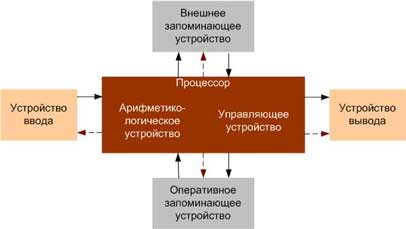
**Вопрос №6**

**1. Архитектура ЭВМ.**

**Архитектура ЭВМ**– это наиболее общие принципы построения ЭВМ, реализующие программное управление работой и взаимодействием основных ее функциональных узлов.  
  
Под архитектурой ЭВМ принято понимать совокупность общих принципов организации аппаратно-программных средств и основных их характеристик, определяющая функциональные возможности ЭВМ при решении соответствующих типов задач. Архитектура ЭВМ включает в себя как структуру, отражающую состав ПК, так и программно – математическое обеспечение. Структура ЭВМ - совокупность элементов и связей между ними. Основным принципом построения всех современных ЭВМ является программное управление.

**Классическая архитектура ЭВМ.**

Основы учения об архитектуре вычислительных машин были заложены [**Джон фон Нейманом**](http://www.calend.ru/person/5725/). Совокупность этих принципов породила классическую (фон-неймановскую) архитектуру ЭВМ.   
  
Фон Нейман не только выдвинул основополагающие принципы логического устройства ЭВМ, но и предложил ее структуру, представленную на рисунке:

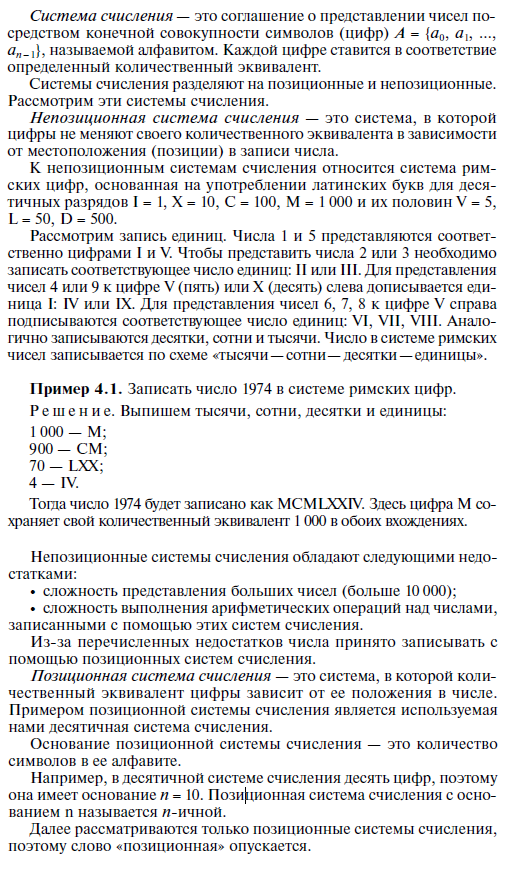
  
  
**Положения фон Неймана:**

* Компьютер состоит из нескольких основных устройств (арифметико-логическое устройство, управляющее устройство, память, внешняя память, устройства ввода и вывода)
* Арифметико-логическое устройство – выполняет логические и арифметические действия, необходимые для переработки информации, хранящейся в памяти
* Управляющее устройство – обеспечивает управление и контроль всех устройств компьютера (управляющие сигналы указаны пунктирными стрелками)
* Данные, которые хранятся в запоминающем устройстве, представлены в двоичной форме
* Программа, которая задает работу компьютера, и данные хранятся в одном и том же запоминающем устройстве
* Для ввода и вывода информации используются устройства ввода и вывода
* Один из важнейших принципов – принцип хранимой программы – требует, чтобы программа закладывалась в память машины так же, как в нее закладывается исходная информация.

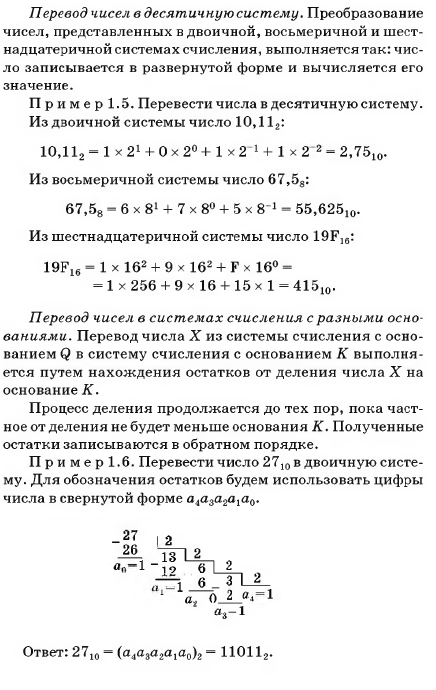
**Арифметико-логическое устройство** и **устройство управления** в современных компьютерах образуют процессор ЭВМ. Процессор, который состоит из одной или нескольких больших интегральных схем называется микропроцессором или микропроцессорным комплектом.   
  
**Процессор** – функциональная часть ЭВМ, выполняющая основные операции по обработке данных и управлению работой других блоков. Процессор является преобразователем информации, поступающей из памяти и внешних устройств.   
  
**Запоминающие устройства** обеспечивают хранение исходных и промежуточных данных, результатов вычислений, а также программ. Они включают: оперативные (ОЗУ), сверхоперативные СОЗУ), постоянные (ПЗУ) и внешние (ВЗУ) запоминающие устройства.   
  
**Оперативные ЗУ** хранят информацию, с которой компьютер работает непосредственно в данное время (резидентная часть операционной системы, прикладная программа, обрабатываемые данные). В СОЗУ хранится наиболее часто используемые процессором данные. Только та информация, которая хранится в СОЗУ и ОЗУ, непосредственно доступна процессору.   
  
**Внешние запоминающие устройства** (накопители на магнитных дисках, например, жесткий диск или винчестер) с емкостью намного больше, чем ОЗУ, но с существенно более медленным доступом, используются для длительного хранения больших объемов информации. Например, операционная система (ОС) хранится на жестком диске, но при запуске компьютера резидентная часть ОС загружается в ОЗУ и находится там до завершения сеанса работы ПК.   
  
**ПЗУ** (постоянные запоминающие устройства) и **ППЗУ** (перепрограммируемые постоянные запоминающие устройства) предназначены для постоянного хранения информации, которая записывается туда при ее изготовлении, например, ППЗУ для BIOS.   
  
В качестве **устройства ввода информации** служит, например, клавиатура. В качестве устройства вывода – дисплей, принтер и т.д.  
  
В построенной по схеме фон Неймана ЭВМ происходит последовательное считывание команд из памяти и их выполнение. Номер (адрес) очередной ячейки памяти, из которой будет извлечена следующая команда программы, указывается специальным устройством – счетчиком команд в устройстве управления.

**2. Арифметические основы ЭВМ (системы счисления, перевод из одной системы счисления в другую, арифметические операции, коды прямой, дополнительный и обратный).**

(книга №1)

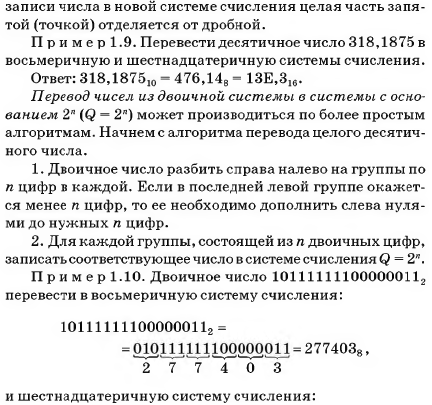
****

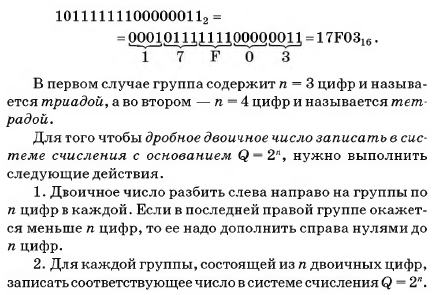
(книга №6)

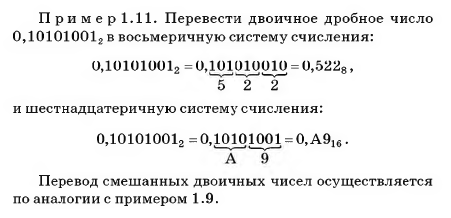
****

****

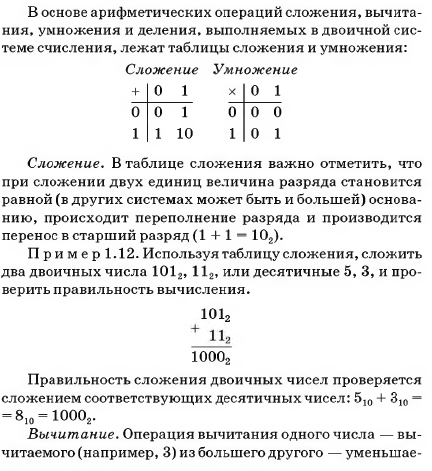
****

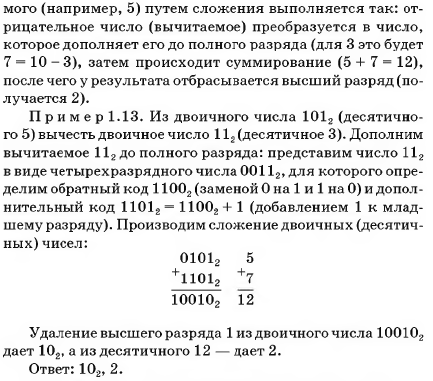


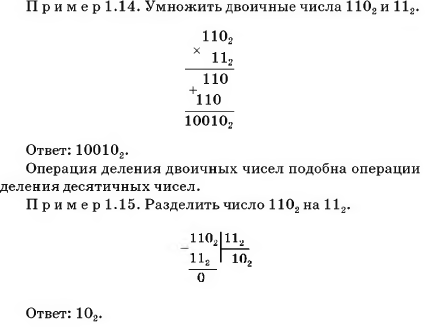




(книга №6)









**12 Представление двоичных чисел со знаком. Прямой дополнительный и обратный коды. Выполнение операций в обратном и дополнительном кодах**

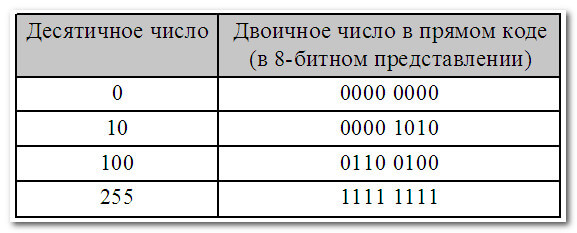
(интернет)

**Прямой код**

Прямой код — способ представления двоичных чисел с фиксированной запятой. Главным образом используется для записи неотрицательных чисел

*Используется в двух вариантах:*

1) Для записи только неотрицательных чисел

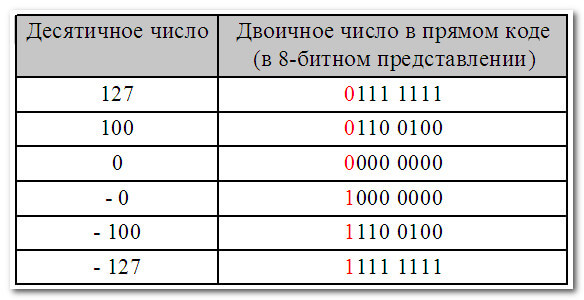


В этом варианте (для восьмибитного двоичного числа) мы можем записать максимальное число 255 (всего чисел 256 — от 0 до 255)

2) Для записи как положительных, так и отрицательных чисел  
 В этом случае старший бит (в нашем случае — восьмой) объявляется знаковым разрядом (знаковым битом).  
При этом, если:  
— знаковый разряд равен 0, то число положительное  
— знаковый разряд равен 1, то число отрицательное



В этом случае диапазон десятичных чисел, которые можно записать в прямом коде составляет от — 127 до +127:



Использование прямого кода для представления отрицательных чисел является неэффективным — очень сложно реализовать арифметические операции и, кроме того, в прямом коде два представления нуля — положительный ноль и отрицательный ноль (чего не бывает).

