абсолютная адресация. В некоторых случаях оказывается необходимым отменить действие принципа относительной адресации для того, чтобы при переносе формулы адрес ячейки не изменялся (т. е. был бы не относительным, а абсолютным). В таком случае применяется прием, который называется замораживанием адреса. Для этой цели в имени ячейки употребляется символ $. Для замораживания всего адреса, значок $ ставится дважды. Например: $B$2. Можно заморозить только столбец ($B2) или только строку (B$2). Тогда часть адреса будет изменяться при переносе формулы, а часть - нет.

	2			
	5			
9	4	=$B$3+B2		
11	2	=$B$3+B3		
	=$B$3+A4	=$B$3+B4	=$B$3+C4	=$B$3+D4
		=$B$3+B5		
		=$B$3+B6		

Стандартные функции.

При записи формул в ЭТ можно использовать **стандартные (встроенные) функции** («Вставка» -«Функция»). Все множество встроенных функций табличного процессора делится на несколько групп: математические, статистические, функции даты и времени и т. д.

К **математическим функциям** относятся такие известные из курса школьной математики функции, как SIN()- синус, COS()-косинус, TAN()-тангенс, LN()-натуральный логарифм, КОРЕНЬ()(SQRT)- квадратный корень числа и т. д. В круглых скобках (сразу за именем функции) записывается ее аргумент. При использовании тригонометрических функций следует учитывать, что аргумент должен быть задан в радианной

мере. В качестве аргумента функции может выступать числовая константа, адрес клетки табличного процессора или диапазон (блок) клеток.

Наибольший интерес представляют функции, аргументом которых является не одна ячейка, а диапазон ячеек. Наиболее часто используемой в табличных вычислениях математической функцией является **функция суммирования** аргументов SUM(). Аргументами этой функции являются либо диапазон клеток, перечисленных через запятую (в некоторых табличных процессорах в качестве разделителя аргументов используется «;»), адреса клеток, числовые константы: SUM(A3:B7;C5;C8:E11). Для автоматического суммирования выделим ячейку, в которой нужно вывести сумму, нажимается кнопка Σ. При этом ЭТ выделит предполагаемые ячейки для суммирования. При необходимости, прижимая ЛКМ, можно выделить другой блок ячеек. Бывает, что нужно кусками складывать, для этого прижимают Ctrl, а дальше продолжают выделение обычным способом, в конце нажать Enter. Сумму можно писать и вручную (см.выше).

Условная функция и логические выражения.

Условная функция. Общий вид условной функции следующий:

ЕСЛИ (<условие>, <выражение 1>, <выражение 2>)

Условие – это логическое выражение, которое может принимать значение **ИСТИНА** или **ЛОЖЬ.** <выражение 1> и <выражение 2> могут быть числами, формулами или текстами.

Условная функция, записанная в ячейку таблицы, выполняется так: если условие истинно, то значение данной ячейки определит <выражение 1>, в противном случае - <выражение 2>.

Логические выражения. Логические выражения строятся с помощью операций отношения (<, >, <=(меньше или равно), >=(больше или равно), =, <>(не равно)) и логических операций (логическое **И**, логическое **ИЛИ,** логическое **НЕ**). Результатом вычисления логического выражения являются логические величины **ИСТИНА** или **ЛОЖЬ.**

Существуют особенности записи логических операций в табличных процессорах: сначала записывается имя логической операции (**И, ИЛИ, НЕ**), а затем в круглых скобках перечисляются логические операции.

Пример. Разработать таблицу, содержащую следующие сведения об абитуриентах: фамилия, оценки за экзамены по математике, русскому и иностранному язык, сумму баллов за три экзамена и информацию о зачислении: если сума баллов больше или равна проходному баллу и оценка за математику – 4 или 5, то абитуриент зачислен в учебное заведение, в противном случае – нет.

Базы данных.
Реляционные (табличные) структуры данных.

База данных (БД) – это информационная модель, позволяющая хранить данные о группе объектов, обладающие одинаковым набором

	A	B	C	D	E	F
1	проходной	балл	13			
2	фамилия	математика	русский	иностранный	сумма	зачислен
3	Антонов	4	5	5	=СУММ(B3:D3)	=ЕСЛИ(И(E3>=$C$1;B3>3);"да";"нет")
4	Воробьев	3	5	5	=СУММ(B4:D4)	=ЕСЛИ(И(E4>=$C$1;B4>3);"да";"нет")

свойств. Например: телефонный справочник, энциклопедии, классный журнал, каталог файлов и т.д.

Мы будем изучать табличные БД.

Реляционные БД – базы данных с табличной формой организации информации. Реляционная БД состоит из одной или нескольких взаимосвязанных таблиц. Данные базы данных представляют собой совокупность строк и столбцов.

Запись – строка таблицы. Одна запись содержит информацию об отдельном объекте, описываемом в БД.

Поле – столбец таблицы. Поле содержит определенное свойство (атрибут) объекта. Каждое поле имеет имя.

Первичный (главный) ключ – это поле или группа полей, с помощью которых можно однозначно идентифицировать запись. Значение первичного ключа не должно повторяться у разных записей. С каждым полем связано еще одно очень важное свойство – тип поля. Тип определяет множество значений, которые может принимать данное поле в различных записях. В реляционных базах данных используется четыре основных типа полей: числовой, символьный, дата, логический.

Числовой тип имеют поля, значение которых могут быть только числами. Числа могут быть целыми и вещественными.

Символьный тип имеют поля, в которых будут храниться символьные последовательности (слова, тексты, коды и т. п.).

Тип «дата» имеют поля, содержащие календарные даты в различной форме.

Логический тип соответствует полю, которое может принимать всего два значения: «да»-«нет» или «истина»-«ложь».

Значения полей – это некоторые величины определенных типов. От типа величины зависят те действия, которые можно с ней производить. Например, с числовыми величинами можно выполнять арифметические операции, а с символьными и логическими – нельзя.

Для полей символьного и числового типов требуется также определить их ширину. При определении ширины поля нужно ориентироваться на максимально длинное значение, которое может храниться в этом поле. В некоторых случаях для числовых полей нужно задавать не ширину, а числовой формат (целое, длинное целое, с плавающей точкой и т. п.). Поля типа «дата» и логического типа имеют стандартную ширину.

База данных состоит из следующих объектов:

- Таблица – основное хранилище данных;
- Запросы – обеспечивают поиск и отбор нужной информации (обычно в таблице)
- Формы – графическое представление данных из таблиц и запросов;
- Отчеты – представление данных, готовых к распечатке.

.С каждым из элементов можно работать в двух режимах: а) режим таблицы – реальное отображение элемента. **Режим таблицы** позволяет просматривать и редактировать данные, организованные в виде строк и столбцов. Каждый столбец таблицы соответствует одному полю, а каждая строка - одной записи. В режиме таблицы на экран выводится сразу несколько записей БД. Его удобнее использовать для просмотра и удаления записей, а также для ввода информации в таблицы, состоящие из небольшого количества полей (так как каждое поле занимает в ширину определенную часть экрана).

б) режим конструктора – внутреннее устройство элемента.

Режим формы удобно использовать для ввода данных в таблицы с большим числом полей. В этом режиме в большинстве СУБД поля таблицы можно расположить в любом месте экрана, для чего существует специальный Конструктор форм. В более простых СУБД существует одна стандартная форма, в которой каждое поле занимает одну или несколько строк (в зависимости от ширины поля). В этом режиме на экран обычно выводится только одна запись таблицы базы данных.

Запрос – это средство извлечения информации из базы данных, отвечающей некоторым условиям, задаваемых пользователем. Результат

запроса (назовем его справкой) обычно выводится в виде таблицы, все записи которой удовлетворяют заданным условиям.

Организация межкомпьютерной связи.

Назовем задачи, которые трудно или невозможно решить без организации информационной связи между различными компьютерами:

- Перенос информации на большие расстояния.
- Совместное использование несколькими компьютерами дорогостоящих аппаратных, программных или информационных ресурсов – мощного процессора, емкого накопителя, высокопроизводительного лазерного принтера, баз данных, программного обеспечения и т.д.
- Совместная работа над большим проектом, когда исполнители должны всегда иметь последние копии общих данных во избежание путаницы и т.д.

Есть три основных способа организации межкомпьютерной связи:

1. объединение двух рядом расположенных компьютеров через их коммуникационные порты посредством специального кабеля.
2. передача данных от одного компьютера к другому посредством модема с помощью проводных или спутниковых линий связи.
3. объединение компьютеров в компьютерную сеть.

Часто при организации связи между двумя компьютерами за одним компьютером закрепляется роль поставщика ресурсов (программ, данных и др.), а за другим – роль пользователя ресурсов. В этом случае первый компьютер называется сервером, а второй клиентом или рабочей станцией. Работать можно только на компьютере-клиенте под управлением специального программного обеспечения.

Сервер (англ. Serve – обслуживать) – это высокопроизводительный компьютер с большим объемом внешней памяти, который обеспечивает обслуживание других компьютеров путем управления распределением ресурсов совместного пользования (данных, программ и периферийного оборудования). Клиент (рабочая станция) – любой компьютер, имеющий доступ к услугам сервера.

Например, сервером может быть мощный компьютер, на котором размещается центральная база данных, а клиентом – обычный компьютер, программы которого по мере необходимости запрашивают данные с сервера. В некоторых случаях компьютер может быть одновременно и клиентом, и сервером. Это значит, что он может предоставлять свои ресурсы и хранимые данные другим компьютерам и одновременно использовать их ресурсы и данные.

Клиентом также называют прикладную программу, которая от имени пользователя получает услуги сервера. Соответственно программное обеспечение, которое позволяет компьютеру предоставлять

услуги другому компьютеру, называют сервером – также как и сам компьютер.

Для преодоления несовместимости интерфейсов отдельных компьютеров вырабатывают специальные стандарты, называемые протоколами коммуникации.

Протокол коммуникации – это согласованный набор конкретных правил обмена информации между разными устройствами передачи данных. Имеются протоколы для скорости передачи, форматов данных, контроля ошибок и др.

Для работы с сетью необходимо наличие специального программного обеспечения, которое обеспечивает передачу данных в соответствии с заданным протоколом.

Протоколы коммуникации предписывают разбить весь объем передаваемых данных на пакеты – отдельные блоки фиксированного размера. Пакеты нумеруются, чтобы их затем можно было собрать в правильной последовательности. К данным, содержащимся в пакете, добавляется дополнительная информация примерно такого формата:

Адрес получателя	Адрес отправителя	Длина	Данные	Поле контрольной суммы

Контрольная сумма данных пакета содержит информацию, необходимую для контроля ошибок. Первый раз она вычисляется передающим компьютером. После того, как пакет будет передан, контрольная сумма повторно вычисляется принимающим компьютером. Если значения не совпадают, это означает, что данные пакета были повреждены при передаче. Такой пакет отбрасывается, и автоматически направляется запрос повторно передать пакет.

При установлении связи устройства обмениваются сигналами для согласования коммуникационных каналов и протоколов. Этот процесс называется подтверждением установления связи (англ. HandShake – рукопожатие).

Компьютерная сеть.

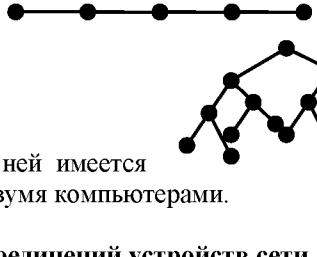
Компьютерная сеть (англ. Computer NetWork от net – сеть и work – работа) – совокупность компьютеров, соединенных с помощью каналов связи и средств коммутации в единую систему для обмена сообщениями и доступа пользователей к программным, техническим, информационным и организационным ресурсам сети. Иными словами, компьютерная сеть – это система компьютеров, связанных каналами передачи информации.

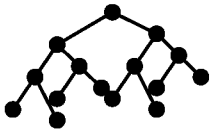
Компьютерную сеть представляют как совокупность узлов (компьютеров и сетевого оборудования) и соединяющих их ветвей (каналов связи).

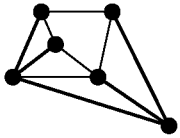
Компьютеры могут объединяться в сеть различными способами. Логический и физический способ соединения компьютеров, кабелей и других компонентов, в целом составляющих сеть, называется ее топологией. Иными словами, общая схема соединения компьютеров в локальные сети называется топологией сети. Топология характеризует свойства сетей, не зависящие от их размеров. При этом не учитываются производительность и принцип работы этих объектов, их типы, длины каналов, хотя при проектировании эти факторы очень важны.

Наиболее распространенные типы сетей:

1. Линейная –
2. Кольцевая –
3. Древовидная –
4. Звездообразная –
5. Ячеистая –
6. Полносвязная – в ней имеется ветвь между любыми двумя компьютерами.







Оборудование соединений устройств сети.

1. Сетевые кабели (коаксиальные, состоящие из двух изолированных между собой концентрических проводников, из которых внешний имеет вид трубки; оптоволоконные; кабели на витых парах, образованные двумя переплетенными друг с другом проводами, и др.)
2. Коннекторы (соединители) для подключения кабелей к компьютеру, разъемы для соединения отрезков кабеля.
3. Сетевые адаптеры для приема и передачи данных. Размещаются в системных блоках компьютеров, подключенных к сети. К разъемам адаптеров подключается сетевой кабель.
4. Трансиверы повышают уровень качества передачи данных по кабелю, отвечают за прием сигналов из сети и обнаружение конфликтов.
5. Хабы (концентраторы) и коммутирующие хабы (коммутаторы) расширяют возможности сетей.
6. Повторители (репитеры) усиливают сигналы, передаваемые по кабелю при его большой длине.

Классификация сетей.

Все многообразие компьютерных сетей можно классифицировать по группе признаков:

• Территориальная распространенность (по степени географического распространения);
• Ведомственная принадлежность;
• Скорость передачи информации;
• Тип среды передачи.

По территориальной распространенности сети могут быть локальными, глобальными, и региональными (городские), корпоративными.

По принадлежности различают ведомственные и государственные сети. Ведомственные принадлежат одной организации и располагаются на ее территории.

По скорости передачи информации компьютерные сети делятся на низко-, средне- и высокоскоростные.

По типу среды передачи разделяются на сети коаксиальные, на витой паре, оптоволоконные, с передачей информации по радиоканалам, в инфракрасном диапазоне.

Компьютерные сети по степени географического распространения.

Региональные компьютерные сети.

Локальные сети не позволяют обеспечить совместный доступ к информации пользователям, находящимся, например, в различных частях города. На помощь приходят региональные сети, объединяющие компьютеры в пределах одного региона (города, страны, континента). MAN – Metropolitan Area NetWork.