**Обратный код**

*Обратный код* — метод вычислительной математики, позволяющий вычесть одно число из другого, используя только операцию сложения.

*Обратный двоичный код положительного числа* состоит из одноразрядного кода знака (битового знака) — двоичной цифры 0, за которым следует значение числа.  
 *Обратный двоичный код отрицательного числа* состоит из одноразрядного кода знака (битового знака) — двоичной цифры 1, за которым следует инвертированное значение положительного числа.

**Для неотрицательных чисел обратный код** двоичного числа имеет тот же вид, что и запись неотрицательного числа в прямом коде.  
 **Для отрицательных чисел обратный код** получается из неотрицательного числа в прямом коде, путем инвертирования всех битов (1 меняем на 0, а 0 меняем на 1).  
 Для преобразования отрицательного числа записанное в обратном коде в положительное достаточного его проинвертировать.

При 8-битном двоичном числе — знаковый бит (как и в прямом коде) старший (8-й)



Диапазон десятичных чисел, который можно записать в обратном коде от -127 до + 127

***Арифметические операции с отрицательными числами в обратном коде:***

**1-й пример (для положительного результата)**  
Дано два числа:  
100 = 0110 0100  
-25 = — 0001 1001  
Необходимо их сложить:  
100 + (-25) = 100 — 25 = 75

**1-й этап**  
Переводим число -25 в двоичное число в обратном коде:  
25 = 0001 1001  
-25= 1110 0110  
и складываем два числа:  
0110 0100 (100) + 1110 0110 (-25) = 1 0100 1010, отбрасываем старшую 1 (у нас получился лишний 9-й разряд — переполнение), = 0100 1010  
**2-й этап**  
Отброшенную в результате старшую единицу прибавляем к результату:  
0100 1010 + 1 = 0100 1011 (знаковый бит =0, значит число положительное), что равно 75 в десятичной системе

**2-й пример (для отрицательного результата)**  
Дано два числа:  
5 = 0000 0101  
-10 = — 0000 1010  
Необходимо их сложить:  
5 + (-10) = 5 — 10 = -5

1-й этап  
Переводим число -10 в двоичное число в обратном коде:  
10 = 0000 1010  
-10= 1111 0101  
и складываем два числа:  
0000 0101 (5) + 1111 0101 (-10) = 1111 1010 (знаковый бит =1, значит число отрицательное)

2-й этап  
Раз результат получился отрицательный, значит число представлено в обратном коде.  
Переводим результат в прямой код (путем инвертирования значения, знаковый бит не трогаем):  
1111 1010 —-> 1000 0101  
Проверяем:  
1000 0101 = — 0000 0101 = -5

***Обратный код****решает проблему сложения и вычитания чисел с различными знаками, но и имеет свои недостатки:*  
 *— арифметические операции проводятся в два этапа*  
 *— как и в прямом коде два представления нуля — положительный и отрицательный*

**Дополнительный код**

*Дополнительный код* — наиболее распространенный способ представления отрицательных чисел. Он позволяет заменить операцию вычитания на операцию сложения и сделать операции сложения и вычитания одинаковыми для знаковых и беззнаковых чисел.

В дополнительном коде (как и в прямом и обратном) старший разряд отводится для представления знака числа (знаковый бит).

Диапазон десятичных чисел которые можно записать в дополнительном коде от -128 до +127. Запись положительных двоичных чисел в дополнительном коде та-же, что и в прямом и обратном кодах.



**Дополнительный код** отрицательного числа можно получить двумя способами  
*1-й способ:*  
— инвертируем значение отрицательного числа, записанного в прямом коде (знаковый бит не трогаем)  
— к полученной инверсии прибавляем 1  
**Пример:**Дано десятичное число -10  
Переводим в прямой код:  
10 = 0000 1010 —-> -10 = 1000 1010  
Инвертируем значение (получаем обратный код):  
1000 1010 —-> 1111 0101  
К полученной инверсии прибавляем 1:  
1111 0101 + 1 = 1111 0110 — десятичное число -10 в дополнительном коде

*2-й способ:*  
Вычитание числа из нуля  
Дано десятичное число 10, необходимо получить отрицательное число (-10) в дополнительном двоичном коде  
Переводим 10 в двоичное число:  
10 = 0000 1010  
Вычитаем из нуля:  
0 — 0000 1010 = 1111 0110 — десятичное число -10 в дополнительном коде



**Арифметические операции с отрицательными числами в дополнительном коде:**

Дано: необходимо сложить два числа -10 и 5  
-10 + 5 = -5  
Решение:  
5 = 0000 0101  
-10 = 1111 0110 (в дополнительном коде)  
Складываем:  
1111 0110 + 0000 0101 = 1111 1011, что соответствует числу -5 в дополнительном коде

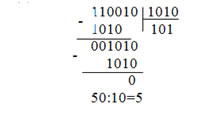
Как мы видим на этом примере — дополнительный код отрицательного двоичного числа наиболее подходит для выполнения арифметических операций сложения и вычитания отрицательных чисел.

**13 Деление двоичных чисел**

Деление в двоичной системе производится вычитанием делителя со сдвигом вправо, если остаток больше нуля.

     **Пример**

Найти частное  двух чисел если:  
1. Делимое больше делителя:



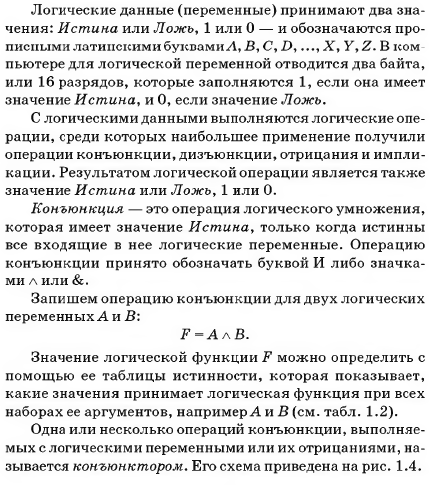
      2. Делимое меньше делителя:

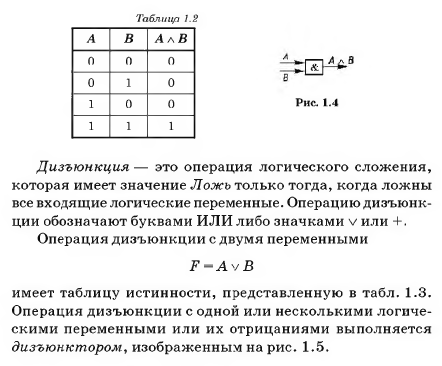


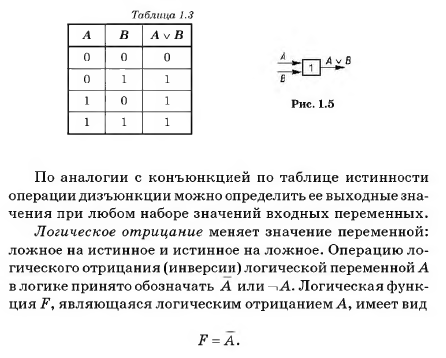
     Как видно из приведенных примеров, операция деления может быть представлена как операции сравнения, сдвига и суммирования.

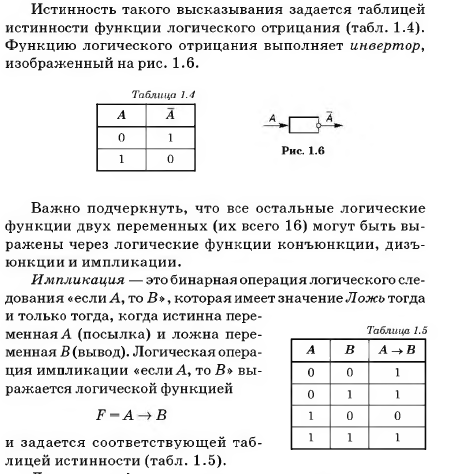
**3. Алгебра логики. Логические функции. Преобразования логических выражений.**

(книга №6)

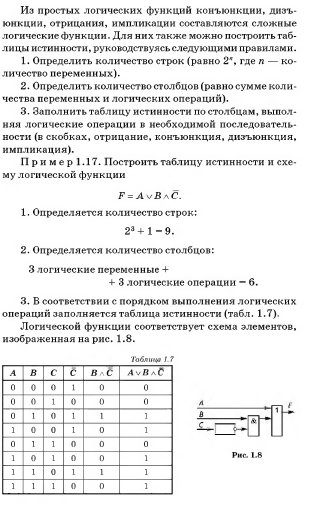




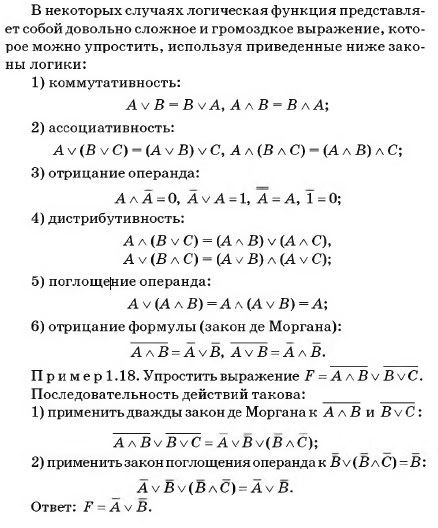






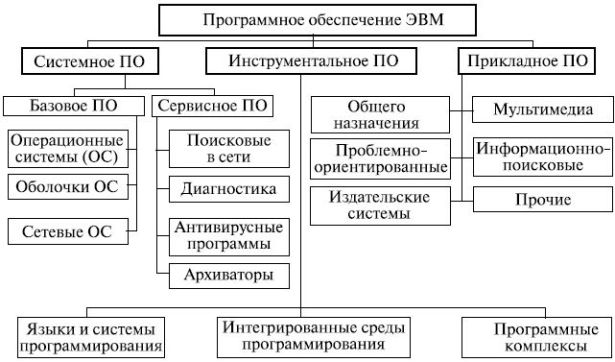


(книга №6)



**4. Программное обеспечение ЭВМ (прикладное и системное).**





Программное обеспечение, можно условно разделить на три категории:

1. системное ПО (программы общего пользования), выполняющие различные вспомогательные функции, например создание копий используемой информации, выдачу справочной информации о компьютере, проверку работоспособности устройств компьютера и т.д.
2. прикладное ПО, обеспечивающее выполнение необходимых работ на ПК: редактирование текстовых документов, создание рисунков или картинок, обработка информационных массивов и т.д.
3. инструментальное ПО (системы программирования), обеспечивающее разработку  новых программ для компьютера на языке программирования.

#### Системное ПО

Это программы общего пользования не связаны с конкретным применением ПК и выполняют традиционные функции: планирование и управление задачами, управления вводом-выводом и т.д.

Другими словами, системные программы выполняют различные вспомогательные функции, например, создание копий используемой информации, выдачу справочной информации о компьютере, проверку работоспособности устройств компьютера и т.п.

К системному ПО относятся:

* операционные системы (эта программа загружается в ОЗУ при включении компьютера);
* программы – оболочки (обеспечивают более удобный и наглядный способ общения с компьютером, чем с помощью командной строки DOS, например, Norton Commander);
* операционные оболочки – интерфейсные системы, которые используются для создания графических интерфейсов, мультипрограммирования и.т.;
* Драйверы (программы, предназначенные для управления портами периферийных устройств, обычно загружаются в оперативную память при запуске компьютера);
* утилиты (вспомогательные или служебные программы, которые представляют пользователю ряд дополнительных услуг).