Корпоративные компьютерные сети.

Многие организации, заинтересованные в защите информации от несанкционированного доступа (например, военные, банковские и пр.), создают собственные, так называемые корпоративные сети. Корпоративная сеть может объединять тысячи и десятки тысяч компьютеров, размещенных в различных странах и городах (в качестве примера можно привести сеть корпорации Microsoft, MSN).

Локальные компьютерные сети.

Локальная сеть объединяет компьютеры, установленные в одном помещении (например, школьный компьютерный класс, состоящий из 8—12 компьютеров), в одном здании (например, в здании школы могут быть объединены в локальную сеть несколько десятков компьютеров, установленных в различных предметных кабинетах), на одном предприятии.

С точки зрения организации взаимодействия отдельных элементов локальной сети выделяют два типа таких систем:

1. Одноранговая сеть.
2. Сеть с выделенным сервером.

В небольших локальных сетях все компьютеры обычно равноправны, т. е. пользователи самостоятельно решают, какие ресурсы своего компьютера (диски, каталоги, файлы) сделать общедоступными по сети. Такие сети называются одноранговыми.

Если к локальной сети подключено более десяти компьютеров, то одноранговая сеть может оказаться недостаточно производительной. Для увеличения производительности, а также в целях обеспечения большей надежности при хранении информации в сети некоторые компьютеры специально выделяются для хранения файлов или программ-приложений. Такие компьютеры называются серверами, а локальная сеть — сетью на основе серверов.

Каждый компьютер, подключенный к локальной сети, должен иметь специальную плату (сетевой адаптер). Между собой компьютеры (сетевые адаптеры) соединяются с помощью кабелей.

Пользователи обращаются к серверу за нужными им файлами, переписывают их на свои рабочие станции и работают с ними. Если пользователю требуется сохранить созданный им документ, программу или другую информацию, то он может записать ее не только на свой диск, но и на диск сервера. Таким образом, сервер играет роль коллективного хранилища файлов. Для выполнения распечатки документа пользователь со своего компьютера может обратиться к принтеру, подключенному к серверу локальной сети, может использовать и другое оборудование сервера.

Итак, сервер – это компьютер, который обеспечивает передачу и (или) хранение информации в компьютерной сети.

Серверы соединены всевозможными каналами связи: проводными, кабельными, оптоволоконными, спутниковыми, радиоканалами и др. Эти каналы связи опутывают практически всю планету. Серверы никогда не выключаются и работают постоянно.

Некоторые серверы предназначены только для организации передачи информации в сети, другие – для хранения и выдачи информации пользователям. В сети Интернет такие серверы называют хостами. Есть серверы сети, которые выполняют обе функции одновременно.

Некоторые серверы

Основой программного обеспечения локальной сети является сетевая операционная система.

Топология локальных сетей.

Топологии сети могут быть различными. Сети Ethernet (эфир – широковещательная сеть, все станции сети могут принимать все сообщения) чаще всего могут иметь топологию «шина» и «звезда». В первом случае все компьютеры подключены к одному общему кабелю (шине), во втором - имеется специальное центральное устройство (хаб), от которого идут «лучи» к каждому компьютеру, т.е. каждый компьютер подключен к своему кабелю.

Структура типа «шина» проще и экономичнее, так как для нее не требуется дополнительное устройство и расходуется меньше кабеля. Но она очень чувствительна к неисправностям кабельной системы. Если кабель поврежден хотя бы в одном месте, то возникают проблемы для всей сети. Место неисправности трудно обнаружить.

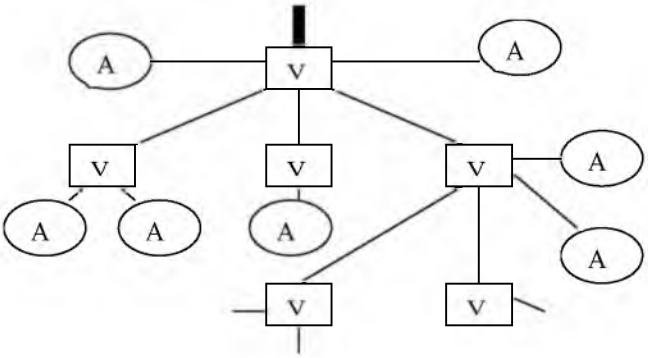
В этом смысле «звезда» более устойчива. Поврежденный кабель – проблема для одного конкретного компьютера, на работе сети в целом это не сказывается. Не требуется усилий по локализации неисправности.

В сети, имеющей структуру типа «кольцо» информация передается между станциями по кольцу с переприемом в каждом сетевом контроллере. Переприем производится через буферные накопители, выполненные на базе оперативных запоминающих устройств, поэтому при выходе их строя одного сетевого контроллера может нарушиться работа всего кольца. Достоинство кольцевой структуры – простота реализации устройств, а недостаток – низкая надежность.

Глобальная компьютерная сеть Интернет.

Глобальную компьютерную сеть называют телекоммуникационной сетью, а процесс обмена информацией по такой сети называют телекоммуникацией (от греч. «tele» - вдаль, далеко и лат.«comunicato» - связь).

Организация в глобальных сетях похожа на организацию телефонной связи. Телефон каждого абонента подключен к определенному узлу-коммутатору. Связь между коммутаторами организована следующим образом, чтобы любые два абонента, где бы они ни находились, могли бы поговорить друг с другом. И такая телефонная связь работает по всему миру. Аналогично работают компьютерные сети. Персональный компьютер пользователя сети (абонент) подключается к определенному узлу коммутации. Узлы связаны между собой, и эта связь действует постоянно.



Каждая региональная или отраслевая компьютерная сеть обычно имеет связь с другими сетями. Для этого один из узлов сети выполняет функцию шлюза. Он соединяется линией связи с аналогичными узлами других сетей. Потребность в шлюзе существует лишь в том случае, если в связываемых сетях используются разные протоколы.

Интернет (от англ. «net» - сеть, «Internet» - объединение сетей) - это глобальная компьютерная сеть, объединяющая многие локальные, региональные и корпоративные сети и включающая в себя десятки миллионов компьютеров.

В каждой локальной или корпоративной сети обычно имеется, по крайней мере, один компьютер, который имеет постоянное подключение к Интернету с помощью линии связи с высокой пропускной способностью (сервер Интернета). Надежность функционирования глобальной сети обеспечивается избыточностью линий связи: как правило, серверы имеют более двух линий связи, соединяющих их с Интернетом.

Основу, «каркас» Интернета составляют более ста миллионов серверов, постоянно подключенных к сети. К серверам Интернета могут подключаться с помощью локальных сетей или коммутируемых телефонных линий сотни миллионов пользователей сети.

Интернет финансируется правительствами, научными и образовательными учреждениями, коммерческими структурами и миллионами частных лиц во всех частях света, но никто конкретно не является ее владельцем. Управляет сетью Совет по архитектуре Интернета, формируемый из приглашенных добровольцев. Сеть была создана в 1984 г.

История Интернета.

1958 г. – в ответ на запуск Советским Союзом 7 октября 1957 г. первого искусственного спутника Земли при Министерстве обороны США было создано Агентство перспективных исследовательских проектов (Advanced Research Projects Agency, ARPA). К работе по заданиям ARPA привлекались лучшие ученые университетов США. Именно перед ARPA была поставлена задача создания компьютерной сети в интересах обороны.

В 1969 году в США была создана компьютерная сеть ARPAnet, объединяющая компьютерные 5 центров министерства обороны и ряда академических организаций. Эта сеть была предназначена для узкой цели: главным образом для изучения того, как поддерживать связь в случае ядерного нападения и для помощи исследователям в обмене информацией. По мере роста этой сети создавались и развивались многие другие сети. Еще до наступления эры персональных компьютеров создатели ARPAnet приступили к разработке программы Internetting Project ("Проект объединения сетей"). Успех этого проекта привел к следующим результатам. Во-первых, была создана крупнейшая в США сеть internet (со строчной буквы i). Во-вторых, были опробованы различные варианты

взаимодействия этой сети с рядом других сетей США. Это создало предпосылки для успешной интеграции многих сетей в единую мировую сеть. Такую "сеть сетей" теперь всюду называют Internet (в отечественных публикациях широко применяется и русскоязычное написание - Интернет). 29 октября впервые по сети из Лос-Анджелеса в Стэнфорд передано текстовое сообщение. После передачи двух букв единственного слова LOGIN работа сети нарушилась.

1972 г. - сеть ARPAnet объединяет 15 локальных сетей (23 хоста).

1983 г. – 1 января сеть ARPAnet была переведена с протокола обмена NCP на протокол TCP/IP, который действует с Интернете до сих пор. К ARPAnet начинают активно подключаться другие самостоятельно возникающие по всему миру компьютерные сети.

1986 г. – создана сеть национального научного фонда США NSFnet, которая поначалу связывала только 5 крупных компьютерных центров США.

1988 г. – роль основной сети, к которой подключаются остальные, переходит к NSFnet.

1990 г. – сеть ARPAnet прекращает существование.

1991 г. – Интернет становится доступным для индивидуальных пользователей.

1995 г. – сеть NSFnet вновь становится чисто научной сетью. Главенство какой-то одной сети в Интернете закончилось. Подключиться к Интернету можно через любую сеть, которая уже стала его частью.

Как можно связаться с Интернетом? Самым распространенный способ – посредством модема и телефонной линии. При этом используются три типа подключения, отличающиеся друг от друга по объему услуг и цене:

1. почтовое – позволяет только обмениваться электронной почтой с любым пользователем Интернета, самое дешевое;
2. сеансное в режиме on-line (на прямом проводе) – работа в диалоговом режиме – все возможности сети на время сеанса;
3. прямое (личное), самое дорогостоящее – все возможности в любое время.

Как связываются между собой сети в Интернете? Отдельные участки Интернета представляют собой сети различной архитектуры. Передаваемы данные разбиваются на небольшие порции, называемые пакетами. Каждый пакет перемещается по сети независимо от других пакетов.

Сети в Интернете неограниченно коммутируются (связываются) друг с другом, потому что все компьютеры, участвующие в передаче данных, используют единый протокол коммуникации TCP/IP (тисипи-айпи).

На самом деле протокол TCP/IP – это два разных протокола, определяющие различные аспекта передачи данных в сети:

- протокол TCP (Transmission Control Protocol) – протокол управления передачей данных, использующий автоматическую повторную передачу пакетов, содержащих ошибки; этот протокол отвечает за разбиение передаваемой информации на пакеты и правильное восстановление информации из пакетов получателя;
- протокол IP (Internet Protocol) – протокол межсетевого взаимодействия, отвечающего за адресацию и позволяющий пакету на пути к конечному пункту назначения проходить по нескольким сетям.

Схема передачи информации по протоколу TCP/IP такова: протокол TCP разбивает информацию на пакеты и нумерует их; далее с помощью протокола IP все пакеты передаются получателю, где с помощью протокола TCP проверяется, все ли пакеты получены; после получения всех пакетов протокол TCP располагает их в нужном порядке и собирает в единое целое.

Каким образом пакет находит своего получателя? Каждый компьютер, подключенный к сети Интернет, имеет два равноценных уникальных адреса: цифровой IP-адрес и символический доменный адрес. Присваивание адресов происходит по следующей схеме: Международная организация Сетевой Информационный центр выдает группы адресов владельцам локальных сетей, а последние распределяют конкретные адреса по своему усмотрению.

IP-адрес компьютера имеет длину 4 байта. Обычно первый и второй байты определяют адрес сети, третий – адрес подсети, а четвертый – адрес компьютера подсети. Для удобства IP-адрес записывают в виде четырех чисел со значением от 0 до 255, разделенных точками, например: 145.37.5.150. Адрес сети – 145.37; адрес подсети – 5; адрес компьютера в подсети – 150. Первый байт может быть только от 1 до 223, остальные зарезервированы за специальными адресами.

Доменный адрес (англ. Domain – область) в отличие от цифрового является символическим и легче запоминается. Пример доменного адреса: barsuk.les.nora.ru. Здесь домен barsuk – имя реального компьютера, обладающего IP-адресом, домен les – имя группы, присвоившей имя этому компьютеру, домен nora – имя более крупной группы, присвоившей имя домену les, и т.д. В процессе передачи данных доменный адрес преобразуется в IP-адрес.

Если Вы вводите DNS (доменный) - адрес, то он сначала направляется в так называемый сервер имен, который преобразует его в 32 - битный IP - адрес для машинного считывания.

DNS - адрес обычно имеет три составляющие (хотя их может быть сколько угодно).

Доменная система имен имеет иерархическую структуру: домены верхнего уровня - домены второго уровня и так далее. Домены верхнего уровня бывают двух типов: географические (двухбуквенные - каждой стране свой код) и административные (трехбуквенные). России принадлежит географический домен ru.

Правила записи доменных имен. Нумерация меток в доменных именах ведется справа налево. Первая метка доменного имени считается меткой первого (высшего) уровня, левее расположена метка второго уровня и т.д.

Метка первого уровня может состоять из двух и более букв. За использование и сопровождение доменных меток первого уровня отвечает неправительственная Интернет-коропорация по присвоению имен и номеров (internet Corpration for Assined Names and numbers? ICAAN). Метки первого уровня из двух букв, как правило, описывают принадлежность к стране.

Среди часто используемых доменов - идентификаторов стран можно выделить следующие:

- at – Австрия
- au - Австралия
- br - Бразилия
- ca - Канада
- ch - Швейцария
- de - Германия
- dk - Дания
- es - Испания
- fi - Финляндия
- fr - Франция
- in - Индия
- it - Италия
- jp - Япония
- mx - Мексика
- ni - Никарагуа
- nl - Нидерланды
- no - Норвегия
- nz - Новая Зеландия
- ru или su- Россия
- se - Швеция
- uk - Украина
- za - Южная Африка
- us - США

Метки первого уровня из трех и более букв являются общеорганизационными и чаще всего описывают принадлежность отрасль, к которой относится организация-владелец компьютера или локальной сети. Нарпимер:

- gov - правительственное учреждение или организация
- mil - военное учреждение
- com - коммерческая организация
- net - сетевая организация
- org - организация, которая не относится не к одной из выше перечисленных.
- Edu – образовательные учреждения

и т.д.

Портал www.klyaksa.net зарегистрировал домен второго уровня klyaksa в административном домене верхнего уровня net. Имена компьютеров, которые являются серверами Интернета, включают в себя полное доменное имя и собственно имя компьютера. Так полный адрес портала имеет вид www.klyaksa.net.

Основные возможности, предоставляемые сетью Интернет.