К утилитам относятся:

* диспетчеры файлов или файловые менеджеры;
* средства динамического сжатия данных (позволяют увеличить количество информации на диске за счет ее динамического сжатия);
* средства просмотра и воспроизведения;
* средства диагностики; средства контроля позволяют проверить конфигурацию компьютера и проверить работоспособность устройств компьютера, прежде всего жестких дисков;
* средства коммуникаций (коммуникационные программы) предназначены для организации обмена информацией между компьютерами;
* средства обеспечения компьютерной безопасности (резервное копирование, антивирусное ПО).

Необходимо отметить, что часть утилит входит в состав операционной системы, а другая часть функционирует автономно. Большая часть общего (системного) ПО входит в состав ОС. Часть общего ПО входит в состав самого компьютера (часть программ ОС и контролирующих тестов записана в ПЗУ или ППЗУ, установленных на системной плате). Часть общего ПО относится к автономными программам и поставляется отдельно.

#### Прикладное ПО

Прикладные программы могут использоваться  автономно или в составе программных комплексов или пакетов.

Прикладное ПО – программы,  непосредственно обеспечивающие выполнение необходимых работ на ПК: редактирование текстовых документов, создание рисунков или картинок, создание электронных таблиц и т.д.

Пакеты прикладных программ – это система программ, которые по сфере применения делятся на проблемно – ориентированные, пакеты общего назначения и интегрированные пакеты. Современные интегрированные пакеты содержат до пяти функциональных компонентов: тестовый и табличный процессор, СУБД, графический редактор, телекоммуникационные средства.

К прикладному ПО, например, относятся:

1. Комплект офисных приложений MS OFFICE.
2. Бухгалтерские системы.
3. Финансовые аналитические системы.
4. Интегрированные пакеты делопроизводства.
5. CAD – системы (системы автоматизированного проектирования).
6. Редакторы HTML или Web – редакторы.
7. Браузеры – средства просмотра Web - страниц.
8. Графические редакторы.
9. Экспертные системы.

И так далее.

#### Инструментальное ПО

Инструментальное ПО или системы программирования  - это системы для автоматизации разработки новых программ на языке программирования.

В самом общем случае для создания программы на выбранном языке программирования (языке системного программирования) нужно иметь следующие компоненты:

1. Текстовый редактор для создания файла с исходным текстом программы.
2. Компилятор или интерпретатор. Исходный текст с помощью программы-компилятора переводится в промежуточный объектный код. Исходный текст большой программы состоит из нескольких модулей(файлов с исходными текстами). Каждый модуль компилируется в отдельный файл с объектным кодом, которые затем надо объединить в одно целое.
3. Редактор связей или сборщик, который выполняет связывание объектных модулей и формирует на выходе работоспособное приложение – исполнимый код. Исполнимый код – это законченная программа, которую можно запустить на любом компьютере, где установлена операционная система, для которой эта программа создавалась. Как правило, итоговый файл имеет расширение .ЕХЕ или .СОМ.
4. В последнее время получили распространение визуальный методы программирования (с помощью языков описания сценариев), ориентированные на создание Windows-приложений. Этот процесс автоматизирован в средах быстрого проектирования. При этом используются готовые визуальные компоненты, которые настраиваются с помощью специальных редакторов.

Наиболее популярные редакторы (системы программирования программ с использованием визуальных средств) визуального проектирования:

1. Borland Delphi - предназначен для решения практически любых задачи прикладного программирования.
2. Borland C++ Builder – это отличное средство для разработки DOS и Windows приложений.
3. Microsoft Visual Basic – это популярный инструмент для создания Windows-программ.
4. Microsoft Visual C++ - это средство позволяет разрабатывать любые приложения, выполняющиеся в среде ОС типа Microsoft Windows.

**5. Операционные системы, их функции, классификация, типы.**

Программы, составляющие ПО, можно разделить на три группы: системное ПО, системы программирования, прикладное ПО. Ядром системного ПО является операционная система (ОС).

ОС - это неотъемлемая часть ПО, управляющая техническими средствами компьютера (hardware).. Операционная система - это программа, координирующая действия вычислительной машины; под ее управлением осуществляется выполнение программ.

Основные функции операционной системы:

* 1.      Обмен данными между компьютером и различными периферийными устройствами (терминалами, принтерами, гибкими дисками, жесткими дисками и т.д.). Такой обмен данными называется "ввод/вывод данных".
* 2.      Обеспечение системы организации и хранения файлов.
* 3.      Загрузка программ в память и обеспечение их выполнения.
* 4.      Организация диалога с пользователем.

**По количеству одновременно работающих пользователей:**

* **Однопользовательские ОС** позволяют работать на компьютере только одному человеку.
* **Многопользовательские ОС** поддерживают одновременную работу на ЭМВ нескольких пользователей за различными терминалами.

**По числу процессов, одновременно выполняемых под управлением системы:**

* **Однозадачные ОС** поддерживают выполнение только одной программы в отдельный момент времени, то есть позволяют запустить одну программу в основном режиме.
* **Многозадачные ОС** (мультизадачные) поддерживают параллельное выполнение нескольких программ, существующих в рамках одной вычислительной системы на некотором отрезке времени, то есть позволяют запустить одновременно несколько программ, которые будут работать параллельно, не мешая друг другу.

Операционные системы классифицируются по:

* количеству одновременно работающих пользователей: однопользовательские, многопользовательские;
* числу процессов, одновременно выполняемых под управлением системы;
* количеству решаемых задач: однозадачные, многозадачные;
* количеству поддерживаемых процессоров: однопроцессорные, многопроцессорные;
* разрядности кода ОС: 8-разрядные, 16-разрядные, 32-разрядные, 64-разрядные;
* типу интерфейса: командные (текстовые) и объектно-ориентированные (графические);
* типу доступа пользователя к ЭВМ: с пакетной обработкой, с разделением времени, реального времени;
* типу использования ресурсов: сетевые, локальные.

**6. Трансляторы, компиляторы и интерпретаторы.**

***Транслятор*** (англ. translator — переводчик) — это программа-переводчик. Она преобразует программу, написанную на одном из языков высокого уровня, в программу, состоящую из машинных команд. Транслятор обычно выполняет также диагностику ошибок, формирует словари идентификаторов, выдаёт для печати тексты программы и т. д. Язык, на котором представлена входная программа, называется исходным языком, а сама программа — исходным кодом. Выходной язык называется целевым языком или объектным кодом.

В общем случае понятие трансляции относится не только к языкам программирования, но и к другим языкам — как формальным компьютерным (вроде языков разметки типа HTML), так и естественным (русскому, английскому и т. п.).

## Виды трансляторов

1. Диалоговый. Обеспечивает использование языка программирования в режиме разделения времени (англ.).
2. Синтаксически-ориентированный (синтаксически-управляемый). Получает на вход описание синтаксиса и семантики языка и текст на описанном языке, который и транслируется в соответствии с заданным описанием.
3. Однопроходной. Формирует объектный модуль за один последовательный просмотр исходной программы.
4. Многопроходной. Формирует объектный модуль за несколько просмотров исходной программы.
5. Оптимизирующий. Выполняет оптимизацию кода в создаваемом объектном модуле.
6. Тестовый. Набор макрокоманд языка ассемблера, позволяющих задавать различные отладочные процедуры в программах, составленных на языке ассемблера.
7. Обратный. Для программы в машинном коде выдаёт эквивалентную программу на каком-либо языке программирования (см.: дизассемблер, декомпилятор).

Трансляторы реализуются в виде компиляторов или интерпретаторов. С точки зрения выполнения работы компилятор и интерпретатор существенно различаются.

***Компилятор*** (англ. compiler — составитель, собиратель) читает всю программу целиком, делает ее перевод и создает законченный вариант программы на машинном языке, который затем и выполняется. Входной информацией для компилятора (исходный код) является описание алгоритма или программа на проблемно-ориентированном языке, а на выходе компилятора — эквивалентное описание алгоритма на машинно-ориентированном языке (объектный код).

## Виды компиляторов

* Векторизующий. Транслирует исходный код в машинный код компьютеров, оснащённых векторным процессором.
* Гибкий. Сконструирован по модульному принципу, управляется таблицами и запрограммирован на языке высокого уровня или реализован с помощью компилятора компиляторов.
* Диалоговый. См.: диалоговый транслятор.
* Инкрементальный. Повторно транслирует фрагменты программы и дополнения к ней без перекомпиляции всей программы.
* Интерпретирующий (пошаговый). Последовательно выполняет независимую компиляцию каждого отдельного оператора (команды) исходной программы.
* Компилятор компиляторов. Транслятор, воспринимающий формальное описание языка программирования и генерирующий компилятор для этого языка.
* Отладочный. Устраняет отдельные виды синтаксических ошибок.
* Резидентный. Постоянно находится в оперативной памяти и доступен для повторного использования многими задачами.
* Самокомпилируемый. Написан на том же языке, с которого осуществляется трансляция.
* Универсальный. Основан на формальном описании синтаксиса и семантики входного языка. Составными частями такого компилятора являются: ядро, синтаксический и семантический загрузчики.

# Виды компиляции

1. Пакетная. Компиляция нескольких исходных модулей в одном пункте задания.
2. Построчная. То же, что и интерпретация.
3. Условная. Компиляция, при которой транслируемый текст зависит от условий, заданных в исходной программе директивами компилятора. Так, в зависимости от значения некоторой константы, можно включать или выключать трансляцию части текста программы.

## Структура компилятора

* 1. Процесс компиляции состоит из следующих этапов:
  2. Лексический анализ. На этом этапе последовательность символов исходного файла преобразуется в последовательность лексем (лексема-слово как абстрактная единица естественного языка).
  3. Синтаксический (грамматический) анализ. Последовательность лексем преобразуется в дерево разбора.
  4. Семантический анализ. Дерево разбора обрабатывается с целью установления его семантики (смысла) — например, привязка идентификаторов к их декларациям, типам, проверка совместимости, определение типов выражений и т. д. Результат обычно называется «промежуточным представлением/кодом», и может быть дополненным деревом разбора, новым деревом, абстрактным набором команд или чем-то ещё, удобным для дальнейшей обработки.
  5. Оптимизация. Выполняется удаление излишних конструкций и упрощение кода с сохранением его смысла. Оптимизация может быть на разных уровнях и этапах — например, над промежуточным кодом или над конечным машинным кодом.
  6. Генерация кода. Из промежуточного представления порождается код на целевом языке.

## Декомпиляция

Существуют программы, которые решают обратную задачу — перевод программы с низкоуровневого языка на высокоуровневый. Этот процесс называют декомпиляцией, а такие программы — декомпиляторами. Но поскольку компиляция — это процесс с потерями, точно восстановить исходный код, скажем, на C++, в общем случае невозможно.

***Интерпретатор***(англ. interpreter — истолкователь, устный переводчик) переводит и выполняет программу строка за строкой, производит пооператорный (покомандный, построчный) анализ, обрабатывает и тут же выполняет исходную программу или запроса (в отличие от компиляции, при которой программа транслируется без её выполнения.

***Интерпретатор* (англ. *interpreter* — истолкователь, устный переводчик) переводит и выполняет программу *строка за строкой*.**

**В отличие от компилятора, интерпретатор не порождает на выходе программу на машинном языке. Распознав команду исходного языка, он тут же выполняет ее.**

После того, как программа откомпилирована, ни сама исходная программа, ни компилятор более не нужны. В то же время программа, обрабатываемая интерпретатором, должна заново переводиться на машинный язык при каждом очередном запуске программы.