- *Telnet* – сетевая программа удаленного доступа позволяет с помощью протокола Telnet с одного компьютера присоединиться к другому и использовать не только его информационные ресурсы, но и запускать программы (как автономные, так и типа клиент-сервер).
- *FTP (File Transfer Protocol – протокол передачи файлов)* – средства доступа к удаленному компьютеру, позволяющие с помощью протокола FTP просматривать его каталоги и файлы, переходить из одного каталога в другой, копировать, удалять и обновлять файлы. Компьютеры, на которых ест файлы для общего пользования, называются FTP-серверами.
- *Gopher* (от слова «рыть») – более развитые средства поиска и извлечения архивной информации с помощью многоуровневых меню ряд дискуссий по конкретным темам, справочных книг, индексных ссылок и т.п.

Системы информационного поиска сети Интернет. В Интернете представлена информация на любые темы, которые можно себе представить. Но найти в ней нужную информацию не так-то легко из-за того, что сеть по своей природе не имеет четкой структуры. Поэтому для ориентировки в Интернете и быстрого получения свежей справочной информации разработаны системы поиска информации. Все системы поиска информации Интернета располагаются на специально выделенных компьютерах с мощными каналами связи. Поисковые системы можно разбить на два типа: предметные каталоги, формируемые специалистами-

редакторами; автоматические индексы, формируемые специальными компьютерными программами, без участия людей.

Системы, основанные на предметных каталогах, используют базы данных, формируемые специалистами-редакторами, которые отбирают информацию, устанавливают связи для баз данных, организуют и снабжают данные в разных поисковых категориях перекрестными ссылками. Компании, владеющие предметными каталогами, непрерывно исследуют, описывают и каталогизируют содержимое WWW-серверов и других сетевых ресурсов, разбросанных по всему миру. В результате этой работы клиенты Интернета имеют постоянно обновляющиеся иерархические (древовидные) каталоги, на верхнем уровне которых собраны самые общие категории, такие, как «бизнес», «наука», «искусство»м и т.д., а элементы самого нижнего уровня представляют собой ссылки на отдельные WWW-страницы и серверы вместе с кратким описанием их содержимого. Например: если надо найти информацию о динозаврах, достаточно спуститься по иерархии: науки – животные – доисторические животные – динозавры.

Каталоги, составленные людьми, более осмысленны, чем автоматические индексы. Их очень мало, т.к. их создание и поддержка требуют огромных затрат. Для примера рассмотрим самый популярный предметный каталог Yahoo!, который обладает одной из крупнейших баз данных, имеет информационные базы для детей и подростков, поддерживает два основных метода работы с каталогом – поиск по ключевым словам и поиск по иерархическому дереву разделов, не принимает запросов на естественном языке

Автоматические индексы. Поиск по ключевым словам в одной базе данных, занимающей в худшем случае несколько секунд, даст те же результаты, что и обшаривание всех WWW-страниц во всей сети Интернет. Автоматический индекс состоит из трех частей: программы-робота; базы данных, собираемой этим роботом; интерфейса для поиска в базе, с которым работает пользователь. Все компоненты функционируют без вмешательства человека. К автоматическим индексам следует прибегать только тогда, когда ключевые слова известны, например фамилия человека или несколько специфических терминов из соответствующей области. Индексы получают информацию из каждого отдельного узла, регистрируют и индексируют ее и добавляют к своим базам данных.

Среди известных индексов выделяется AltaVista, Rambler, Yandex, Yahoo, Au, Aport – одни из самых мощных, полностью автоматических поисковых систем, обладает полнотекстовой базой данных, выдает наибольшее количество ссылок.

- *Система телеконференций Usenet (Users Network)* организует коллективные обсуждения по различным направлениям, называемые телеконференциями. В каждой телеконференции проводится ряд

дискуссий по конкретным вопросам. Сегодня Usenet имеет более десяти тысяч дискуссионных групп (NewsGroups) или телеконференций, каждая из которых посвящена определенной теме и является средством обмена мнениями.

Любая телеконференция строится по определенным правилам, которые оговариваются в момент ее открытия. Конференции бывают открытые (т.е. доступные для всех желающих) и закрытые (для участия в которых допускаются только избранные пользователи).

Телеконференции располагаются в сети на серверах конференций, электронные адреса которых выглядят следующим образом:

News@имя_сервера

Newsserv@имя_сервера

Newsserver@имя_сервера

News-server@имя_сервера

У каждой телеконференции есть свой собственный адрес в сети, организованный так же, как и обычный почтовый адрес пользователя, например: news.kids.rus. В этом адресе отсутствует лишь левая часть – персональное имя почтового ящика пользователя. Каждое слово в адресе телеконференции служит для уточнения содержания вопросов, обсуждаемых в телеконференции. Самое первое слово означает принадлежность конференции к определенному разделу в общепринятой иерархии телеконференций, совпадающей чаще всего с названием сети, где они проводятся. Например:

fido7.ccmail

relcom.comp.os.windows

glasnet.news.eng.

Телеконференции разбиты на несколько групп:

- News. – вопросы, касающиеся системы телеконференций;
- Comp.– компьютеры и программное обеспечение;
- Rec. – развлечения, хобби и отдых;
- Sci. – научно-исследовательская деятельность и приложения;
- Soc. – социальные вопросы;
- Talk. – дебаты по различным спорным вопросам;
- Misc. – все остальное.
- Humanities. – гуманитарные науки и искусство;
- Alt. – альтернативные (отличные от общепринятых) дискуссии;
- Bionet – биологические исследования;
- Info. – информация различного характера;
- Biz. – деловая информация о товарах и услугах;
- K12. – школьные проекты;
- Schl. – конференции для школьников и студентов.

Внутри этих категорий существует иерархия. Так, например, rec.music.beatles – это дискуссия о творчестве «Битлз», входящая в подгруппу «музыка» группы дискуссий по искусству. Существует

большой выбор программ чтения телеконференций, которые формируют материал дискуссий в упорядоченном виде и предоставляют в распоряжение корреспондентов. Аналог телеконференций в других сетях – «электронная доска объявлений» (Bulletin Board system, BBS).

- *Электронная почта (Electronic mail, англ. mail – почта, сокращенно E-mail).* Служит для передачи самой разнообразной информации: письма, объявления, реклама, программное обеспечение, компьютерные игры, деловая информация, звуковая, графическая и двоичная информация. Обмен письмами в глобальных компьютерных сетях называется электронной почтой. Электронное письмо – это обычный файл, содержащий электронный адрес получателя и тест письма. Зарегистрированный абонент сети получает на почтовом сервере провайдера (распространителя), с которым он непосредственно соединен, так называемый почтовый ящик. Почтовый ящик – это раздел внешней памяти почтового сервера, отведенный для абонента. Каждому почтовому ящику присваивается свое, отличное от других, имя. В этот ящик почтовый сервер помещает письма, поступающие к абоненту.

Передать письмо в почтовый ящик может любой абонент сети, если он знает электронный адрес. Извлечь (прочитать или переписать на свою машину) письмо из почтового ящика может только его владелец. Доступ к информацию защищен паролем, который знает только один хозяин ящика.

Другое преимущество состоит в том, что электронное письмо может содержать не только текстовое сообщение, но и вложенные файлы (программы, графику, звук и пр.). Однако не рекомендуется пересылать по почте слишком большие файлы, так как это замедляет работу сети. Для того чтобы этого не происходило, на некоторых почтовых серверах вводятся ограничения на размер пересылаемых.

Кроме того, электронная почта позволяет:

• посылать сообщение сразу нескольким абонентам;

• пересылать письма на другие адреса;

• включить автоответчик, на все приходящие письма будет автоматически отсылаться ответ;

• создать правила для выполнения определенных действий с однотипными сообщениями (например, удалять рекламные сообщения, приходящие от определенных адресов) и так далее.

Для работы с электронной почтой необходимы специальные почтовые программы, причем для любой компьютерной платформы существует большое количество почтовых программ. Можно выделить пару программ - почтовый сервер и почтовый клиент. Почтовые серверы

работают на узловых компьютерах Интернета, а почтовые клиенты должны быть у каждого пользователя e-mail. Простейший почтовый клиент — программа Microsoft Outlook Express. Она входит в стандартный пакет Windows (начиная с Windows 98) и потому есть в наличии на каждом компьютере, работающем в этой операционной системе.

Природа сообщений электронной почты.

Принципы работы электронной почты существенно отличаются от принципов работы других программных систем. Сообщение электронной почты — это не файл. Именно поэтому не принято говорить «отправка письма», «получение документа» и т. п. Вместо этого обтекаемо говорят об отправке и получении сообщений.

Кроме файлов данные могут храниться в базах данных. Базу данных проще всего представлять в виде большой таблицы, в которой строки называются записями, а столбцы — полями. Вся таблица может храниться в файле, но каждая отдельная запись — это не файл, а самостоятельный логический блок данных, который так и называется записью. Сообщения электронной почты — это не файлы, а именно записи в какой-то (обычно невидимой для пользователя) базе данных.

Когда на компьютере пользователя устанавливается почтовый клиент, вместе с ним создается и почтовая база данных. Каждое полученное или отправленное письмо — это отдельная запись (строка) в этой почтовой базе. Саму почтовую базу на компьютере мы не видим. Вместо этого почтовый клиент позволяет нам просматривать отдельные записи в этой базе («читать письма») и создавать новые записи (то есть как бы «писать письма»).

На почтовом сервере тоже существует почтовая база данных, только она гораздо больше по размеру, потому что у сервера не один клиент, а множество. Сообщения, поступившие в наш адрес, заносятся в эту базу в виде записей, точно так же, как и сообщения, отправленные нами. Такие же почтовые базы существуют на всех почтовых серверах Интернета, а дальше между базами происходит операция, которая называется синхронизацией.

Допустим, мы подготовили несколько сообщений для отправки в другие города (готовить сообщения можно не подключаясь к Интернету). При этом в почтовой базе клиентской программы возникнут новые записи. В то же самое время на сервер могли поступить сообщения, адресованные нам. Поскольку у нас нет соединения с Интернетом, они запишутся в почтовую базу сервера в виде записей, где и будут храниться.

Как только мы подключимся к Интернету и свяжемся с почтовым сервером, начнется сличение клиентской и серверной почтовых баз. Если в них будут обнаружены различия (а они действительно будут обнаружены, раз мы подготовили сообщения для отправки), то начнется копирование записей в ту и другую сторону. Оно продолжится до тех пор, пока различия не будут ликвидированы — это и есть синхронизация почтовых

баз. В ходе синхронизации на сервер копируются записи с сообщениями, подготовленными для отправки, а на клиентский компьютер — записи с сообщениями, поступившими в его адрес. Таким образом, прием и отправка почтовых сообщений происходит в рамках одной операции синхронизации. Потом сервер точно так же синхронизирует свою базу с базой соседнего сервера, а тот — с базой своих соседей. В итоге сообщения движутся по Сети от отправителей к адресатам, хотя никакие файлы при этом не передаются и не принимаются. Если вы, все же, захотите сохранить какое-то сообщение в файл, то вам необходимо открыть его в почтовом клиенте и дать команду Файл→Сохранить как…

Учетная запись электронной почты.

Выше мы говорили о том, что каждый клиент может создать «почтовый ящик». Никаких личных ящиков, контейнеров и других «хранилищ» на самом деле нет. Вместо них на сервере ведется одна почтовая база на всех клиентов. Все сообщения, поступающие к клиентам, становятся записями этой единой базы, а дальше работают фильтры. Когда к серверу подключается Иванов, его клиентская программа видит сквозь фильтр Иванова только те записи, которые адресованы Иванову. Почтовый клиент Петрова подключается к серверу через фильтр Петрова и потому получает доступ к совершенно другим записям, и т. д.

В итоге для разных клиентов сервер должен хранить разные совокупности настроек (включая фильтры). Хранятся они тоже в базе данных и тоже в виде записей, а называются эти записи учетными записями. Таким образом, у каждого клиента есть своя учетная запись. Ее в просторечии и называют «почтовым ящиком», хотя эта аналогия и неправильна. В почтовых ящиках обычно хранят почту. В учетной записи хранятся сведения о клиенте и параметры для настройки его фильтра.

Протоколы электронной почты.

У каждой сетевой службы должен быть свой протокол. Он определяет порядок взаимодействия клиентской и серверной программ. От него зависит, что может запросить та или иная сторона, а что — не может; на что может ответить сторона, а на что — не должна. Он же определяет, в какой форме должен быть сделан запрос и как должен быть представлен ответ.

В электронной почте e-mail используют не один прикладной протокол, как в других службах Интернета, а два. По одному протоколу происходит отправка почты, а по другому — ее прием. Необходимость в двух протоколах связана с требованиями безопасности. Так, например, при отправке сообщений можно не проверять личность отправителя — это аналогично тому, что письмо брошено в уличный почтовый ящик. Другое дело — получение сообщений. Здесь надо предъявить свои права и пройти идентификацию. Так, например, при получении заказных писем в

почтовом отделении всегда необходимо предъявить паспорт или заменяющий его документ. Кому попало чужую почту в руки не отдадут.

Для отправки на сервер и для пересылки между серверами используют протокол, который называется SMTP (Simple Mail Transfer Protocol — простейший протокол передачи сообщений). Он не требует идентификации личности.

Для получения поступившей почты используется протокол POP3 (Post Office Protocol 3 — протокол почтового отделения, версия 3). Он требует идентификации личности, то есть должно быть предъявлено регистрационное имя (Login) и пароль (Password), который подтверждает правомочность использования имени.

При использовании электронной почты каждому абоненту присваивается уникальный почтовый адрес. Электронный адрес используется для определения местонахождения почтового ящика адресата в сети. Электронный адрес включает в себя две части, отделенные между собой символом @: имя_пользователя@адрес(имя)_ почтового_сервера.
Пример: somov@pgu.perm.ru – он состоит из таких частей. Страна – Россия, город – Пермь, имя сервера – ПГУ, имя почтового ящика – Сомов.
Точки и символ @ - разделительные знаки. Разделенные точками части электронного адреса – домены, их количество может быть разным.
В адресах не всегда используется административно-географический принцип (страна-город-район и т.д.). Вся часть адреса, расположенная справа от значка @, является доменным именем почтового сервера, содержащего ящик абонента.

Куда: frolov@mgu.moskva.ru
Откуда: somov@pgu.perm.ru
О чем: приглашение

Приглашаю Вас принять участие в научной конференции Пермского университета! С уважением Е.Сомов

В целом электронное письмо состоит из конверта и текста. Обычно на конверте записывается адрес получателя, отправителя и краткая информация о назначении письма (два последних элемента необязательных).
Пример:
Иногда в Интернет-сетях вместо кода страны используется название самой сети, в которой работает пользователь (INFO@BITNIC.BITNET). В системе адресации, принятой в США, суффикс страны не указывается, а обозначает тип организации, которой принадлежит сервер.