Откомпилированные программы работают быстрее, но интерпретируемые проще исправлять и изменять.

## Типы интерпретаторов

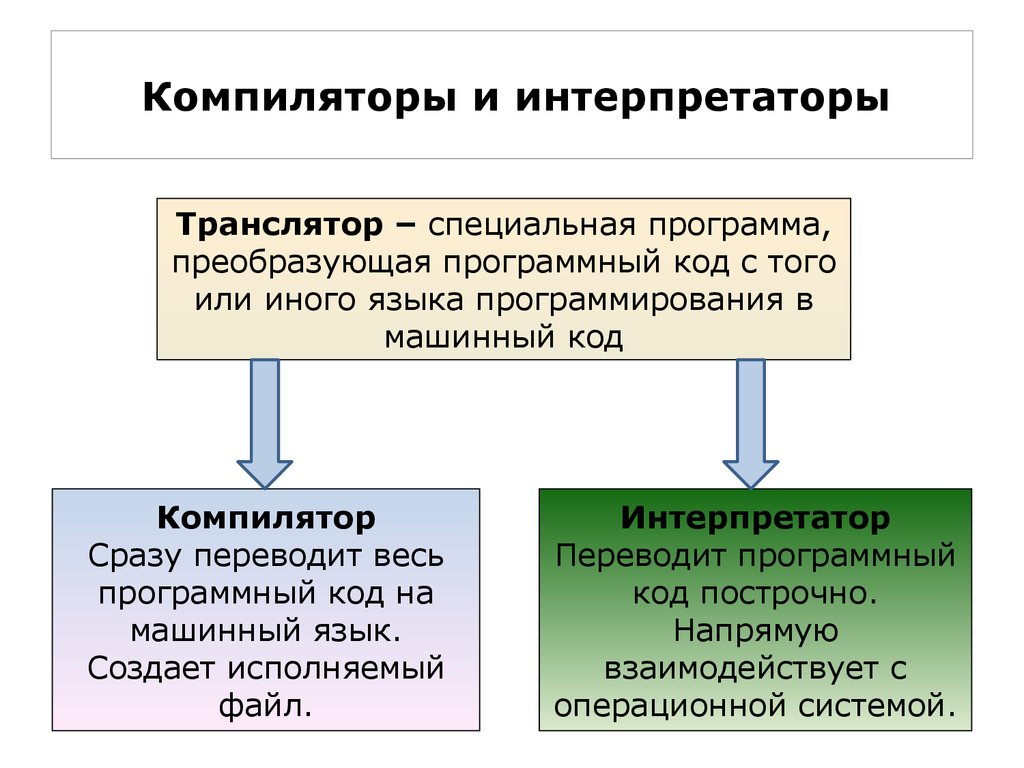
* Простой интерпретатор анализирует и тут же выполняет (собственно интерпретация) программу покомандно (или построчно), по мере поступления её исходного кода на вход интерпретатора. Достоинством такого подхода является мгновенная реакция. Недостаток — такой интерпретатор обнаруживает ошибки в тексте программы только при попытке выполнения команды (или строки) с ошибкой.
* Интерпретатор компилирующего типа — это система из компилятора, переводящего исходный код программы в промежуточное представление, например, в байт-код или p-код, и собственно интерпретатора, который выполняет полученный промежуточный код (так называемая виртуальная машина). Достоинством таких систем является большее быстродействие выполнения программ (за счёт выноса анализа исходного кода в отдельный, разовый проход, и минимизации этого анализа в интерпретаторе). Недостатки — большее требование к ресурсам и требование на корректность исходного кода. Применяется в таких языках, какJava,Tcl,Perl(используется байт-код), а также в различных СУБД.

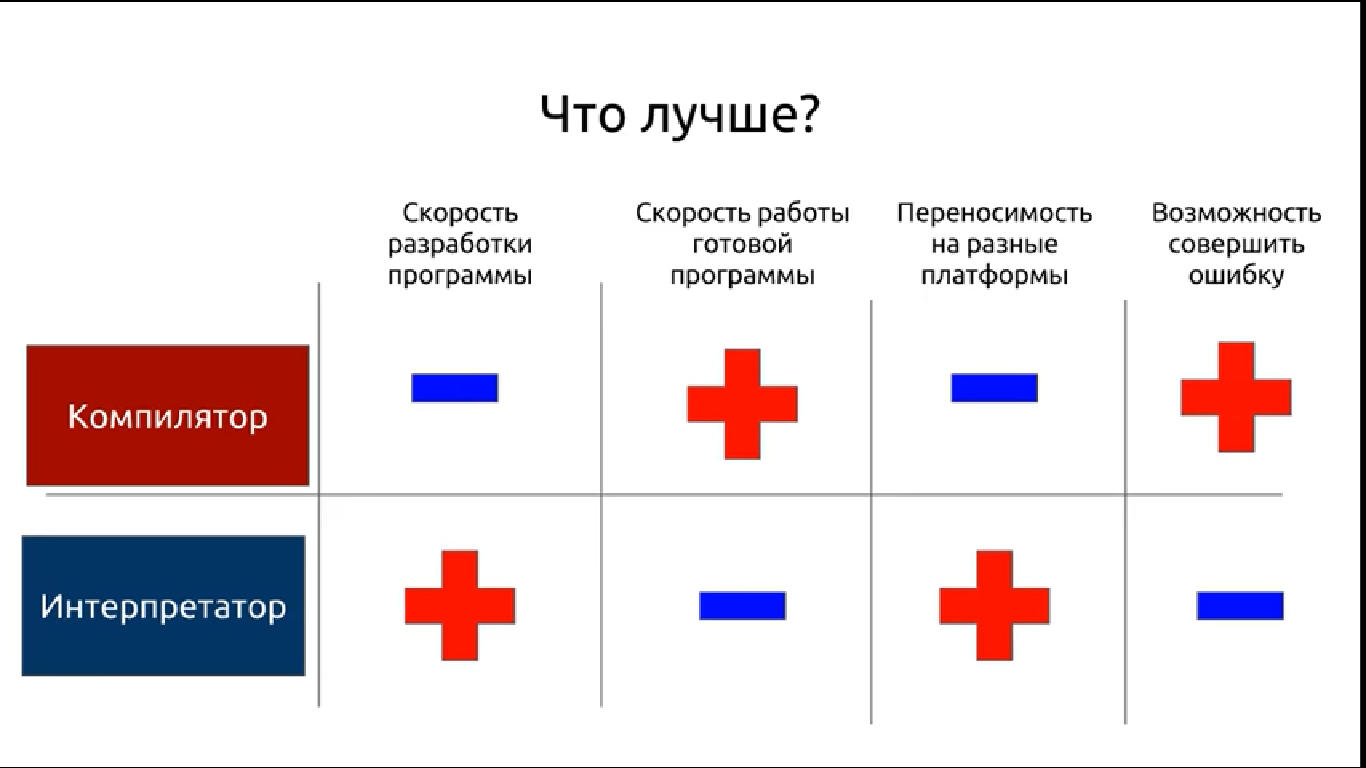
В случае разделения интерпретатора компилирующего типа на компоненты получаются компилятор языка и простой интерпретатор с минимизированным анализом исходного кода. Причём исходный код для такого интерпретатора не обязательно должен иметь текстовый формат или быть байт-кодом, который понимает только данный интерпретатор, это может быть машинный код какой-то существующей аппаратной платформы.

Некоторые интерпретаторы (например, для языков Лисп, Scheme,Python, Бейсик и других) могут работать в режиме диалога или так называемого цикла чтения-вычисления-печати. В таком режиме интерпретатор считывает законченную конструкцию языка, выполняет её, печатает результаты, после чего переходит к ожиданию ввода пользователем следующей конструкции.

Уникальным является язык Forth, который способен работать как в режиме интерпретации, так и компиляции входных данных, позволяя переключаться между этими режимами в произвольный момент, как во время трансляции исходного кода, так и во время работы программ.

Следует также отметить, что режимы интерпретации можно найти не только в программном, но и аппаратном обеспечении. Так, многие микропроцессоры интерпретируют машинный код с помощью встроенных микропрограмм, а процессоры семейства x86, начиная сPentium, во время исполнения машинного кода предварительно транслируют его во внутренний формат (в последовательность микроопераций).











**7. Виды сетей ЭВМ. Архитектура сетей.**

**Сети ЭВМ. Основные понятия. Классификация.**

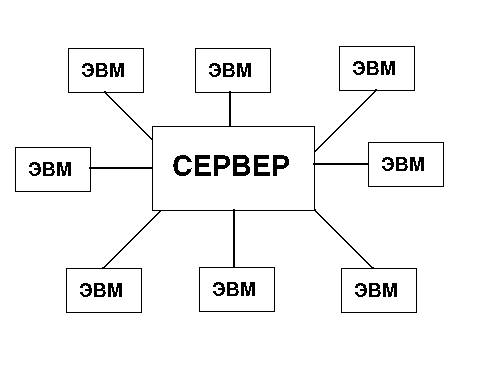
   Под **сетью ЭВМ**понимают соединение двух и более ЭВМ с целью совместного использования их ресурсов (процессоров, устройств памяти, устройств ввода/вывода, данных). По степени охвата территории различают сети:

* **локальные** (местные) - в пределах одного учреждения, помещения (или при максимальном удалении ЭВМ не более 1км.)
* **региональные** - внутри населенного пункта, района
* **национальные** - внутри государства
* **глобальные**

По степени доступности различают **корпоративные** и **общедоступные** сети.

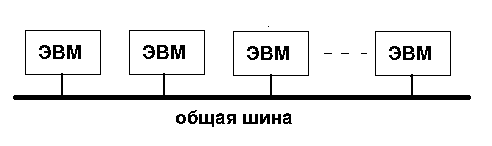
По топологии (способу объединения ЭВМ) различают:

* звездообразную топологию



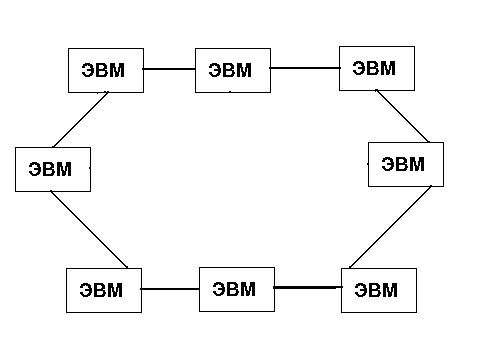
При таком способе обмен данными между ЭВМ осуществляется через более мощную ЭВМ - сервер. Недостатком такого соединения является низкая живучесть сети - выход из строя сервера означает прекращение функционирование сети. Однако, простота и дешевизна реализации сделала эту структуру популярной в локальных сетях.

* топологию с общей шиной



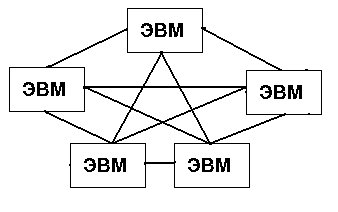
При этом способе обмен данными происходит через общую шину, которую используя механизм прерывания может "захватывать" тот или иной компьютер. Характерной особенностью здесь является отсутствие сервера. Очень часто используется в локальных сетях, а уж в "домашних" повсеместно.