- *WWW (World Wide Web – всемирная паутина)* – интерактивная гипертекстовая информационно-поисковая система в Интернете. Блоки данных WWW («страницы») размещаются на отдельных компьютерах, называемых WWW-серверами и принадлежащих отдельным организациям

или частным лицам. Видимо, имея в виду такую аналогию, создатели ввели понятие Web-страницы.

Web-страница – это электронный документ, в котором кроме текста содержаться отдельные команды форматирования, а также встроенные объекты (рисунки, аудио- и видеоклипы и др.), хранящийся на Web-сервере.

Важнейшим свойством WWW является гипертекстовая организация связей между Web-страницами. Причем эти связи действуют не только между страницами на одном сервере, но и между разными серверами WWW.

Просматривают Web-страницы с помощью специальных программ, называемых браузерами, так, что браузер — это не просто клиент WWW, служащий для взаимодействия с удаленными Web-серверами, это еще и средство просмотра Web-документов. Так, например, если Web-страница была сохранена на жестком диске, ее можно просмотреть с помощью браузера без подключения к Интернету. Такой просмотр называют автономным.

В отличие от печатных электронных документов, Web-страницы имеют не абсолютное, а относительное форматирование, то есть они форматируются в момент просмотра в соответствии с тем, на каком экране и с помощью какого браузера их просматривают. Строго говоря, одна и та же Web-страница при просмотре в разных браузерах может выглядеть по-разному — это зависит от того, как браузер реагирует на команды, которые встроил в Web-страницу ее автор.

Browse – обозревать, осматривать. Сегодня наиболее популярны Netscape Navigator, Microsoft Internet Explorer, . Web-браузер – прикладная программа для просмотра Web-страниц. Браузер выводит на экран Web-страницу по прямому указанию ее адреса, введенному пользователем или извлеченному из гиперссылки.

Для поиска информации в Интернете используется универсальная адресация в виде адресов – определителей местонахождения информационных ресурсов - URL (Uniformed Resource Locator) или, сокращенно, URL-адресом. URL-адрес содержит информацию не только о том, где находится ресурс, но и как к нему обращаться. URL-адрес состоит из трех частей и читается слева направо в отличие от доменных имен. Пример URL: http://klyaksa.net/htm/exam/answers/images/a23_1.gif. или http://имя сервера/путь/файл. Первая (левая) часть содержит имя прикладного протокола или, иными словами, тип связи, по которому осуществляется доступ к данному ресурсу. Для службы World Wide Web это протокол передачи гипертекста HTTP (Hyper Text Transfer Protocol). У других служб — другие протоколы, например: ftp:// - используется протокол ftp при обращении у ftp-серверам, gopher:// - подключение к серверам Gopher. Имя протокола отделяется от остальных частей адреса двоеточием и двумя косыми чертами.

Второй элемент – доменное имя компьютера, на котором хранится данный документ, т.е. где именно расположен данный ресурс (имя соответствующего сервера). Со структурой доменного имени мы уже знакомы – его элементы разделяются точками. После доменного имени ставится косая черта.

Последний элемент адреса - путь доступа к файлу, содержащему Web-документ, на указанном компьютере. С записью пути доступа к файлу в операционной системе Windows мы уже знакомы, но здесь есть важное отличие. В Windows принято разделять каталоги и папки символом обратной косой черты «\», а в Интернете положено использовать обычную косую черту «/». Это связано с тем, что Интернет зарождался на компьютерах, работающих в операционной системе UNIX, а там принято разделять каталоги именно так.

Задачи.

1. а) Идентификатор некоторого ресурса сети Интернет имеет следующий вид: http://www.ftp.ru/index.html. Какая часть этого идентификатора указывает на протокол, используемый для передачи ресурса? 1) www; 2) ftp; 3) http; 4) html.

б) Доступ к файлу http.txt, находящемуся на сервере www.net осуществляется по протоколу ftp. В таблице фрагменты адреса файла закодированы буквами от А до Ж. Запишите последовательность этих букв, кодирующую адрес указанного файла. А) :// Б) http В) ftp Г) .net Д) .txt Е) / Ж) www

в) При использовании протокола передачи файлов был осуществлен доступ к документу Document1, созданному при помощи программы MS Word. Укажите правильный адрес для доступа к данному файлу. 1) ftp://www.net/http/Document1.doc; 2) ftp://http/Document1.txt; 3) http://www.net/ftp/Document1.doc; 4) http://ftp/www.net/Document1.doc; 5) http://ftp/www.net/Document1.txt.

г) Доступ к файлу ftp.net , находящемуся на сервере txt.org , осуществляется по протоколу http. В таблице фрагменты адреса файла закодированы буквами от А до Ж. Запишите последовательность этих букв, кодирующего адрес указанного файла в сети Интернет. А) .net Б) ftp В) :// Г) http Д) / Е) .org Ж) txt

д) Доступ к файлу www.txt, находящемуся на сервере ftp.net, осуществляется по протоколу http. В таблице фрагменты адреса файла закодированы буквами от А до Ж. Запишите последовательность этих букв, кодирующую адрес указанного файла. А) .txt Б) http В) / Г) :// Д) .net Е) www Ж) ftp.

е) Доступ к файлу http.net, находящемуся на сервере www.net, осуществляется по протоколу ftp. В таблице фрагменты адреса файла закодированы буквами от А до Ж. Запишите последовательность этих букв, кодирующих адрес указанного файла в сети Интернет.

А	Б	В	Г	Д	Е	Ж

://	.net	.txt	http	ftp	/	www

ж) Доступ к файлу uk.net, находящемуся на сервере org.de, осуществляется по протоколу http. В таблице фрагменты адреса файла закодированы буквами от А до Ж. Запишите последовательность этих букв, кодирующих адрес указанного файла в сети Интернет.

А	Б	В	Г	Д	Е	Ж
org	uk	.de	net	://	ftp	/

2. Петя записал IP-адрес школьного сервера на листочке бумаги и положил в карман куртки. Петина мама случайно постирала куртку вместе с запиской. После стирки Петя обнаружил в кармане четыре обрывка с фрагментами IP-адреса. Эти фрагменты обозначены буквами А, Б, В и Г. Восстановите IP-адрес. В ответе укажите последовательность букв, обозначающих фрагменты, в порядке, соответствующем IP-адресу.

А)

3.133	3.14	.82	22
А	Б	В	Г

Б)

.32	3.212	2.20	20
А	Б	В	Г

Контрольные вопросы.

1. Дать понятие сервера и клиента.
2. Перечислить типы сетей по типу и организации (по группе признаков).
3. Что такое локальная компьютерная сеть, и каких видов она бывает?
4. Что такое протокол коммуникации TCP/IP - назначение?
5. Что такое Telnet?
6. Какие виды поисковых систем вы знаете, и чем они отличаются?
7. Дать понятие протокола коммуникации.
8. Привести примеры типов сетей и изобразить их.
9. Дать понятие корпоративной компьютерной сети.
10. Написать правило составления IP-адреса.
11. Что такое FTP?

12. Что такое электронная почта и электронное письмо? Записать правило составления электронных адресов.
13. Дать понятие компьютерной сети.
14. Дать понятие региональной компьютерной сети.
15. Что такое глобальная компьютерная сеть?
16. Написать правило составления доменных адресов.
17. Что такое Gopher?

Что такое WWW, Web-страница, Web-браузер? Записать правило составления URL-адресов.

Понятие о гипертекстовом документе

Гипертекстовые документы создаются с помощью особого языка Hyper Text Markup Language (HTML)- язык гипертекстовой разметки. Он содержит особые команды – тэги (tag) (дескрипторы разметки текста). По этим командам в окне программы-клиента (браузера) информационные материалы просматриваются в соответствующем виде.

В мае 1996 года появился стандарт HTML 3.2, который разработала организация World Wide Web Consorcium (W3C). 18 декабря 1997г. была утверждена новая версия стандарта: HTML 4. Все браузеры и серверы поддерживают этот стандарт.

Существует два наиболее распространённых типа браузера **Microsoft Internet Explorer и Netscape Communicator**, которые разработали две конкурирующие фирмы: Microsoft и T\Netscape. Каждая фирма разработала серию браузеров, причём более поздние версии имеют больше возможностей, но требуют использования более мощных компьютеров. Следует учитывать то, что не все тэги Microsoft читаются браузерами Netscape и наоборот.

Все что необходимо, чтобы прочитать HTML-документ - это WEB-браузер, который интерпретирует тэги HTML и воспроизводит на экране документ в виде, который ему придает автор.

Запомните, что основное преимущество HTML заключается в том, что ваш документ может быть просмотрен на WEB-браузерах различных типов и на различных платформах.

Понимание HTML. Как это работает.

HTML - это обычный, текстового вида файл, в котором то, что мы обычно видим на страничках, перемежается невидимым для просмотра из браузера кодом. Вот этот-то невидимый код и есть язык разметки HTML.

HTML - это не язык программирования, - он служит лишь для разметки странички, придания определенного вида тому или иному элементу, будь то таблица, текст или картинки.

Параметры отображения элементов задаются при помощи тэгов, в которых и задается желаемый вид того или иного элемента нашей странички.

HTML-тэги могут быть условно разделены на две категории:

- тэги, определяющие, как будет отображаться WEB-браузером тело документа в целом
- тэги, описывающие общие свойства документа, такие как заголовок или автор документа

HTML странички состоят из тэгов (пример: <hr> - горизонтальная линия), т.е. все, что находится между < и > называется тэгом. Все HTML документы начинаются с тэга <html> и заканчиваются тэгом </html>. Тэги бывают разными: только открывающимися и открывающимися/закрывающимися. Пример только открывающегося тэга <br>. Пример открывающегося/закрывающегося тэга: <html></html>. Есть тэги, которые обязательны.

Обязательные тэги

```
<html>
<head>
</head>
<body>
</body>
</html>
```

Принцип наследования тэгов. **Тэги-контейнеры.**

Прежде всего, стоит определить то, как эти самые тэги располагаются. Дело в том, что тэги в большинстве своем состоят из двух частей - это открывающий (он же - содержащий параметры) и замыкающий, то есть, "конец" тэга. Заданные в тэге параметры действуют только между его началом и концом, то есть, только внутри тэга:

```
<начало 1-го тэга>
    текст
    <начало 2-го тэга>
        текст
        <начало 3-го тэга>
            текст
        <конец 3-го тэга>
    <конец 2-го тэга>
<конец 1-го тэга>
```

Вот эти "отступы" от левого края - они тоже не случайны - они автоматически выставляются программами, формирующими код HTML для упрощения его восприятия. Достаточно понимать, что чем дальше от левого края отстоит тэг, тем "глубже" он лежит, тем большее количество "старших" тэгов может на него действовать. В некоторых случаях параметры ранее заданного тэга могут распространяться и на тэг, размещенный в нем, то есть на содержимое *вложенных* тэгов.

Также следует заметить, что некоторые параметры вложенных тэгов могут воздействовать и на вышестоящие, "старшие" тэги. Например, это типично для таблиц, вложенных одна в другую.

Все пробелы больше одного браузеры просто "не замечают". То есть, сколько ни ставьте вы пробелов в тексте, браузер отобразит все равно с одним пробелом. Пробелы можно наставить принудительно. Делается это размещением в коде: ; - обозначение пробела. Неразрывного пробела. То есть в этом месте будет пустое пространство. Но если простой текст с пробелами браузер может перенести на другую строку на месте любого пробела, то слова, разделенные ; , он воспримет как единое целое. Вот потому не стоит злоупотреблять "гибкостью" и слишком часто использовать обозначение ; . Не стоит использовать его и для "выравнивания" текста внутри таблицы. Для этого есть более простые и правильные способы. Не рекомендую употреблять ; больше двух-трех раз подряд. Можете считать это правилом, поскольку при большем количестве символов наверняка проще прописать требуемое выравнивание в тэг нужного элемента. Это будет и проще, и нагляднее, и "чище", в плане того, что ваши тексты не будут чересчур тяжелыми и долгогрузящимися.

Создание HTML-документов

HTML – документы хранятся в текстовых файлах. Существует много специализированных средств создания HTML – документов, однако создаваемый этими средствами текст HTML – документа, во-первых, часто содержит ненужные символы и команды, а во-вторых, затрудняет процесс изучения языка HTML, скрывая детали разработки документа. Редактор «Блокнот» является самым простым средством разработки HTML – документа.

Для того, чтобы сохранить набранный в редакторе «Блокнот» текст, необходимо обратиться к пункту – меню «Файл» и выбрать «Сохранить как». Далее в меню «Тип файла» необходимо выбрать пункт «Все файлы». Имя файла должно иметь расширение **htm**. Просмотр документа осуществляется в браузере (чаще всего в Internet Explorer, Netscape Navigator или Opera). Для просмотра документа в браузере, необходимо обратиться к пункту – меню «Файл» и выбрать «Открыть».

Структура HTML – документа

(Web- страница), как правило, начинается с дескриптора (тэга) **<html>** и заканчивается **</html>**. Тэги можно набирать как строчным, так и прописными буквами.

Составными частями документа является заголовок (шапка документа) и главная часть (тело). Заголовок располагается в начальной части HTML – документа. Заголовок документа представляет собой особый тип информационного блока, в котором содержатся необходимые для работы с документом сведения, например, его название, описание связей с другими документами. Вся эта информация располагается между дескрипторами разметки **<head> и </head>**. В частности, здесь между дескрипторами **<title>...</title>** задаётся имя HTLM – документа. В

частности, этим именем браузер пользуется, для обозначения документа строке заголовка окна в списке закладок.

В теле HTLM – документа располагается содержимое самой Web-страницы с графикой, текстом и ссылками на другие ресурсы Интернет. Эта главная часть документа дескрипторами **<body>** и **</body> , которые, в свою очередь, заключаются между** дескрипторами **<html>** и **</html>.** Задавая значения атрибутов (параметров) дескриптора **<body>**, можно осуществить некоторые важные установки параметров Web- страницы, которые, в основном, касаются внешнего вида Web- страницы. Все, что написано внутри данных тэгов можно назвать телом документа.

Пример: <TITLE> Заголовок документа </TITLE>

Внимание! Дополнительные пробелы, символы табуляции и возврата каретки, добавленные в исходный текст HTML-документа для его лучшей читаемости, будут проигнорированы WEB-браузером при интерпретации документа. HTML-документ может включать вышеописанные элементы, только если они помещены внутрь тэгов <PRE> и </PRE>. Более подробно о тэгах <PRE> будет написано ниже.

Внимание! Технически, стартовые и завершающие тэги типа <HTML>, <HEAD> и <BODY> необязательны. Но настоятельно рекомендуется их использовать, поскольку использование данных тэгов позволяет WEB-браузеру уверенно разделить заголовочную часть документа и непосредственно смысловую часть.

Комментарии

Как любой язык, HTML позволяет вставлять в тело документа комментарии, которые сохраняются при передаче документа по сети, но не отображаются браузером. Синтаксис комментария: <!-- Это комментарий -->

Создание простой HTML странички.

Для начала откроем любой блокнот. Например, стандартный блокнот Windows(Пуск>Программы>Стандартные>Блокнот). Открыли? Хорошо. Теперь напишите следующий текст в блокноте

```
<html>
 <head>
   <title>
       Мой первый шаг
    </title>
 </head>
 <body>
  Здравствуйте, это моя первая страница.
 <br> Добро пожаловать!
 </body>
</html>
```

А теперь сохраним этот документ, присвоив ему имя *.html

Создадим на диске папку с именем **HTML странички** и сохраним туда наш документ. Теперь откроем имеющийся браузер, заходим в меню Файл>Открыть. Нажимаем на кнопку Обзор и находим наш документ, который мы сохранили, нажимаем кнопку Открыть. Посмотреть пример. Для того, чтобы отредактировать файл, необходимо щелкнуть ПКМ по открытому htm-файлу и выбрать просмотр HTML-кода, или пункт меню Вид и выбрать просмотр HTML-кода, или по кнопке Править в панели инструментов и выбрать соответствующий пункт.

Если вы измените что-нибудь в нашем *.html файле и захотите посмотреть изменения, то откройте браузер и загрузите ваше страничку. Если она у вас уже загружена, то просто нажмите F5 или на кнопку Обновить. Если мы чего-то изменили в нашем *.html документе (в блокноте), то, чтобы посмотреть как это выглядит в Internet Explorer, надо не забывать нажимать в браузере кнопку ОБНОВИТЬ. Если изменений не видно, то это значит, что вы где-то что-то неправильно написали, или забыли сохранить документ.

Основные атрибуты дескриптора ***<body>***

Атрибуты пишутся внутри тэга.

Установление цвета фона и текста.

Атрибут	**Название атрибута**
bgcolor	Цвет фона
text	Цвет текста
background	Повторяющаяся фоновая графика

Значение атрибута **bgcolor** задаётся как шестнадцатеричный код трехбайтного RGB-цвета (совместно со знаком #) или как имя одного из 16-ти предлагаемых по умолчанию (стандартных) цветов. Значение атрибута text задаётся так же, как и для атрибута **bgcolor.**

Пример использования атрибутов: **<body bgcolor =red text=# 0A23FA >.**

<body background="Pictures\fon.gif">. Вместо одноцветной заливки в качестве фоновой заставки можно использовать иллюстрацию, которая повторяется как узор на обоях, заполняя весь фон странички. Следует задавать фоновый цвет даже в случае использования фоновой иллюстрации, поскольку возможны ситуации, когда пользователь запрещает загрузку фоновой иллюстрации: **<body bgcolor = red text=# FA23FA background="Pictures\fon.gif">**

Наберите и просмотрите пример

```
<html>
 <head>
  <title>Мой второй шаг</title>
 </head>
 <body bgcolor=#0A23FA text=yellow>
  Здравствуйте, это моя вторая страничка.
```

```
<br>
<font color=red>Добро пожаловать!</font>
</body>
</html>
```

Параметр **text** устанавливает цвет текста по всему документы, т.е. если вы укажете этот атрибут, то любой текст напечатанный в документе будет одного цвета. Если хотите изменить цвет определенного текста, используйте тэг . Тэг - многофункционален. Им может задаваться не только цвет текста в конкретной части документа, но и размер шрифта, и вид шрифта (Arial), но об этом чуть позже.

Атрибут **topmargin=число** (<body **topmargin=0**>) устанавливает расстояние текста до верха окна, а **leftmargin=число** (<body **leftmargin=0**>) устанавливает расстояние до левого края окна.

Наберите и просмотрите пример:

```
<html>
<head>
<title>Мой второй шаг</title>
</head>
<body topmargin=100 background="Pictures\BETON_BG.GIF" text=blue>
Здравствуйте, это моя вторая страничка.
<br>
```

Название цвета на английском языке	**Название цвета на русском языке**	**Представление цвета в виде кода**		
		красная составляющая	**зелёная составляющая**	**синяя составляющая**
Aqua	Бирюзовый	0	Ff	Ff
Black	Чёрный	0	0	0
Blue	Синий	0	0	Ff
Fuchsia	Розовый	Ff	0	Ff
Gray	Серый	80	80	80
Green	Зелёный	0	80	0
Lime	Светло-зеленый	0	Ff	0
Marron	Коричневый	80	0	0
Navy	Ультрамариновый	0	0	80
Olive	Оливковый	80	80	0
Purple	Пурпурный	80	0	80
Red	Красный	Ff	0	0

Silver	Серебряный	C0	C0	C0
Teal	Тёмно-зелёный	0	80	80
Yellow	Жёлтый	Ff	Ff	0
White	Белый	Ff	Ff	Ff

```
  <font color=red>Добро пожаловать!</font>
 </body>
</html>
```

Форматирование текста.

*Изучаемые тэги: <basefont>, <font>, <pre>, <i>, <b>, <u>, <big>, <small>, , *

Теоретическая часть

В некоторых случаях для большей выразительности требуется начать печать с новой строки. Для этой цели используется тэг **
. Атрибут **clear в тэге **
** используется для того, чтобы остановить в заданной точке обтекание текстом объекта и затем продолжить текст в пустой области за объектом. Продолжающийся за объектом текст выравнивается в соответствии со значениями **left, right** или **all** атрибута **clear**:

<br clear=left> Текст будет продолжен, начиная с ближайшего пустого левого поля.

<br clear=right> Текст будет продолжен, начиная с ближайшего пустого правого поля.

<br clear=all>Текст будет продолжен тогда, когда и левое, и правое поля окажутся пустыми.

Когда в гипертекстовый документ необходимо включить текст, уже имеющий форматирование (отступы, пробелы, таблицы…), то используется пара тэгов: **<pre>, </pre>**. В этом случае для отображения текста используется моноширинный шрифт.

Адресная часть страницы

Адресная информация располагается между двумя тэгами: **<address>** и **</address>**. Соответствующий текст выделяется курсивом.

Горизонтальный разделитель ***<hr>*** имеет атрибуты:
align – выравнивание по горизонтали, **noshade** – отключение 3D- графика, **size** – высота линии в пикселях, **width** – ширина линии в пикселях, **color** – цвет целесообразно ставить без 3D- графики.

<u>Наберите и просмотрите пример:</u>

```
<html>
 <head>
  <title>Мой третий шаг</title>
 </head>
 <body topmargin=100 background="Pictures/BETON_BG.GIF" text=blue>
```

Здравствуйте, это моя третья страничка.
<hr align=right color=green size=10 width=400 noshade>

Добро пожаловать! в некоторых случаях для большой выразительности требуется начать печать с новой строки. для этой цели используется тэг br. Когда в гипертекстовый документ необходимо включить текст, уже имеющийся форматирование, то используется пара тэгов pre.

<hr align=left size=10 width=400>
<address>ул.Добросельская, д.179а</address>
</body>
</html>

Форматирование текстовых абзацев

Дескриптор ** ** задаёт начало абзаца и вставляет соответствующий пустой интервал для отделения данного абзаца от предыдущего. На практике концевой дескриптор ** ** можно опустить, так как он обычно игнорируется большинством браузеров. Дескриптор ** ** имеет атрибут **align.**

С помощью параграфов мы можем центрировать текст: `<p align=center>текст</p>`, выровнять текст по левому краю: `<p align=left>текст</p>` или по правому краю документа: `<p align=right>текст</p>`.

Наберите и просмотрите пример:

```
<html>
 <head>
  <title>Мой четвертый шаг</title>
 </head>
 <body topmargin=100 background="Pictures/BETON_BG.GIF" text=blue>
  <p align=center>
  Здравствуйте, это моя третья страничка.
  <hr align=right color=green size=10 width=400 noshade>
  <br>
   <font color=red>Добро пожаловать! в некоторых случаях для большой выразительности требуется начать печать с новой строки. для этой цели используется тэг br. Когда в гипертекстовый документ необходимо включить текст, уже имеющийся форматирование, то используется пара тэгов pre.
   </font>:-))))
   </p>
```

```
    <hr      align=left  size=10 width=400>
    <address>ул.Добросельская, д.179а</address>
  </body>
</html>
```

Есть тэг альтернативный <p align=center>: `<center>текст</center>`

Примените данный тэг к предыдущему примеру.

У тэга <p> существует еще и четвертое, но опасного значение (атрибута): `<p align=justify>текст</p>`. Он выравнивает текст по обоим краям документа. Он не работает в старых версиях браузеров.

Наберите и просмотрите пример:

```
<html>
 <head>
  <title>Мой пятый шаг</title>
 </head>
 <body topmargin=100 background="Pictures/BETON_BG.GIF" text=blue>
  <center>
  <h2>
  Здравствуйте, это моя пятая страничка.</h2>
```

<hr align=right color=green size=10 width=400 noshade>

```
<br>
  <font color=red>Добро пожаловать! в некоторых случаях для большой выразительности требуется начать печать с новой  строки. для этой цели используется тэг br. Когда в гипертекстовый документ необходимо включить текст, уже имеющийся форматирование, то используется пара тэгов pre.
  </font>:-))))
  </center>
  <hr align=left size=10 width=400>
  <address>ул.Добросельская, д.179а</address>
    <p align=justify>
  <font color=green>Я совсем недавно начал(а) знакомиться с виртуальной жизнью, но мне по давней традиции тоже захотелось создать свою домашнюю  страничку для моих новых виртуальных друзей и знакомых, чтобы они могли  посмотреть мои фотографии, почитать обо мне, черкнуть пару строчек в  мою гостевую книгу. А может и просто случайный посетитель вдруг захочет  познакомится со мной, и у меня появиться еще один виртуальный друг? :)</font>
  </p>
 </body>
```

</html>

Заголовки.

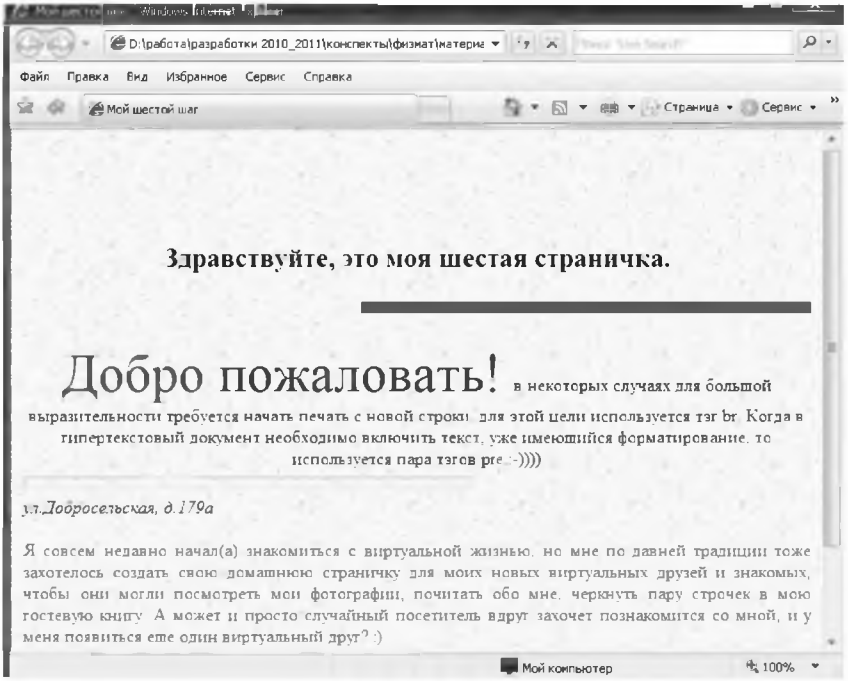
<H1> текст </H1>

<H2> текст </H2>
<H3> текст </H3>
<H4> текст </H4>
<H5> текст </H5>
<H6> текст </H6>От самого крупного до более мелкого. Заголовок выделяется жирным текстом.

Использование атрибута align:

- **align=left** – заголовок выравнивается по левому краю области текста.
- **align=center-** заголовок центрируется по горизонту.
- **align=right-** заголовок смещается к правому краю области текста.

В пример (см.ниже) введен заголовок третьего уровня <H2></H2>. Создайте такой же документ. При этом цвета заголовка, абзацев, адреса, выделенного предложения различны. Обратите внимание на выравнивание.



Размер и шрифт текста.

Установка размера шрифта

Чтобы задать размер шрифта для достаточно длинного текста, целесообразно использовать дескриптор разметки **<basefont>**, который может размещаться либо в заголовочном разделе, либо в теле документа.

<basefont size=n>. Здесь **n** может принимать значения от 1 (самый мелкий шрифт) до 7 (самый крупный шрифт).

Применив несколько **<basefont>**-дескрипторов в документе, можно установить базовые размеры шрифтов для отдельных разделов. Концевой дескриптор разметки **</basefont>** возвращает размер шрифта к значению **size=3**. Если концевой дескриптор **</basefont>** не указывать, то действие **<basefont>** с атрибутом size распространяется либо до конца документа, либо до следующего тэга **<basefont>**.

Чтобы изменить параметры шрифта для отдельного фрагмента текста, необходимо его «окружить» тэгами: **** и ****. Тэг **** имеет атрибуты: **color, size, face** (цвет, размер и тип шрифта).

**** Пример ****

Можно задавать размер шрифта относительным способом:
**** Пример **.** Здесь размер шрифта устанавливается на три единицы больше размера, задаваемого тэгом **<basefont>**.

<u>Наберите и просмотрите пример:</u>

```
<html>
  <head>
     <title>Мой первый шаг </title>
  </head>
  <body text="#336699" bgcolor="#000000">
      <center>
      <H3>Здравствуйте, это моя первая страница.</H3>
      <br>
      <font color=#CC0000> Добро пожаловать!</font> :) </center>
         <p align=justify>
Я совсем недавно начал(а) знакомиться с виртуальной жизнью, но мне по
давней традиции тоже захотелось создать свою домашнюю страничку для
моих новых виртуальных друзей и знакомых, чтобы они могли посмотреть
мои фотографии, почитать обо мне, черкнуть пару строчек в мою гостевую
книгу. А может и просто случайный посетитель вдруг захочет
познакомится со мной, и у меня появится <font size=24> еще один
виртуальный друг? :)
    </font>
      </p>
  </body>
</html>
```

Создайте подобный документ по рисунку.