* кольцевая топология



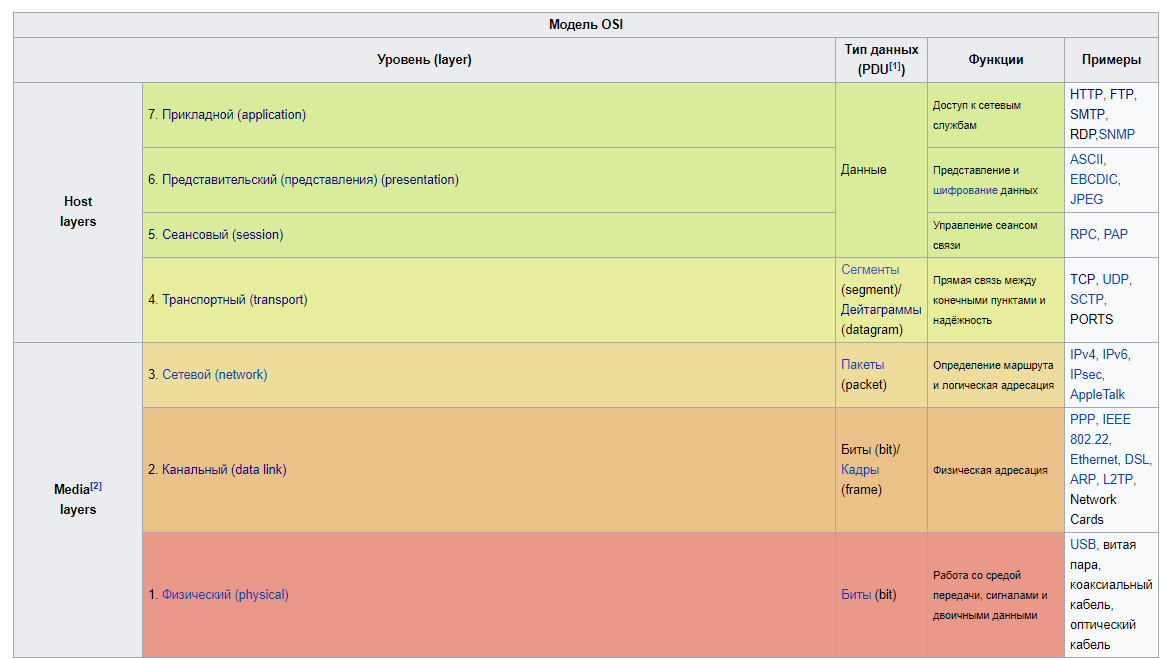
В этой структуре каждая ЭВМ используя механизм прерывания работает в качестве ретранслятора. Обратите внимание, живучесть сети повышена - при одиночном  обрыве связи между соседними ЭВМ сеть продолжает функционировать.

* полная топология



Соединение ЭВМ "каждая с каждой" позволяют получить сеть самую дорогую, но и обладающую максимальной живучестью.

**8. Понятие о стандарте OSI. Уровни OSI и их назначение.**



**Протокол прикладного уровня** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Application layer*) — протокол верхнего (7-го) уровня [сетевой модели OSI](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI), обеспечивает взаимодействие сети и пользователя. Уровень разрешает приложениям пользователя иметь доступ к сетевым службам, таким, как обработчик запросов к базам данных, доступ к файлам, пересылке электронной почты.

**Уровень представления** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Presentation layer*) — шестой уровень [сетевой модели OSI](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI).

Этот уровень отвечает за преобразование протоколов и кодирование/декодирование данных. Запросы приложений, полученные с уровня приложений, он преобразует в формат для передачи по сети, а полученные из сети данные преобразует в формат, понятный приложениям. На этом важном уровне может осуществляться сжатие/распаковка или кодирование/декодирование данных, а также перенаправление запросов другому сетевому ресурсу, если они не могут быть обработаны локально.

**Сеансовый уровень** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Session layer*) — 5-й уровень [сетевой модели OSI](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI), отвечает за поддержание [сеанса связи](https://ru.wikipedia.org/w/index.php?title=%D0%A1%D0%B5%D0%B0%D0%BD%D1%81_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)&action=edit&redlink=1)[[en]](https://en.wikipedia.org/wiki/Session_(computer_science)), позволяя приложениям взаимодействовать между собой длительное время. Уровень управляет созданием/завершением сеанса, обменом информацией, синхронизацией задач, определением права на передачу данных и поддержанием сеанса в периоды неактивности приложений. Синхронизация передачи обеспечивается помещением в поток данных контрольных точек, начиная с которых возобновляется процесс при нарушении взаимодействия.

**Транспортный уровень** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Transport layer*) — 4-й уровень [сетевой модели OSI](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI), предназначен для доставки данных. При этом не важно, какие данные передаются, откуда и куда, то есть, он предоставляет сам механизм передачи. Блоки данных он разделяет на фрагменты, размеры которых зависят от протокола: короткие объединяет в один, а длинные разбивает. Протоколы этого уровня предназначены для взаимодействия типа точка-точка. Пример: [TCP](https://ru.wikipedia.org/wiki/TCP), [UDP](https://ru.wikipedia.org/wiki/UDP), [SCTP](https://ru.wikipedia.org/wiki/SCTP).

**Сетевой уровень** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Network layer*) — 3-й уровень [сетевой модели OSI](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI), предназначается для определения пути передачи данных. Отвечает за трансляцию [логических адресов](https://ru.wikipedia.org/wiki/IP-%D0%B0%D0%B4%D1%80%D0%B5%D1%81) и имён в [физические](https://ru.wikipedia.org/wiki/MAC-%D0%B0%D0%B4%D1%80%D0%B5%D1%81), определение кратчайших маршрутов, [коммутацию](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BC%D1%83%D1%82%D0%B0%D1%86%D0%B8%D1%8F_(%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D1%8B%D0%B5_%D1%81%D0%B5%D1%82%D0%B8)) и [маршрутизацию](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%80%D1%88%D1%80%D1%83%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F), отслеживание неполадок и заторов в сети. На этом уровне работает такое сетевое устройство, как [маршрутизатор](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%80%D1%88%D1%80%D1%83%D1%82%D0%B8%D0%B7%D0%B0%D1%82%D0%BE%D1%80).

**Канальный уровень** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Data Link layer*) — второй уровень [сетевой модели OSI](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI), предназначенный для передачи данных [узлам](https://ru.wikipedia.org/wiki/%D0%A3%D0%B7%D0%B5%D0%BB_%D1%81%D0%B5%D1%82%D0%B8), находящимся в том же [сегменте](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D0%B3%D0%BC%D0%B5%D0%BD%D1%82_%D1%81%D0%B5%D1%82%D0%B8" \o "Сегмент сети)[локальной сети](https://ru.wikipedia.org/wiki/%D0%9B%D0%BE%D0%BA%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C). Также может использоваться для обнаружения и, возможно, исправления ошибок, возникших на [физическом уровне](https://ru.wikipedia.org/wiki/%D0%A4%D0%B8%D0%B7%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D1%83%D1%80%D0%BE%D0%B2%D0%B5%D0%BD%D1%8C). Примерами протоколов, работающих на канальном уровне, являются: [Ethernet](https://ru.wikipedia.org/wiki/Ethernet" \o "Ethernet) для локальных сетей (многоузловой), [Point-to-Point Protocol (PPP)](https://ru.wikipedia.org/wiki/PPP_(%D1%81%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D0%BF%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB)" \o "PPP (сетевой протокол)), [HDLC](https://ru.wikipedia.org/wiki/HDLC) и [ADCCP](https://ru.wikipedia.org/w/index.php?title=ADCCP&action=edit&redlink=1) для подключений [точка-точка](https://ru.wikipedia.org/wiki/%D0%A2%D0%BE%D1%87%D0%BA%D0%B0-%D1%82%D0%BE%D1%87%D0%BA%D0%B0) (двухузловой).

**Физический слой** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *physical layer*) или **Физический уровень** — первый уровень [сетевой модели OSI](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI). Это нижний уровень модели OSI — физическая и электрическая среда для передачи данных. Физический уровень описывает способы передачи бит (а не пакетов данных) через физические среды линий связи, соединяющие сетевые устройства. На этом уровне описываются параметры сигналов, такие как амплитуда, частота, фаза, используемая модуляция, манипуляция. Решаются вопросы связанные с синхронизацией, избавлением от помех, скорости передачи данных.

**9. Понятие протокола, стек протоколов, протокол TCP/IP (общие характеристики).**

**Понятие протокола**

* ***Протокол*** — это набор правил и процедур, устанавливающих порядок осуществления связи.
* Все компьютеры, участвующие в обмене, должны работать по одним и тем же протоколам, чтобы по завершении передачи информация восстанавливалась в первоначальном виде.
* В соответствии с разделением сетевых функций по уровням сетевой модели OSI протоколы также делятся по уровням.

Остановимся на особенностях протоколов более высоких уровней, реализуемых программно.

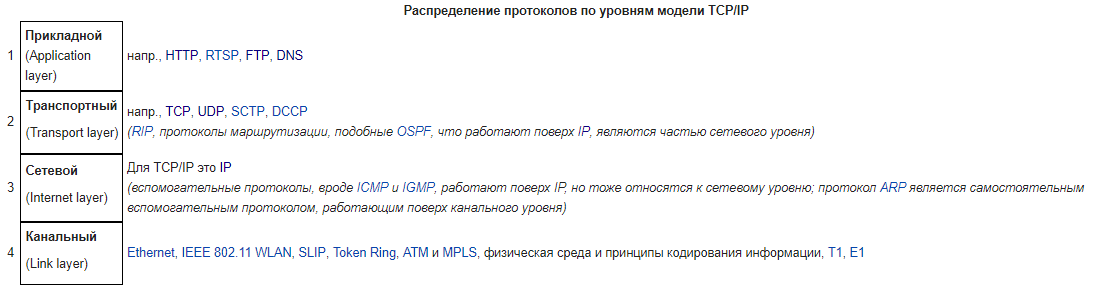
Протоколы перечисленных наборов делятся на три основные типа:

* ***прикладные*** протоколы (выполняющие функции прикладного, представительского и сеансового уровней модели OSI);
* ***транспортные*** протоколы (выполняющие функции транспортного уровня OSI);
* ***сетевые*** протоколы (выполняющие функции трех нижних уровней OSI).

TCP - Transfer Control Protocol (4-й уровень модели OSI - Транспортный)

IP - Internet Protocol (3-й уровень модели OSI - Сетевой)

TCP/IP – базовые транспортные и сетевые протоколы, которые обеспечивают установление надежного соединения между двумя машинами и собственно передачу данных, разбитых на последовательность фрагментов, между ними и осуществляя перепосылку в случае сбоя.



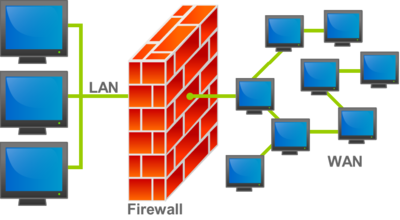
**10. Сетевое программное обеспечение (браузеры, фаерволы, поисковики).**

**Бра́узер**, или **веб-обозреватель** (от [англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *web browser*, МФА: [[wɛb ˈbraʊ.zə(ɹ), -zɚ]](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B6%D0%B4%D1%83%D0%BD%D0%B0%D1%80%D0%BE%D0%B4%D0%BD%D1%8B%D0%B9_%D1%84%D0%BE%D0%BD%D0%B5%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B0%D0%BB%D1%84%D0%B0%D0%B2%D0%B8%D1%82); устар. *бро́узер*[[1]](https://ru.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80" \l "cite_note-1)[[2]](https://ru.wikipedia.org/wiki/%D0%91%D1%80%D0%B0%D1%83%D0%B7%D0%B5%D1%80#cite_note-2)) — [прикладное программное обеспечение](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) для просмотра [веб-страниц](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0), содержания [веб-документов](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%B4%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82), [компьютерных файлов](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D1%8B%D0%B9_%D1%84%D0%B0%D0%B9%D0%BB) и их [каталогов](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D1%80%D0%B5%D0%BA%D1%82%D0%BE%D1%80%D0%B8%D1%8F_(%D1%84%D0%B0%D0%B9%D0%BB%D0%BE%D0%B2%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0)); управления [веб-приложениями](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B5); а также для решения других задач. В [глобальной сети](https://ru.wikipedia.org/wiki/%D0%92%D1%81%D0%B5%D0%BC%D0%B8%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D1%83%D1%82%D0%B8%D0%BD%D0%B0) браузеры используют для запроса, обработки, манипулирования и отображения содержания [веб-сайтов](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%B0%D0%B9%D1%82). Многие современные браузеры также могут использоваться для обмена файлами с серверами [ftp](https://ru.wikipedia.org/wiki/FTP" \o "FTP), а также для непосредственного просмотра содержания файлов многих графических форматов ([gif](https://ru.wikipedia.org/wiki/Gif" \o "Gif), [jpeg](https://ru.wikipedia.org/wiki/Jpeg" \o "Jpeg), [png](https://ru.wikipedia.org/wiki/Png" \o "Png), [svg](https://ru.wikipedia.org/wiki/Svg" \o "Svg)), аудио-видео форматов ([mp3](https://ru.wikipedia.org/wiki/Mp3), [mpeg](https://ru.wikipedia.org/wiki/Mpeg" \o "Mpeg)), текстовых форматов ([pdf](https://ru.wikipedia.org/wiki/Pdf" \o "Pdf), [djvu](https://ru.wikipedia.org/wiki/Djvu" \o "Djvu)) и других файлов.