Курсив, жирный текст, подчеркнутый текст.

Тэги логического форматирования текста

Текст, расположенный между тэгами <**big**>, </**big**> или **<small>, </small>,** будет соответственно больше или меньше на одну ступень по сравнению с предыдущим текстом. Полужирное начертание текста обеспечивается тэгами <**strong>, **, а курсив – **, .**

Тэги физического форматирования текста

В отличие от логического форматирования эти тэги отображают текст заданным образом независимо от настроек браузера. Следует отметить, что предпочтительным является логическое форматирование, так как оно позволяет более гибко управлять видом документа при использовании более сложных и совершенных средств создания гипертекстовых документов.

Примеры физического форматирования текста:

Курсив **Ваш текст** - будет выглядеть так: *курсив*(Italic)

Жирный текст **Ваш текст** - будет выглядеть так: **жирный текст**(Bold)

Подчеркнутый **<u>Ваш текст</u>** - будет выглядеть так: <u>подчеркнутый текст</u>

Этими тэгами можно комбинировать. Например: Курсив и жирный текст Ваш текст - будет выглядеть так: ***курсив и жирный текст***

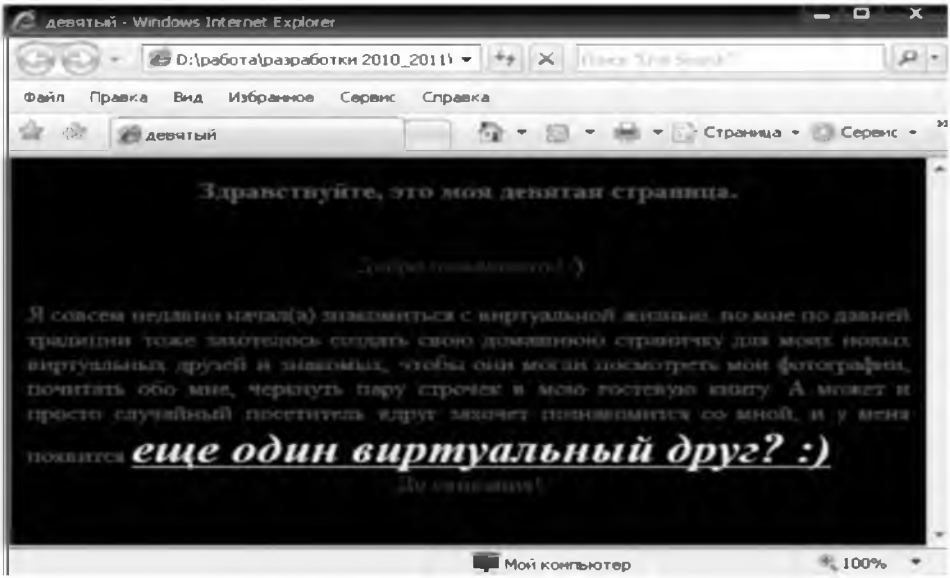
Наберите и просмотрите документ.

В данном упражнении используются различные способы выделения фрагментов текста: курсив, подчеркивание, использование жирного шрифта…. Кроме того, используется предварительно отформатированный фрагмент текста. Текст упражнения сохранить в файле

```
 <html>
<head>
 <title> Форматирование текста в документе</title>
 <basefont size=4>
</head>
<body>
 <center>
 <font color=blue face=Arial size=+1>
   Тэги физического форматирования текста
  </font>
 </center>
<p align=justify>


        В отличие от <em>логического</em> форматирования эти тэги
отображают
<strong>текст</strong> заданным образом
 <u>независимо</u>
  от настроек браузера.
<br clear=right>
<pre>
Следует отметить, что <font color=red face=Arial
size=+1>предпочтительным</font> является логическое форматирование,
так как оно позволяет боле гибко управлять видом документа при
использовании более сложных и совершенных средств
создания гипертекстовых документов.
</pre>
</body>
</html>
```

Создайте подобный документ по рисунку, используя тэги физического форматирования.



Встраивание графики в гипертекстовый документ

Изучаемый тэг: <img>

Теоретическая часть

Атрибуты тэга <img>

Иллюстрации вставляются в Web-страницы с помощью дескриптора **<img>**, концевой дескриптор не предусмотрен. Традиционными для графических файлов являются форматы **GIF** и **JPEG**.

Атрибуты дескриптора <img>

Атрибут	**Функция**
align	Выравнивание по горизонтали
border	Ширина рамки вокруг иллюстрации(в пикселах)
height	Высота иллюстрации(в пикселах)
hspace	Расстояние от иллюстрации до текста по горизонтали(в пикселах)
src	Исходный файл иллюстрации
vspace	Расстояние до текста по вертикали(в пикселах)
width	Ширина иллюстрации (в пикселах) по умолчанию реальный размер картинки
Alt	Краткое описание картинки, при наведении мышки всплывает надпись

align=top – выравнивает верхний край иллюстрации по верхнему краю самого «высокого» элемента в текущей текстовой строке, не меняя позицию по горизонтали.

align=middle – центрирует иллюстрацию по вертикали на оси текущей текстовой строки, не меняя позицию по горизонтали.

align=bottom – выравнивает нижний край иллюстрации по горизонтали на оси текущей текстовой строки.

align=left – смещает графику к левому краю рабочей зоны. Последующий текст начинает «обтекать» графику.

align=right – смещает графику к правому краю рабочей зоны. Последующий текст начинает «обтекать» графику.

Значение атрибута **border=n** позволяет задать рамку иллюстрации шириной в ***n*** пикселов. Атрибут **src** должен быть задан для любой иллюстрации.

Пример использования тэга **<img>:**

<img src=”Pictures1\logotip.gif” align=right height=100 width=220 border=4>

Здесь изображение вставляется в страницу со смещением к правому краю документа. Оно снабжается рамкой шириной в четыре пикселя. Высота изображения в документе составляет 100, а ширина – 220 пикселей. **=”Pictures1\logotip.gif”** – прописывается полный путь к картинке, если она не находится в текущем каталоге. Можно указывать и сайт, где находится картинка: <img src="сайт/имя картинки или сайт/папка/имя картинки">.

Пример: <img **src="http://www.igorenat.hut.ru/b2.gif или http://www.igorenat.hut.ru/ban/b2.gif"**>

Alt=”Описание картинки”

Для того, чтобы картинку сделать фоновой, нужно в дескрипторе Body прописать **<body bgcolor=yellow background=”Pictures\fon.gif” txt=red>.** При этом цвет фона следует также назначать, т.к.картинка может не загрузиться.

Наберите и просмотрите пример:

```
<html>
<head>
 <title>Мой первый шаг с картинками</title>
</head>
<body text=aqua bgcolor=marron>
<center>
 <H3>Здравствуйте, это моя первая страница с картинкой.</H3>
 <br>
 <font color=olive> Добро пожаловать!</font> :)
</center>
<p align="justify">
 <img src="Pictures1\143028.JPG"> Я совсем недавно начал(а) знакомиться с виртуальной  жизнью, но мне по давней традиции тоже захотелось
```

создать свою домашнюю страничку для моих новых виртуальных друзей и знакомых, чтобы они могли посмотреть мои фотографии, почитать обо мне, черкнуть пару строчек в мою гостевую книгу. А может и просто случайный посетитель вдруг захочет познакомиться со мной, и у меня появится еще один виртуальный друг? :)
</p>
</body>
</html>

Получилось не очень красиво. Попробуем улучшить. Введем дополнительные атрибуты с параметрами: align="left" HSPACE=30 VSPACE=5 alt="это пустыня" . Наша картинка будет прижата к левому краю экрана, текст будет обтекать ее справа, расстояние до текста по горизонтали - 30 пикселей, по вертикали - 5 пикселей (чтобы красиво все смотрелось), ну, и если вы наведете на картинку курсор, то выскочит надпись - "это пустыня".

***Задание**: измените предыдущий пример так, чтобы 1. картинка была прижата к левому краю, и при наведении на нее курсора всплывало ее описание; 2. картинка была выровнена по нижнему краю по верхней строке текста, каждый абзац с новой строки и разного форматирования текста.*

Наберите и просмотрите пример.

```
<html>
<head>
<title>Все о работе с картинками. Вставка картинок и их атрибуты.</title>
</head>
<body background="Pictures\1W.GIF" bgproperties="fixed" text="yellow"
bgcolor="black" link="#CC0000" vlink="#FFFF00" alink="#008000">
фоном будет картинка как водяной знак, чтобы фоном было просто
картинка
удалите атрибут bgproperties="fixed".
<center><h3>Добро пожаловать! Это моя фотогалерея.</h3> </center>
<p align="justify">
<img src="Pictures1\144027.JPG" align="left" Hspace="5" alt="в море"
border="2">
вставили картинку. Равняется левому краю, текст справа от картинки на
расстоянии 5 пикселей, рамка 2 пикселя. При наведении на картинку мыши
появится описание "в море".
<font color="green" size="5">Эта фотография была сделана на в мирное
время в бархатный сезон. наступило утро. на берегу никого не видно.
солнце встает из-за горизонта. все просыпается. </font>
Компьютеры могут объединяться в сеть различными способами.
Логический и физический способ соединения компьютеров, кабелей и
```

других компонентов, в целом составляющих сеть, называется ее топологией. Иными словами, общая схема соединения компьютеров в локальные сети называется топологией сети. Топология характеризует свойства сетей, не зависящие от их размеров. При этом не учитываются производительность и принцип работы этих объектов, их
типы, длины каналов, хотя при проектировании эти факторы очень важны.
Наиболее распространенные типы сетей:1. Линейная – 2. Кольцевая – 3. Древовидная – 4. Звездообразная – 5. Ячеистая –6. Полносвязная – в ней имеется ветвь между любыми двумя компьютерами.
Классификация сетей.
Все многообразие компьютерных сетей можно классифицировать по группе признаков: • Территориальная распространенность (по степени географического распространения); • Ведомственная принадлежность; • Скорость передачи информации; • Тип среды передачи.
По территориальной распространенности сети могут быть локальными, глобальными, и региональными (городские), корпоративными. По принадлежности различают ведомственные и государственные сети. Ведомственные принадлежат одной организации и располагаются на ее территории. По скорости передачи информации компьютерные сети делятся на низко-, средне- и высокоскоростные. По типу среды передачи разделяются на сети коаксиальные, на витой паре, оптоволоконные, с передачей информации по радиоканалам, в инфракрасном диапазоне. Советую вам потренироваться еще с атрибутами. Вставляйте их, изменяйте значения. Если перейдете сразу на следующий урок, то просто забудете их. А пока на этом уроке все.
</p>
</body>
</html>

Таблицы

Изучаемые тэги: <table>, <caption>, <tr>, <th>, <td>

Тэги для построения таблицы

Таблицы широко используются при построении гипертекстовых документов, так как позволяют удобно разместить на странице различную текстовую и графическую информацию. Таблица формируется дескрипторами **<table>** и **</table>,** дескрипторы **<tr>** и **</tr>** выделяют строки, которые, в свою очередь, разбиваются на ячейки заголовков **<th>** или ячейки данных **<td>**. Ячейки заголовков отображаются более жирным шрифтом. В ячейки можно помещать текст и изображения.

Атрибуты дескриптора **<table>**

Атрибут	**Функция**
align	Выравнивание по горизонтали

border	Ширина рамки таблицы
cellpadding	Расстояние между содержимым и рамкой ячеек таблицы
cellspacing	Расстояние между ячейками таблицы
width	Ширина таблицы (задается в процентах от ширины страницы или в пикселях)
bgcolor	Цвет фона таблицы (задается так же, как и фон всего документа)

Обычно таблица отображается на левом краю страницы. Однако ее можно сместить, используя специальные дескрипторы форматирования (например, ** **). Кроме того, таблица может выравниваться атрибутом **align** дескриптора **<table>**: **align=left** выравнивает таблицу по левому краю страницы; **align=right** выравнивает таблицу по правому краю.

Таблица может снабжаться рамкой. **<table border=n>.** Здесь **n** определяет ширину рамки таблицы. Если атрибут **border** отсутствует или имеет нулевое значение, то рамки у таблицы не отображаются. Если указать атрибут **border**, а значение не задать, то по умолчанию **n**=1.

Тэги **<tr>** и **</tr>** имеют атрибуты: **align** (значения: **left, right, center**) **width, valign** (выравнивание по вертикали, основные значения: **top, center, bottom**), **colspan** (объединения столбцов) и **rowspan** (объединение строк).

Таблица может иметь заголовок, расположенный либо над ней, либо под ней. Расположение заголовка определяется тэгом **<caption>…</caption>** с атрибутом **align**, который принимает либо значение **top** (заголовок – над таблицей), либо – **bottom** (заголовок – под таблицей). При отсутствии атрибута считается, что **align=top.**

Итак, перед вами таблица из двух строк и трех столбцов (ячеек).

```
<table>
<tr></tr>
<tr></tr>
</table>
```

Сначала задаем строки. В нашем примере их две. Теперь в каждой строке зададим по три столбца (ячейки).

Итак, к

```
<table>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
```

```
</table>
```

Итак, теперь нам надо заполнить получившийся каркас:

```
<table>
<tr>
<td>1x1</td>
<td>1x2</td>
<td>1x3</td>
</tr>

<tr>
<td>2x1</td>
<td>2x2</td>
<td>2x3</td>
</tr></table>
```

1x1 1x2 1x3

2x1 2x2 2x3

Набрать и рассмотреть следующий пример:

```
<table>
<tr>
<td bgcolor=yellow>1x1</td>
<td bgcolor=red>1x2</td>
<td bgcolor=yellow
>1x3</td>
</tr>
<tr>
<td bgcolor=red >2x1</td>
<td bgcolor=yellow >2x2</td>
<td bgcolor=red>2x3</td>
</tr>
</table>
```

1	2	3	
1x1	1x2	1x3	1
2x1	2x2	2x3	2

Задание: задать и вставить в предыдущий пример ширину и высоту в пикселях для столбцов (ячеек). См.рисунок.

1x1	1x2	1x3
2x1	2x2	2x3

Вы можете задать высоту и ширину для всей таблицы, тогда все ячейки (столбцы) и ряды поделят данное им пространство поровну, если не задавать им это пространство персонально (в процентах от общей ширины (высоты) таблицы или пикселях).

1x1	1x2	1x3
2x1	2x2	2x3

Поговорим о вертикальном выравнивании содержимого таблицы, т.е. о том, как можно сделать так, чтобы

1x1	1x2	1x3
2x1	2x2	

содержимое ячейки не только располагалось ровно посередине ее (как по умолчанию), а еще вверху или внизу. Вертикальное выравнивание задается следующим атрибутом - **valign**=middle (top, bottom) - содержимое конкретной ячейки будет находиться в середине ячейки (наверху или внизу).

Задание: изобразить две таблицы с заданным выравниванием (см.рисунки).

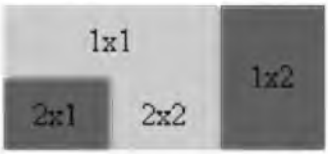
В этой главе мы поговорим о параметрах colspan и rowspan. Colspan - определяет количество столбцов, на которые простирается данная ячейка, а rowspan - количество рядов (эти параметры могут принимать значение от 2 и больше, т.е. наша ячейка может растягиваться на два и более столбца (ряда)).

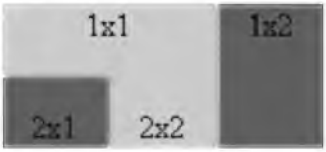
Наберите и просмотрите пример:

```
<table>
<tr>
<td height=35 bgcolor=yellow colspan=2>
<center>1x1</center> </td>
<td width=50 bgcolor=red> <center>1x2</center> </td>
</tr>
<tr>
<td height=35 width=50 bgcolor=red>
<center>2x1</center> </td>
<td width=50 bgcolor=yellow> <center>2x2</center>
</td>
<td width=50 bgcolor=red> <center>2x3</center> </td>
</tr>
</table>
```

Обратите внимание, на то, что параметр width для ячейки 1x1 в нашем примере не указан, если вас так тянет задавать этот параметр, то в нашем примере для ячейки 1x1 его надо было бы прописать равным 100 пикселям, т.к. все-таки ячейка 1x1 длинее других в два раза. И второе, на что прошу вас обратить внимание, в нашем примере нет ячейки 1x3, т.е. в первом ряду всего лишь две ячейки, т.к. ячейка 1x1 равна сама по себе двум ячейкам по длине (что мы и указали параметром colspan). Если бы мы прописали ячейку 1x3, тогда у нас получилась бы (см.рис.):

Задание: напишите код для двух заданных таблиц.





От расстояния между ячейками в таблицах можно с помощью атрибута cellspacing, равного нулю:

```
<table cellspacing=0>
```

Можно наоборот увеличить пространство между ячейками, допустим пусть cellspacing=5.

Обычно атрибут cellspacing, рассматривается в руководствах и учебниках в паре с атрибутом cellpadding, который добавляет свободное пространство между содержимым ячейки и ее границами.

Наберите и просмотрите пример:

```
<table cellpadding=5
cellspacing=5>
<tr>
<td height=35 bgcolor=yellow colspan=2 valign=top>
<center>1x1</center> </td>
<td width=50 bgcolor=red rowspan=2 valign=top>
<center>1x2</center> </td>
</tr>
<tr>
<td height=35 width=50 bgcolor=red valign=bottom>
<center>2x1</center> </td>
<td width=50 bgcolor=yellow valign=bottom>
<center>2x2</center> </td>
</tr>
</table>
```

Задание 1.

Построить таблицу из трех строк и трех столбцов. Первая строка содержит одну заголовочную ячейку, которая объединяет три столбцов. Вторая ячейка второй строки должна объединить две строки (вторую и третью).

1	2	3
4	5 8	6
7		9

Задание 2.Вставить в предыдущую таблицу картинку в 4-ю и 6-ю картинки, а 5-8-ю – картинку сделать как фон ячейки.

Вложенные таблицы - это обычные таблицы, которые располагаются в ячейках другой таблицы.

Наберите и просмотрите пример:

```
<table>
<tr>
```

```
<td bgcolor=white><center>ячейка 1</center>
 <table border=1 bordercolor=red cellspacing=1>
<tr>
<td bgcolor=black><font color=white>таблица 2 ячейка 1</font></td>
</tr>
</table>
 </td>
</tr>
<tr>
<td bgcolor=white><center>ячейка 2</center>
 <table border=1 bordercolor=red cellspacing=1>
<tr>
<td bgcolor=black><font color=white>таблица 3 ячейка
1<font></font></font></td>
</tr>
</table>
 </td>
</tr>
</table>
```

ячейка 1

таблица 2 ячейка 1

ячейка 2

таблица 3 ячейка 1

Этот пример рассчитан на **Internet Explorer**. На **Netscape N** нужно будет применять картинку как фон вложенной таблицы. Делается этот так <td **background="картинка"**>. В пустые ячейки (<td></td>) лучше ставить пробел ** ** (<td>** **</td>).

Допустим у нас уже есть большая таблица, две таблица 3 ячейка 1 забиты текстом какого-то отчета, а третья между ними для красоты:

С каждым днем в Интернете появляется все больше юношей и девушек пользователей. Это особенные существа, роль которых в развитии современного общества не понятна, но тем не менее само их присутствие заметно.	Вот таблица, которая показывает, сколько бук, бяк юношей и девушек и других обитает в Интернете: юноши 65% населения девушки 20% населения др. 15% населения

***Задание**. Создать следующую таблицу из одного ряда, с тремя ячейками (столбцами), в третьей ячейке нельзя не заметить вложенную таблицу (фон в виде картинки).* Замечание: 1. вроде было бы логичнее прописать

просто белый фон для вложенной таблицы (bgcolor="#ffffff"), а не мучаться, создавая просто белую картинку и делая ее фоном (background="white.gif"), ведь так оно быстрее. Да, быстрее, но дело в том, что если IE (Internet Explorer) отображает параметр bgcolor для вложенных таблиц, то NN (Netscape Navigator) этот параметр для вложенных таблиц отказывается отображать, поэтому приходиться идти обходным путем, используя background.

<td width=10 background="white.gif"> </td>

Чтобы ячейка не пустовала, в нее введен   (символ неразрывного пробела). Это не просто так. Дело в том, что есть любители использовать конструкцию типа: <td></td>. Вот такую конструкцию NN просто напросто игнорирует, т.к. не любит пустых ячеек, поэтому для корректного отображения вашей таблицы везде вставляйте в пустые ячейки или , или мелкую картинку 1x1 пикселов.

Задание: Рамка вводится параметром border. Зададим рамку равную 3 пикселям: `<table border=3>`. Остальную таблицу допишете сами.

таблица с рамкой

Нашей рамке мы можем задать цвет. Пусть он будет черный в нашем примере: `<table border=3 bordercolor=black>`

таблица с рамкой

В руководствах и справочниках вы можете встретить параметры bordercolorlight и bordercolordark - эти параметры понимает только ИЕ (Internet Explorer), поэтому не рекомендую их употреблять. Вообще-то не

Различные таблицы в IE

Различные таблицы в NN

так трудно создать таблицу, гораздо труднее, чтобы это выглядело под разными браузерами одинаково. Возьмем те же рамки:

Списки

*Изучаемые тэги: <ol>, <ul>, *

Упорядоченные и неупорядоченные списки

Имеются следующие виды списков: ненумерованный (неупорядоченный – Unordered Lists), и нумерованный (упорядоченный - Ordered Lists). Тэги **
** и **
**, а также **
** и **
** – это тэги начала и конца ненумерованного и нумерованного списков соответственно. Тэг <**il**> (List Item) задает тэг элемента списка. В списке можно создать иерархию.

Тэги имеют атрибут type для выбора разных типов маркеров в ненумерованных списках и разных схем нумерации в нумерованных. Чтобы сменить тип маркера в середине списка, атрибут type включают в тэги <li>. После появления нового атрибута все последующие маркеры в списке будут иметь новый вид.

Пример ненумерованного списка:

- первый элемент списка
- второй элемент списка
- третий элемент списка

Данный список задается следующим образом:

<ul>
<li>первый элемент списка
<li>второй элемент списка
<li>третий элемент списка
**
**

Примеры задания маркеров: **<ul type=disk>**-маркеры в виде закрашенных кружков;

<ul type=circle>- маркеры в виде окружностей;

<ul type=square>- сплошные квадратные маркеры.

```
<ul>Ваш текст</ul>
<ul><ul>Два отступа</ul></ul>
<ul><ul><ul>Три отступа</ul></ul></ul>
```

Тэг **<ol>** вместе с атрибутом **type** позволяет создать нумерованные списки, используя в качестве номеров не только обычные числа, но и строчные и прописные буквы, а также строчные и прописные римские цифры. При необходимости можно смешивать эти типы нумерации в одном списке.

<ol type= 1> - список с нумерацией в формате 1., 2., 3., 4. …
<ol type= A> - список с нумерацией в формате A., B., C., D. …
<ol type= a> - список с нумерацией в формате a., b., c., d. …
<ol type= I> - список с нумерацией в формате I., II., III., IV. …

Задание. Составить многоуровневый список.

Ссылки.

Организация связей между HTML-документами осуществляется с помощью ссылок. Ссылки нужны нам для того, чтобы пользователь смог загружать другие страницы вашего сайта, картинки, другие сайты. А теперь рассмотрим тэг определяющий ссылку: **<a>** - тэг ссылки (от английского слова anchor – якорь).

Основные атрибуты тэга **<a>**:

href (HyperText Reference) – значение атрибута задает адрес документа, на который указывает данная ссылка;

target – данный атрибут используется, если окно браузера разбито на несколько фреймов, в этом случае значением атрибута является имя фрейма, где должен быть отображен результат ссылки;

name – используется, если необходимо отметить фрагмент текста, на который затем можно сослаться, значением атрибута является имя ссылки.

Примеры использования атрибутов:

Содержание,

** <img src="Pictures1\logitip.gif">**
,
****
Содержание .

В первом примере объектом ссылки является документ, который хранится в файле index.htm, который, в свою очередь, находится по адресу **www.vpti.vladimir.ru.** Слово **Содержание**, находящееся между тэгами **<a>** и **,** выделяется в тексте цветом.

Во втором примере ссылка производится по изображению **logitip.gif**. Оно в документе выделяется рамкой

В третьем примере слово **Содержание** указывает на место текста, которое помечено тэгом ****, причем этот текст будет отображен во фрейме с именем **left.**

Рассмотри данные примеры подробнее.

Тскст в атрибуте href пишите ссылку на страницу или сайт. Между тэгом <a> **вот тут** пишется текст. При клике на этот текст пользователь попадет на страничку или сайт написанный вами в атрибуте href.

Ссылки бывают разные. Пример ссылок.

1. <**a href="страничка"**>Ваш текст</**a**>

Пример: Главная страничка</**a**>

Этот пример ссылки показывает, что ссылка стоит на страничку index.htm, которая находится в одной папке с этой страничкой.

2. <**a href="имя папки/етраничка**">Ваш текст</**a**>

Пример: <**a href="about/index.htm"**>Обо мне</**a**>

Этот пример ссылки показывает, что ссылка стоит на страничку index.htm в папке about, которая находится в одной папке с этой страничкой.

3. <a **href="../страничка"**>Назад</**a**>

Пример: <a **href="../08.htm"**>Назад</**a**>

4. ****Ваш текст****

Пример: <**a hrcf="http://www.igorenat.hut.ru**">Мой сайт****

Этот пример ссылки показывает, что ссылка стоит на сайт www.igorenat.hut.ru.

5. Ссылки можно открыть в новом окне. Для этого нужен атрибут target.

****Текст ссылки****

Пример: ****Мои друзья****

6. Так же можно установить закладку. Для установления закладки нужно, в том месте, где ты хочешь поставить закладку написать:

<a **name="имя закладки">Тут можете написать текст, но не обязательно**

Пример: <a **name="top">**

Закладку мы поставили, а теперь нужно сделать на нее ссылку, чтобы на нее попасть. Делается это следующим образом:

<a **href="имя вашей закладки, на которую нужно перейти">**Текст

Пример: <a **href="#top">**Вверх

7. ссылка на ваш почтовый ящик(e-mail).

Напишите мне

Пример: <a **href="mailto:i-sof@mail.ru">**Напишите мне

Замените i-sof@mail.ru на ваш ящик и все. Mailto: должно быть всегда, если делаете ссылку на ящик.

8. Картинки как ссылки. Это почти то же самое, что и ссылка с текстом. Просто вместо текста ставится картинка. Пример ниже.

<img src="../001.jpg" Vspace="5" align="left" Alt="Военный сайт...">

Так как ссылка выделяется цветом, то для четкого ее изображения на цветном фоне документа в тэге **<body>** используются атрибуты: **link, alink, vlink**. Пример использования атрибутов: **<body bgcolor=blue text=yellow link=white alink=green vlink=red>**

Здесь **link=white** обозначает, что цвет ссылки, к которой еще не обращались – белый, **alink=green** – зеленый цвет активизированной ссылки, **vlink=red** – красный цвет ссылки, к которой уже обращались.

Набрать и просмотреть пример, при этом необходимо создать дополнительно 4 файла (contact.htm, servis.htm, products.htm, news.htm), которые будут появляться при работе ссылок, и положить в отдельную папку вместе с данным файлом.

```
<html>
<head>
<title>Ссылки</title>
</head>
<body link=red alink=green vlink=blue>
<h2 align=center>
предприятии</h2>
<p align=center>
<a href="news.htm" target="_blank">новости</a>
<a href="products.htm">товары</a>
```

```
<a href="servis.htm">услуги</a>
<a href="contact.htm">контакты</a>
</body>
</html>
```

Но поговорим особо о картинках, многие из вас, наверное, сталкивались с таким явлением: когда жмешь на маленькую картинку, то загружается большая в том же или в новом окне. Как это делается? Тут никаких хитростей: делаем ссылкой картинку (допустим small.jpg) и ссылаемся на другую картинку (допустим big.jpg):

`<a href="big.jpg"> <img src="small.jpg"> </a>.`

В этом случае большая картинка откроется в том же окне. Но как же сделать так, чтобы картинка (или любой другой файл-документ) открылась в новом окне? Дело в том, что у тэга <a> есть параметр target:

`<a href="big.jpg" target="_blank"> <img src="small.jpg"> </a>`Итак, target="_blank" - указывает на то, что документ (картинка в нашем случае), на который ведет ссылка, откроется в новом окне браузера.

Специальные символы.

Они нам нужны для того, чтобы писать символы типа < > & " и пробел. А вы спросите: **А почему просто не поставить < или >?** А потому-то и нельзя, потому что, как мы уже знаем, символ < и > используются при написании тэгов. Если вы напишете: Я сравнил 1 < 100 > 20, то у вас < 100 > будет как тэг. И что у вас получится? Так вот для чего они и нужны. Список этих символов ниже.

< - символ <
> - символ >
** ** - ставит пробел
" - " (кавычка)
& - & (амперсант)

Пример: <u>1<100>20</u>=
1<100>20

Запомните. Спецсимволы не пишутся в тэгах. Они как текст.