**Межсетево́й экра́н**, **сетево́й экра́н** — [программный](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) или программно-аппаратный элемент [компьютерной сети](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C), осуществляющий контроль и фильтрацию проходящего через него [сетевого трафика](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D1%82%D1%80%D0%B0%D1%84%D0%B8%D0%BA) в соответствии с заданными правилами[[1]](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B6%D1%81%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D1%8D%D0%BA%D1%80%D0%B0%D0%BD#cite_note-.D0.9B.D0.B5.D0.B1.D0.B5.D0.B4.D1.8C.E2.80.942002.E2.80.94.E2.80.9422-1).

Другие названия[[2]](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B6%D1%81%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D1%8D%D0%BA%D1%80%D0%B0%D0%BD#cite_note-.D0.A8.D0.B0.D0.BD.D1.8C.D0.B3.D0.B8.D0.BD.E2.80.942011.E2.80.94.E2.80.94193-2):

* **Брандма́уэр** ([нем.](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BC%D0%B5%D1%86%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Brandmauer* — [противопожарная стена](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%B8%D0%B2%D0%BE%D0%BF%D0%BE%D0%B6%D0%B0%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D1%80%D0%B5%D0%B3%D1%80%D0%B0%D0%B4%D0%B0)) — заимствованный из немецкого языка термин;
* **Файрво́л** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *Firewall* — противопожарная стена) — заимствованный из английского языка термин.



Среди задач, которые решают межсетевые экраны, основной является защита [сегментов сети](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D0%B3%D0%BC%D0%B5%D0%BD%D1%82_%D1%81%D0%B5%D1%82%D0%B8) или отдельных [хостов](https://ru.wikipedia.org/wiki/%D0%A5%D0%BE%D1%81%D1%82) от [несанкционированного доступа](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D1%81%D0%B0%D0%BD%D0%BA%D1%86%D0%B8%D0%BE%D0%BD%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B4%D0%BE%D1%81%D1%82%D1%83%D0%BF) с использованием уязвимых мест в протоколах [сетевой модели OSI](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%B0%D1%8F_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D1%8C_OSI) или в программном обеспечении, установленном на компьютерах сети. Межсетевые экраны пропускают или запрещают трафик, сравнивая его характеристики с заданными шаблонами[[3]](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B6%D1%81%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D1%8D%D0%BA%D1%80%D0%B0%D0%BD#cite_note-.D0.9B.D0.B5.D0.B1.D0.B5.D0.B4.D1.8C.E2.80.942002.E2.80.94.E2.80.9422.E2.80.9425-3).

Наиболее распространённое место для установки межсетевых экранов — граница периметра [локальной сети](https://ru.wikipedia.org/wiki/%D0%9B%D0%BE%D0%BA%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C) для защиты внутренних хостов от атак извне. Однако атаки могут начинаться и с внутренних узлов — в этом случае, если атакуемый хост расположен в той же сети, трафик не пересечёт границу сетевого периметра, и межсетевой экран не будет задействован. Поэтому в настоящее время межсетевые экраны размещают не только на границе, но и между различными сегментами сети, что обеспечивает дополнительный уровень безопасности[[4]](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B6%D1%81%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D1%8D%D0%BA%D1%80%D0%B0%D0%BD#cite_note-.D0.9B.D0.B0.D0.BF.D0.BE.D0.BD.D0.B8.D0.BD.D0.B0.E2.80.942014.E2.80.94.E2.80.9443-4).

**Поиско́вая систе́ма** ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) *search engine*) — это [компьютерная система](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE-%D0%B0%D0%BF%D0%BF%D0%B0%D1%80%D0%B0%D1%82%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%BC%D0%BF%D0%BB%D0%B5%D0%BA%D1%81), предназначенная для [поиска информации](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B9_%D0%BF%D0%BE%D0%B8%D1%81%D0%BA). Одно из наиболее известных применений поисковых систем — [веб-сервисы](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D0%BB%D1%83%D0%B6%D0%B1%D0%B0) для поиска текстовой или графической информации во [Всемирной паутине](https://ru.wikipedia.org/wiki/%D0%92%D1%81%D0%B5%D0%BC%D0%B8%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D1%83%D1%82%D0%B8%D0%BD%D0%B0). Существуют также системы, способные искать [файлы](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB) на [FTP](https://ru.wikipedia.org/wiki/FTP)-серверах, товары в [интернет-магазинах](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D0%BC%D0%B0%D0%B3%D0%B0%D0%B7%D0%B8%D0%BD), информацию в группах новостей [Usenet](https://ru.wikipedia.org/wiki/Usenet" \o "Usenet).

Для поиска информации с помощью поисковой системы пользователь формулирует [поисковый запрос](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B8%D1%81%D0%BA%D0%BE%D0%B2%D1%8B%D0%B9_%D0%B7%D0%B0%D0%BF%D1%80%D0%BE%D1%81)[[1]](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B8%D1%81%D0%BA%D0%BE%D0%B2%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0#cite_note-Chu_.26_Rosenthal.E2.80.941996.E2.80.94.E2.80.94129-1). Работа поисковой системы заключается в том, чтобы по запросу пользователя найти документы, содержащие либо указанные [ключевые слова](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87%D0%B5%D0%B2%D1%8B%D0%B5_%D1%81%D0%BB%D0%BE%D0%B2%D0%B0), либо слова, как-либо связанные с ключевыми словами[[2]](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B8%D1%81%D0%BA%D0%BE%D0%B2%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0#cite_note-Tarakeswar_.26_Kavitha_.E2.80.942011.E2.80.94.E2.80.9429-2). При этом поисковая система генерирует [страницу результатов поиска](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0_%D1%80%D0%B5%D0%B7%D1%83%D0%BB%D1%8C%D1%82%D0%B0%D1%82%D0%BE%D0%B2_%D0%BF%D0%BE%D0%B8%D1%81%D0%BA%D0%B0). Такая поисковая выдача может содержать различные типы результатов, например: [веб-страницы](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0), [изображения](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D0%B0%D1%84%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B5_%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D1%8B), [аудиофайлы](https://ru.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D1%8B%D0%B5_%D0%B0%D1%83%D0%B4%D0%B8%D0%BE%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D1%8B). Некоторые поисковые системы также извлекают информацию из подходящих [баз данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) и [каталогов ресурсов в Интернете](https://ru.wikipedia.org/wiki/%D0%9A%D0%B0%D1%82%D0%B0%D0%BB%D0%BE%D0%B3_%D1%80%D0%B5%D1%81%D1%83%D1%80%D1%81%D0%BE%D0%B2).

**11. Способы передачи данных: модуляция, защита от ошибок, сжатие данных.**

*Модуляция* - преобразования сообщения в сигнал путем изменения параметров переносчика под действием сообщения.

Под *переносчиком* или несущим колебанием в системах телекоммуникаций понимается физический процесс, способный распространяться по линии связи и изменять свои параметры под действием сообщения.

Различают два вида переносчика: *непрерывный и импульсный*.

В качестве *непрерывного* переносчика используется гармоническое колебание

, (1)

которое характеризуется тремя параметрами: амплитудой , частотой  и фазой .

В качестве *импульсного* переносчика используется периодическая последовательность импульсов (рис.1), которая характеризуется амплитудой импульса , длительностью импульса τ0, периодом следования импульсов T0.

T0

u0(t)

t

u0



Рис.1

В зависимости от того, какой из параметров импульсного переносчика изменяется под действием сообщения, различают:

* амплитудно-импульсную (АИМ),
* широтно-импульсную (ШИМ),
* фазо-импульсную (ФИМ) модуляцию.

(интернет)

Выделяют две основные ***причины возникновения ошибок***при передаче информации в сетях:

* *сбои* в какой-то части оборудования сети или возникновение неблагоприятных объективных событий в сети.
* *помехи*, вызванные внешним источниками и атмосферными явлениями. Помехи – это электрические возмущения, возникающие в аппаратуре или попадающие в нее извне.

Выделяют три группы методов защиты от ошибок:

***1) помехоустойчивое кодирование***

Управление правильностью передачи информации выполняется с помощью помехоустойчивого кодирования. Разработка и использование корректирующих (помехоустойчивых) кодов широко применяется не только в ТКС, но и в ЭВМ для защиты от ошибок при передаче информации между устройствами машины. Оно позволяет получить более высокие качественные показатели работы систем связи. Его основное назначение - обеспечение малой вероятности искажений передаваемой информации, несмотря на присутствие помех или сбоев в работе сети.

Существует довольно большое количество различных помехоустойчивых кодов, отличающиеся друг от друга по ряду показателей и прежде всего по своим корректирующим возможностям, различают коды, обнаруживающие ошибки, и корректирующие коды, которые дополнительно к обнаружению еще и исправляют ошибки.

К числу наиболее важных *показателей корректирующих кодов* относятся:

* значимость кода, или длина кодовой комбинации, включающей информационные символы (***m***) и проверочные, или контрольные. символы (***k***). Обычно значимость кода ***n*** есть сумма ***m+k***?, (***n = m + k***);
* ***избыточность кода Кизб, выражаемая отношением числа контрольных*** символов в кодовой комбинации к значимости кода ***К=k/n***;
* ***корректирующая способность кода Ккс, представляет собой отношение*** числа кодовых комбинаций ***L***, в которых ошибки были обнаружены и исправлены, к общему числу переданных кодовых комбинаций ***М Ккс=L/M***

Выбор корректирующего кода зависит от требований по достоверности передачи информации. Для правильного выбора кода необходимы статистические данные о закономерностях появления ошибок, их характере, численности и распределении во времени. Например, корректирующий код, обнаруживающий и исправляющий одиночные ошибки эффективен лишь при условии, что ошибки статически независимы, а вероятность их появления не превышает некоторой величины. Он оказывается непригодным, если ошибки появляются группами. При выборе кода надо стремится, чтобы он имел меньшею избыточность. Чем больше коэффициент Кизб, тем менее эффективно используется пропускная способность канала связи и больше затрачивается времени на передачу информации, но зато выше помехоустойчивость системы.

В качестве надежных обнаруживающих кодов применяются *циклические*коды. Примером корректирующего кода является код Хемминга, позволяющий обнаруживать и исправлять одиночные ошибки.

В ТКС корректирующие коды в основном применяются для обнаружения ошибок, исправление которых осуществляется путем повторной передачи искаженной информации. С этой целью в сетях используются системы передачи с обратной связью (наличие между абонентами дуплексной связи облегчает применение таких систем).

***2) методы защиты от ошибок в системах передачи с обратной связью;***

**Системы передачи с обратной связью**делятся на:

* ***системы с решающей обратной связью;***
* ***системы с информационной обратной связью.***

Особенностью *систем с решающей обратной связью* (иначе систем с перезапросом) является то, что решение о необходимости повторной передачи информации (сообщения, пакета) принимает приёмник. Здесь обязательно применяется помехоустойчивое кодирование, с помощью которого на приемной станции осуществляется проверка принимаемой информации. При обнаружении ошибки на передающею сторону по каналу обратной связи посылается сигнал перезапроса, по которому информация передается повторно. Канал обратной связи используется также для посылки сигнала подтверждения правильности приема, автоматически определяющего начало следующей передачи.

В *системах с информационной обратной связью* передача информации осуществляется без помехоустойчивого кодирования, приемник, приняв информацию по прямому каналу и зафиксировав её в своей памяти, передает её в полном объеме по каналу обратной связи передатчику, где переданная и возвращенная информация сравнивается. При совпадении передатчик посылает приемнику сигнал подтверждения, в противном случае происходит повторная передача всей информации. Таким образом, здесь решение о необходимости повторной передачи принимает передатчик.

Обе рассмотренные системы обеспечивают практически одинаковую достоверность, однако в системах с обратной решающей связью пропускная способность каналов используется эффективнее, поэтому они получили большее распространение.

***3) групповые методы.***

Из *групповых методов* получили широкое применение:

* мажоритарный метод;
* метод передачи информационными блоками с количественной характеристикой блока.

Суть ***мажоритарного метода,*** давно и широко используемого в телеграфии, состоит в следующим. Каждое сообщение ограниченной длины передается несколько раз, чаще всего три раза. Принимаемые сообщения запоминаются, а потом производится их поразрядное сравнивание. Суждение о правильности передачи выносятся по совпадению большинства из принятой информации методом «два из трех». Например, кодовая комбинация 01101 при трехразовой передаче была частично искажена помехами, поэтому приемник принял такие комбинации: 10101, 01110, 01001. в результате проверки каждой позиции отдельно правильной считается комбинация 01101.

***Другой*** групповой ***метод***, так же не требующий перекодирования информации, предполагает ***передачу данных блоками с количественной характеристикой блока***. Такими характеристиками могут быть: число единиц и нулей в блоке, контрольная сумма передаваемая, остаток от деления контрольной суммы на постоянную величину и др. на приемном пункте эта характеристика вновь подсчитывается и сравнивается с переданной по каналу связи. Если характеристики совпадают, считается, что блок не содержит шибок. В противном случае на передающею сторону передается сигнал с требованием повторной передачи блока. В современных ТВС такой метод получил самое широкое распространение.

(лекции Гаврилова)

**Сжатие данных**

* *Сжатие (компрессия)* данных применяется для сокращения времени их передачи.
* Т.к. на компрессию данных передающая сторона тратит дополнительное время, к которому нужно еще прибавить затраты времени на декомпрессию этих данных принимающей стороной, то выгоды от сокращения времени на передачу сжатых данных обычно бывают заметны только для низкоскоростных каналов (мало времени на обработку на фоне длительной передачи). Этот порог скорости составляет около 64 Кбит/с.
* Многие программные и аппаратные средства сети способны выполнять *динамическую компрессию* данных в отличие от статической, когда данные предварительно сжимаются (например, с помощью популярных архиваторов типа WinZip), а уже затем отсылаются в сеть.
* На практике может использоваться ряд алгоритмов компрессии, каждый из которых применим к определенному типу данных. Некоторые модемы (называемые интеллектуальными) предлагают *адаптивную компрессию,* при которой в зависимости от передаваемых данных выбирается определенный алгоритм компрессии.

Рассмотрим некоторые из общих алгоритмов компрессии данных.

***Десятичная упаковка***

Когда данные состоят только из чисел, значительную экономию можно получить путем уменьшения количества используемых на цифру бит с 7 до 4, используя 4-х-разрядное двоичное кодирование десятичных цифр вместо 7-разрядного кода ASCII.

Просмотр таблицы ASCII показывает, что старшие три бита всех кодов десятичных цифр содержат комбинацию 011.

Если все данные в кадре информации состоят из десятичных цифр, то, поместив в заголовок кадра соответствующий управляющий символ, можно существенно сократить длину кадра.

***Относительное (адаптивное) кодирование***

Альтернативой десятичной упаковке при передаче числовых данных с небольшими отклонениями между последовательными цифрами является передача только этих отклонений вместе с известным опорным значением. Такой подход используется, в частности, в рассмотренном ранее методе цифрового кодирования голосовых сообщений ADPCM (адаптивная импульсно-кодовая модуляция), передающем в каждом такте только разницу между соседними замерами голоса.

***Символьное замещение***

Часто передаваемые данные содержат большое количество повторяющихся байт. Например, при передаче черно-белого изображения черные поверхности будут порождать большое количество нулевых значений, а максимально освещенные участки изображения — большое количество байт, состоящих из всех единиц. Передатчик сканирует последовательность передаваемых байт и, если обнаруживает последовательность из трех или более одинаковых байт, заменяет ее специальной трехбайтовой последовательностью, в которой указывает значение байта, количество его повторений, а также отмечает начало этой последовательности *специальным управляющим символом.*

***Статистическое кодирование (неравномерное кодирование*** или ***коды переменной длины)****.*

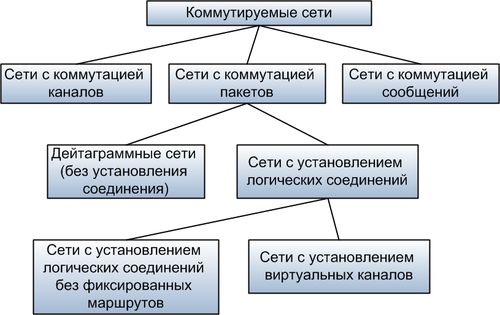
Этот метод кодирования использует тот факт, что не все символы в передаваемом кадре встречаются с одинаковой частотой. Поэтому кодовые комбинации часто встречающихся символов заменяют комбинации меньшей длины, а редко встречающихся — комбинации большей длины. Из-за того, что комбинации имеют различную длину, для передачи кадра возможна только бит-ориентированная передача

Такое неравномерное кодирование наиболее эффективно, когда неравномерность распределения частот передаваемых символов достаточна велика, как при передаче длинных текстовых строк. Напротив, при передаче двоичных данных, например кодов программ, оно малоэффективно, так как 8-битовые кодовые комбинации при этом распределены почти равномерно.

* Одним из наиболее распространенных алгоритмов, на основе которых строятся неравномерные коды, является ***алгоритм Хафмана***, позволяющий строить коды автоматически, на основании известных частот (вероятностей) появления символов.
* Существуют адаптивные модификации метода Хафмана, которые позволяют строить дерево кодов «на ходу», по мере поступления данных от источника.
* Многие модели коммуникационного оборудования, такие как модемы, мосты, коммутаторы и маршрутизаторы, поддерживают протоколы *динамической компрессии*, позволяющие сократить объем передаваемой информации в 4, а иногда и в 8 раз. В таких случаях говорят, что протокол обеспечивает коэффициент сжатия 1:4 или 1:8.
* Существуют стандартные протоколы компрессии, например **V.42bis**, a также большое количество нестандартных, фирменных протоколов.
* Реальный коэффициент компрессии зависит от типа передаваемых данных, так, графические и текстовые данные обычно сжимаются хорошо, а коды программ — хуже.

**12. Сети с коммутацией каналов.**

*Коммутация* — это процесс соединения различных абонентов коммуникационной сети через транзитные узлы. Коммуникационные сети должны обеспечивать связь своих абонентов между собой. Абонентами могут выступать ЭВМ, сегменты локальных сетей, факс-аппараты или телефонные собеседники.



***Коммутация каналов***

Коммутация каналов подразумевает образование непрерывного составного физического канала из последовательно соединенных отдельных канальных участков для прямой передачи данных между узлами. Отдельные каналы соединяются между собой специальной аппаратурой - коммутаторами, которые могут устанавливать связи между любыми конечными узлами сети. В сети с коммутацией каналов перед передачей данных всегда необходимо выполнить процедуру установления соединения, в процессе которой и создается составной канал.

Время передачи сообщения при этом определяется пропускной способностью канала, длинной связи и размером сообщения.

Коммутаторы, а также соединяющие их каналы должны обеспечивать одновременную передачу данных нескольких абонентских каналов. Для этого они должны быть высокоскоростными и поддерживать какую-либо технику мультиплексирования абонентских каналов.

*Достоинства коммутации каналов:*

1) постоянная и известная скорость передачи данных;

2) правильная последовательность прихода данных;

3) низкий и постоянный уровень задержки передачи данных через сеть.

*Недостатки коммутации каналов:*

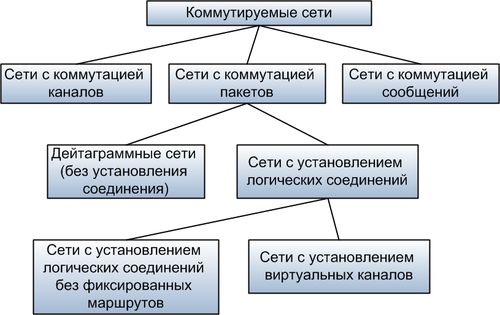
1) возможен отказ сети в обслуживании запроса на установление соединения;

2) нерациональное использование пропускной способности физических каналов, в частности невозможность применения пользовательской аппаратуры, работающей с разной скоростью. Отдельные части составного канала работают с одинаковой скоростью, так как сети с коммутацией каналов не буферизуют данные пользователей;

3) обязательная задержка перед передачей данных из-за фазы установления соединения.

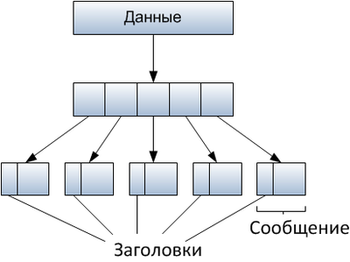
**13. Сети с коммутацией сообщений.**

*Коммутация* — это процесс соединения различных абонентов коммуникационной сети через транзитные узлы. Коммуникационные сети должны обеспечивать связь своих абонентов между собой. Абонентами могут выступать ЭВМ, сегменты локальных сетей, факс-аппараты или телефонные собеседники.



***Коммутация сообщений***

Коммутация сообщений – разбиение информации на сообщения, каждый из которых состоит из заголовка и информации.



Это способ взаимодействия, при котором создается логический канал, путем последовательной передачи сообщений через узлы связи по адресу указанному в заголовке сообщения.

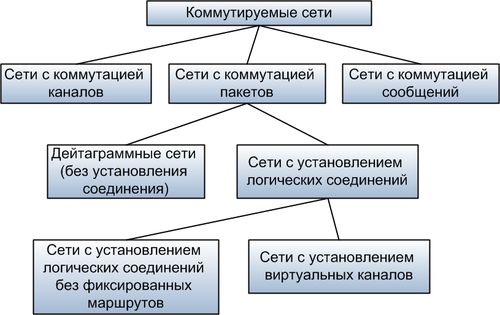
При этом каждый узел принимает сообщение, записывает в память, обрабатывает заголовок, выбирает маршрут и выдает сообщение из памяти в следующий узел.

Время доставки сообщения определяется временем обработки в каждом узле, числом узлов и пропускной способности сети. Когда заканчивается передача информации из узла А в узел связи В, то узел А становится свободным и может участвовать в организации другой связи между абонентами, поэтому канал связи используется более эффективно, но система управления маршрутизации будет сложной.

Сегодня коммутация сообщений в чистом виде практически не существует.