Фреймы

Рамочная структура документа

С помощью техники фреймов окно браузера разбивается на строки и столбцы, из которых образуется несколько независимых прямоугольных подокон (фреймов). Порядок разделения окна браузера на фреймы задается в рамочном фрагменте документа, ограниченном дескрипторами **<frameset>** и **</frameset>.** Этот фрагмент содержит также ссылки на отображаемые внутри фреймов HTML-документы. Внутри фрейма может

находиться целая фреймовая область, описываемая собственным дескриптором. Для описания самого фрейма используется дескриптор **<frame>**. HTML-код, определяющий разделение окна браузера на фреймы (рамочная структура окна), хранится в отдельном файле.

Атрибуты тэга **<frameset>**

Атрибут	Функция
name	Имя фрейма
src	HTML-документ для данного фрейма

Атрибуты тэга **<frame>**

Атрибут	Функция
cols	Количество и ширина столбцов
rows	Количество и высота строк
border	Ширина рамки (в пикселях)

Значения атрибутов тэга **<frameset>** задаются либо в пикселях, либо в процентах от ширины или высоты окна, либо знаком * для обозначения оставшейся части окна.

Упражнение. Построить фреймовую структуру согласно данному ниже коду (все файлы упражнения сохранить в папке **ФреймУпр**).

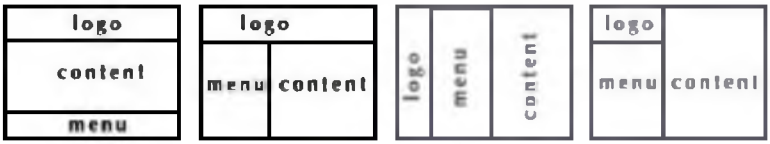
Рамочная часть документа (сохранить в файле **РамкаУпр.htm**). **<html>** **<head>** **<title>Рамка фрейма** **</title>** **</head>** **<frameset rows="50,*" border=4>** **<frame src="Заголовок.htm" name="top"** **<frameset cols="36%,*">** **<frame src="Содержание.htm" name="left">** **<frame src="Информация.htm" name="right">** **</frameset>** **</frameset>** **</html>**	*HTML-документ для верхнего фрейма* (сохранить в файле **Заголовок.htm**). **<html>** **<head>** **<title>Верхний фрейм** **</title>** **</head>** **<body>** **<h2 align=center>** **Заголовок** **</h2>** **</body>** **</html>**
HTNL-документ для левого фрейма (сохранить в файле **Содержание.htm)** **<html>**	*HTNL-документ для правого фрейма* (сохранить в файле **Информация.htm**)

<head> **<title>Левый фрейм** **</title>** **</head>** **<body>** **<h2 align=center>** **Содержание** **</h2>** **</body>** **</html>**	**<html>** **<head>** **<title>Правый фрейм </title>** **</head>** **<body>** **<h2 align=center>** **Основная информация** **</h2>** **</body>** **</html>**

Подробнее о понятии «фрейм».

Фреймы позволяют нам открыть в окне браузера - не один, а сразу несколько документов (допустим, документ menu.html, который содержит меню, logo.html - документ, который содержит логотип, шапку страницы, и content.html - документ с непосредственным содержанием нашего сайта). Итак, для того, чтобы браузер показал одновременно несколько документов, надо создать специальный фрейм-документ, в котором необходимо указать сколько документов откроется в одном окне браузера, сколько места будет занимать каждый, каким образом они будут располагаться. Т.к. первый документ на сайте, который показывается посетителю это index.html (или вроде него), то он будет фрейм-документом. Создадим документ index.htm html.

Предлагаются классический вариант - logo.html, menu.html, content.html. Расположить мы можем это по-разному, вот для примера четыре варианта из множества возможных:



Для того, чтобы браузер открыл сразу несколько документов, надо создать страничку в которой мы укажем сколько документов нам нужно показать, их размеры и расположение. Назовем ее index.html.

```
<html>
<head>
<title>Вступление. Понятие о фреймах.</title>
</head>
</html>
```

Тэга **<body> нет**! А на этой страничке он нам и не нужен. Вместо него нам **нужен (обязательно, если хотим фреймы) тэг <frameset></frameset>.** Он пишется между тэгами <head> и </head>.

```
<html>
<head>
<title>Вступление. Понятие о фреймах.</title>
<frameset></frameset>
</head>
</html>
```

Создаем простой фрейм.

Назовите файл index.html и напишите там следующий код:

```
<html>
<head>
<title>Создаем простой фрейм</title>
<frameset cols="100,*">
<frame name="menu" target="main" src="menu.html">
<frame name="main" src="main.html">
</frameset>
</head>
</html>
```

Атрибут **cols="размер"(rows="размер").** Cols - это колонки? т.е. фреймы будут расположены как колонки. Так же можно сделать фреймы в ряд. Для этого вместо cols надо написать **rows**. У нас документ делится на 2 части: 1 - меню, 2 - содержимое. Так вот, первое число 50 означает, что меню будет размером в 100 пикселей, а знак * означает, что содержимое будет занимать всю оставшуюся часть.

Атрибуты **name="main" и name="menu"** - это имена фреймов.

Что означает атрибут **target="main"?** Это значит, что, если пользователь кликнит по ссылке на странице menu.html, то ссылка откроется в содержимом.

Теперь о тэге **<frame>.** Этот тэг устанавливает какие документы в каком ряду у нас будут грузиться. Первый ряд - меню, второй - содержимое. Если что-то вас не устраивает, то можете поменять эти тэги местами и расположение изменится. Также у тэга <frame> есть атрибут **src (<frame src="страничка">).** Этот атрибут загружает указанную страничку во фрейм.

Итак, сначала создадим такой вариант:

```
<html>
<head>
<title>Хождение по фреймам</title>
<frameset rows="100,*,150">
<frame src="logo.html">
```

```
<frame src="content.html">  <frame src="menu.html">
</frameset>
</head>
</html>
```

Параметр rows - в нашем примере это выглядит так: rows="100,*,150" - а если перевести, то мы получим следующее - "... наш документ делиться на несколько рядов (строк). *Высота* первого ряда - 100 пикселов, третьего - 150, а второй занимает все оставшееся пространство". Тэг frame сообщает браузеру какие же документы у нас будут в каждом ряду (строке). В нашем случае: первый ряд - logo.html (документ с логотипом), второй ряд - займет документ с непосредственным содержанием (content.html), а третий - меню. Если вы хотите, чтобы меню было во втором ряду, то вам следует поменять его местами с content.html

```
<html>
<head>
<title>Хождение по фреймам</title>
<frameset rows="100,*,150">
<frame src="logo.html">
<frame src="menu.html">
<frame src="content.html">
</frameset>
</head>
</html>
```

Теперь надо задать новые значения параметру rows, чтобы меню у нас снова занимало только 150 пикселов по высоте, а содержание - все остальное:

```
<html>
<head>
<title>Хождение по фреймам</title>
<frameset rows="100,150,*">
<frame src="logo.html">
<frame src="menu.html">
<frame src="content.html">
</frameset>
</head>
</html>
```

Задание: замените параметр rows, на cols и получите следующую страничку:

Итак, мы можем делить окно нашего браузера либо на ряды, либо на колонки, с помошью параметров тэга <frameset> Cols и Rows. Другого способа деления не существует. Одновременно эти параметры использовать нельзя. При помощи rows мы разбиваем окно на ряды и задаем какую *высоту* будет иметь каждый ряд, при помощи cols разбиваем окно на колонки и задаем какую *ширину* будет иметь каждая колонка.

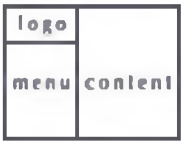
Кстати, ширина и высота могут задаваться не только в пикселях, но и в процентах от общей ширины (высоты) окна: `<frameset cols="10%,15%,75%">`. Помните, что в сумме это все должно равняться 100%.

Каким образом мы будем делить окно? - На ряды. В первом ряду у нас будет располагаться logo.html, а второй ряд мы поделим на две колонки, в которых будут располагаться документы menu.html и content.html.

Принцип построения ясен, только вот как обозначить ряд, разбитый на две колонки? Тут нам поможет тэг Frameset.

```
<html>
<head>
<title>Хождение по фреймам</title>
<frameset rows="100,*">
<frame src="logo.html">
<frameset cols="150,*">
<frame src="menu.html">
<frame src="content.html">
</frameset>
</frameset>
</head>
</html>
```

Первый ряд мы оформили как положено, при помощи тэга frame. Во втором ряду с помощью параметра cols тэга <frameset></frameset> мы делим второй ряд на две колонки (первая шириной 150 пикселов, вторая по ширине занимает все оставшееся пространство). А тэги <frame>, которые содержит <frameset></frameset> определяют какие документы будут показаны в колонках (menu.html и content.html).



Задание: изобразить еще один вариант странички.

Scrolling - параметр тэга <frame>. Он может принимать несколько значений: no - это значит совсем не будет полосы прокрутки, ни при каких обстоятельствах; yes - это значит полоса прокрутки будет всегда; auto - полоса прокрутки появиться только тогда, когда она нужна. Собственно, параметр scrolling="auto", можно не прописывать, т.к. если параметр scrolling не задан, то полоса прокрутки появиться, если она нужна, а если нет - ее не будет.

Теперь давайте избавимся от рамок между фреймами. Для этого мы используем параметр border, с которым мы раньше уже встречались. Итак, border="0", используется как атрибут тэга <frameset>.

Задание: применить данные атрибуты к одному из вариантов.

Рассмотрим пример: <frame src="logo.html" scrolling="no" **marginwidth="0" marginheight="0"**>

Marginheight определяет ширину (в пикселах) верхнего и нижнего полей фрэйма, а marginwidth определяет ширину левого и правого полей фрэйма. В нашем примере мы избавились от полей во фрейме, содержащим logo.html, задав значение marginheight и marginwidth равное нулю.

Это **marginwidth="число в пикселях" marginheight="число в пикселях".** Вставьте эти атрибуты в тэг <frame> и поизменяйте их значениея.

Надо сделать так, чтобы документ при нажимании на ссылку открылся во фрейме с основным содержанием, а меню осталось в нетронутом виде. Как это сделать? Для начала, познакомьтесь с новым параметром тэга <frame> - name.

Наберите и просмотрите пример:

```
<html>
<head>
<title>Хождение по фреймам</title>
<frameset cols="100,*" border="0">
<frameset rows="100,*">
<frame src="logo.html" scrolling="no" marginwidth="0"
marginheight="0">
<frame src="menu.html">
</frameset>
<frame src="content.html" name="window-1">
</frameset>
</head>
</html>
```

Параметр name задает имя для фрейма (в нашем случае для того, который содержит документ content.html). Имя фрейма может быть в дальнейшем использовано для ссылки на него из других документов (фреймов), с помощью параметра тэга <a> target (target="имя_фрейма").

В нашем случае window-1 - это имя фрейма, в котором у нас располагается документ с основным содержанием (content.html).

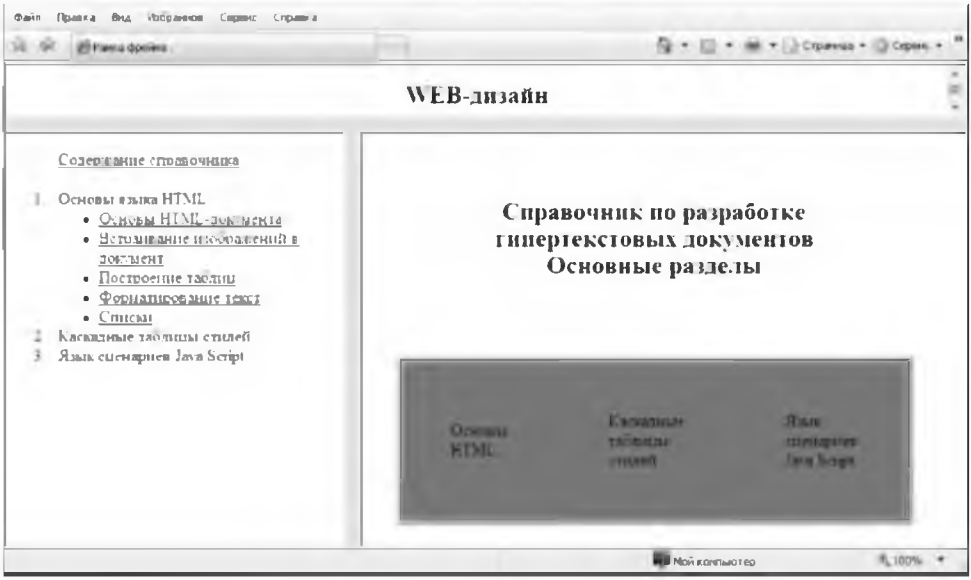
***Задание**: Создайте страничку, аналогичную данному рисунку. В левом фрейме в списке должны быть ссылки, при нажимании, на которые соответствующие файлы будут появляться в правом окне. Для этого необходимо первоначально создать отдельную папку. В ней должны находиться все используемые в этой страничке файлы*

Просмотрите и разберите следующие файлы-рекомендации для выполнения данного задания.

1. <html>
<head>
<title>Рамка фрейма </title>

```
</head>
        <frameset rows="50,*"border=14>


<frame src="Заголовок.htm" name="top">
<frameset cols="36%,*">
<frame src="Содержание.htm" name="left">
<frame src="Информация.htm" name="right">
<frameset>
</html>
```




```
2. <html>
<head>
<title>Левый фрейм</title>
</head>
<body>
<ul align=center><a href="contact_2.htm" target=right>
Содержание справочника</a>
</ul>
<ol type=1>
<li>Основы языка HTML
<ul type=disc>
<li>
```

```
<a href="news.htm"                    target=right>Основы HTML-
документа</a>

<li>
<a  href="products.htm"  target=right>Встраивание  изображений  в
документ</a>
<li>
<a href="servis.htm" target=right>Построение таблиц</a>
<li>
<a href="contact.htm" target=right>Форматирование текст</a>
<li>
<a href="contact_1.htm" target=right>Спиеки</a>
</ul>
<li>Каскадные таблицы стилей
<li>Язык сценариев Java Script
</ol>
</h2>
</body>
</html>
```

Бывают ситуации, когда нам нужно, чтобы открываемый документ открылся во все окно, для этого надо параметру target задать значение _top:

```
<a href="project.html" target="_top">Мой проект
о рыбках</a>
```

И последнее о фреймах - IFrame - "плавающий" фрейм (поддерживаются только IE). Итак, IFrame - "плавающий" или встроенный фрейм. Пользователи IE, наверное, много раз встречали его на различных сайтах.