**14. Сети с коммутацией пакетов.**

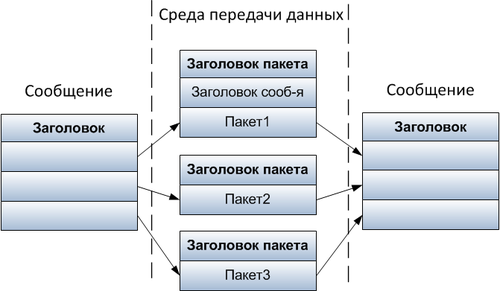
*Коммутация* — это процесс соединения различных абонентов коммуникационной сети через транзитные узлы. Коммуникационные сети должны обеспечивать связь своих абонентов между собой. Абонентами могут выступать ЭВМ, сегменты локальных сетей, факс-аппараты или телефонные собеседники.



***Коммутация пакетов***

Коммутация пакетов - это особый способ коммутации узлов сети, который специально создавался для наилучшей передачи компьютерного трафика (пульсирующего трафика). Опыты по разработке самых первых компьютерных сетей, в основе которых лежала техника коммутации каналов, показали, что этот вид коммутации не предоставляет возможности получить высокую пропускную способность вычислительной сети. Причина крылась в пульсирующем характере трафика, который генерируют типичные сетевые приложения.

При коммутации пакетов все передаваемые пользователем сети сообщения разбиваются в исходном узле на сравнительно небольшие части, называемые пакетами. Необходимо уточнить, что сообщением называется логически завершенная порция данных - запрос на передачу файла, ответ на этот запрос, содержащий весь файл, и т. п. Сообщения могут иметь произвольную длину, от нескольких байт до многих мегабайт. Напротив, пакеты обычно тоже могут иметь переменную длину, но в узких пределах, например от 46 до 1500 байт (EtherNet). Каждый пакет снабжается заголовком, в котором указывается адресная информация, необходимая для доставки пакета узлу назначения, а также номер пакета, который будет использоваться узлом назначения для сборки сообщения.



Коммутаторы пакетной сети отличаются от коммутаторов каналов тем, что они имеют внутреннюю буферную память для временного хранения пакетов, если выходной порт коммутатора в момент принятия пакета занят передачей другого пакета.

*Достоинства коммутации пакетов:*

1) более устойчива к сбоям;

2) высокая общая пропускная способность сети при передаче пульсирующего трафика;

3) возможность динамически перераспределять пропускную способность физических каналов связи.

*Недостатки коммутации пакетов:*

1) неопределенность скорости передачи данных между абонентами сети;

2) переменная величина задержки пакетов данных;

3) возможны потери данных из-за переполнения буферов;

4) возможны нарушения последовательности прихода пакетов.

В компьютерных сетях применяется коммутация пакетов.

**15. Маршрутизация в телекоммуникационных системах.**

Теперь мы вплотную подошли к проблеме выбора оптимального маршрута доставки информации от отправителя к получателю. Этот процесс называется ***маршрутизацией***, а выбор маршрута производится в соответствии с используемым ***алгоритмом маршрутизации***.

ПРИМЕЧАНИЕ

Если говорить о сетях, то в подсети, использующей дейтаграммную службу выбор маршрута для каждого пакета должен производиться заново, т.к. оптимальный маршрут мог измениться. Если используются виртуальные каналы, маршрут выбирается только при создании нового виртуального канала [12].

***Основную цель маршрутизации*** можно определить следующим образом – доставка пакета получателю за минимально возможное время, при сохранении требуемой пропускной способности и минимальных потерях информации. Здесь приходится учитывать и топологию сети и то, что маршрутизаторы (коммуникационные устройства, осуществляющие пересылку и маршрутизацию), а также линии связи могут выйти из строя.

В отечественной литературе, например, в [2] выделяют три способа маршрутизации: ***централизованная маршрутизация*** (выбор маршрута для каждого пакета осуществляется в центре управления сетью, что чревато отказом всей сети при выходе из строя центрального узла); ***распределенная*** ***маршрутизация*** (функции управления маршрутизацией распределены между узлами сети, а значит, обеспечивается большая гибкость);***смешанная маршрутизация*** сочетает принципы централизованной и распределенной маршрутизации.

Почему вопросами создания алгоритмов маршрутизации до сих пор занимается огромное количество ученых и инженеров, несмотря на то, что технология с успехом работает много лет? Ответ предельно просто и заключается в следующем – мы не знаем как будет изменяться нагрузка в сети в следующий момент времени. Хотя крайне важными параметрами остаются топология сети и ее изменение (в результате отказов узлов и линий связи, а также при подключении новых), а также различная пропускная способность участков сети. Но оба эти параметра можно определить и передать маршрутизаторам. А значит, во всех случаях алгоритмы маршрутизации выполняются в условиях неопределенности текущего и будущего состояний телекоммуникационной системы [2].

Рассмотрим несколько наиболее известных и широко применяемых алгоритмов маршрутизации. Сначала давайте проведем черту между существующими алгоритмами. Они разделяются на два больших класса:***адаптивные*** и ***неадаптивные***. Вместо того чтобы учитывать топологию сети и изменение ее состояния, а также измерять текущий трафик ***неадаптивные алгоритмы*** выбирают маршруты заранее. Полученный список маршрутов загружается в маршрутизаторы на этапе загрузке сети. Эта процедура называется ***статической маршрутизацией***. ***Адаптивные алгоритмы*** напротив охотно пользуются всеми параметрами, которые можно измерить.

Вне зависимости от топологии сети и интенсивности трафика все алгоритмы маршрутизации базируются на***принципе оптимальности*** и ***концепции кратчайшего пути***. В соответствии с ***принципом оптимальности***, если маршрутизатор *B* располагается на оптимальном маршруте от маршрутизатора *A* к маршрутизатору *C*, то оптимальный маршрут от маршрутизатора *B* к маршрутизатору *C* совпадает с частью первого маршрута [12].

ПРИМЕЧАНИЕ

Концепция кратчайшего пути, чаще всего поясняется на графах, где каждый узел сети – вершина графа, а дуга – линия связи. В этом случае для поиска кратчайшего пути на графах используется, например, алгоритм Дейкстры.

Наиболее простой статический алгоритм маршрутизации – ***заливка***. Здесь можно провести аналогию с методом полного перебора из методов оптимизации и поиска экстремума функции. Суть ***заливки***заключается в том, что каждый пришедший пакет посылается на все исходящие линии, кроме той, по которой он поступил. Это порождает бесконечное число дублированных пакетов. Для ограничения количества тиражируемых пакетов используются счетчики, которые помещаются в заголовок пакета и уменьшаются при прохождении каждого маршрутизатора. Если счетчик обнуляется, то такой пакет удаляется. Кроме данного метода применяются еще ряд вариаций на тему счетчиков и помещения в заголовки номера пройденного узла.

Однако на практике применяется ***выборочная заливка***. Отличие его состоит в том, что пакеты посылаются не на все исходящие линии, а только на те, которые идут в приблизительно правильном направлении. Такой «громоздкий» алгоритм может подойти военным, где вероятность потери части сети из-за боевых действий велика, а также для тестирования других алгоритмов. Напомним, что также как и метод полного перебора, заливка рано или поздно найдет оптимальный маршрут.

На практике, современные системы телекоммуникаций применяю не статические, а динамические алгоритмы маршрутизации.

**16. Цифровые сети с интеграцией служб (ISDN).**

Технология ISDN появилась в 1984 году. Цифровая сеть с интегрированными услугами (ISDN - Integrated Services Digital Network) - система, в которой по телефонным каналам передаются только цифровые сигналы, в том числе и по абонентским линиям, т.е. конечный абонент передает данные непосредственно в цифровой форме.

ISDN позволяет объединить передачу голоса, данных и изображения. Интеграция разнородных трафиков ISDN выполняется, используя способ временного разделения (TDM – Time Division Multiplexing). ISDN использует цифровые каналы в режиме коммутации каналов.

Цифровые сети с интеграцией услуг ISDN можно использовать при передаче голоса и данных, для объединения удаленных ЛВС, для доступа к сети Internet и для различных видов трафика, в том числе мультимедийного.

Оконечными устройствами в сети ISDN могут быть: цифровой телефонный аппарат, компьютер с ISDN-адаптером, видео- и аудиооборудование.

Суть технологии ISDN, состоит в том, что различные устройства, например, телефоны, компьютеры, факсы и другие устройства, могут одновременно передавать и принимать цифровые сигналы после установления коммутируемого соединения с удаленным абонентом.

Сети ISDN состоят из двух В-каналов, дополнительного D-канала и H-канала.

В ISDN основной поток информации (голос и данные) передается по В-каналам. Эти каналы коммутируются между парой абонентов с помощью информации, передаваемой по дополнительному каналу – D-каналу. H-канал - это канал высокоскоростной передачи данных со скоростями 384 кбит/с (канал H0), 1563 кбит/с (канал H11), 1920 кбит/с (канал H12).

После коммутации каждый В-канал представляет собой две “трубы”, пропускающие во встречных направлениях потоки битов со скоростью 64 кбит/с. Служебный канал – также двунаправленный, его пропускная способность может быть 16 или 64 кбит/с в зависимости от типа сервиса.

Скорость передачи данных в ISDN может быть: 64 кбит/с., 128 кбит/с, а в широкополосных каналах связи до 155 Мбит/с. Через линии ISDN возможна передача данных с помощью технологий и протоколов глобальных сетей: Х.25, Frame Relay.

**17. Локальные вычислительные сети.**

**Понятие локальной вычислительной сети (ЛВС)**

Существуют разные определения локальной сети. В эти определения авторы закладывают разные отличительные признаки, в числе которых:

* территориальный,
* количество компьютеров, объединяемых ЛВС,
* характер среды распространения сигналов (однородная или неоднородная),
* скорость передачи данных.

Наиболее точно представляется определить ЛВС как систему, которая позволяет не замечать задержек связи, т.е. как систему, в которой компьютеры объединены в один ***виртуальный компьютер***, ресурсы которого могут быть доступны любому пользователю, причем этот доступ не менее удобен, чем доступ к ресурсам собственного компьютера. Под удобством в первую очередь понимается ***высокая скорость доступа***, при которой обмен данными между приложениями происходит незаметно для пользователя. Под такое определение не попадают ни глобальные сети, ни медленная связь через последовательный порт компьютера.

Из такого определения следует, что скорость передачи по ЛВС должна расти с ростом быстродействия компьютеров. Это и наблюдается на практике. Если раньше приемлемой считалась скорость 1-10 Мбит/с, то сейчас среднескоростной считается сеть, работающая со скоростью 100 Мбит/с и входят в жизнь сети со скоростью передачи 1000 Мбит/с и более. При меньших скоростях сеть станет узким местом, будет замедлять работу объединенного виртуального компьютера.

**Серверные и одноранговые сети**

* В ***одноранговой*** сети все компьютеры равноправны – могут выполнять как функции клиента, так и сервера.
* В ***серверной*** сети (с централизованным управлением) функции сервера и клиента разделены и принадлежат определенным компьютерам.

**Назначение ЛВС**

ЛВС применяются для:

* совместного использования ресурсов (дисковой память, принтеры, сканеры, выход в глобальную сеть),
* совместной (распределенной) обработки информации, что позволяет многократно ускорить решение сложных математических задач;
* оперативного обмена информацией;
* управления сложными технологическими процессами.

**Отличительные признаки ЛВС**

* высокая скорость передачи данных;
* низкий уровень ошибок или высокая достоверность приема (допустимая вероятность ошибок должна быть порядка 10-7-10-8);
* Ограниченное число компьютеров, подключаемых к сети;
* качественные, специально прокладываемые линии связи (в тоже время в жизнь входят беспроводные ЛВС – wireless network).

**Недостатки ЛВС**

Всегда следует помнить следующее.

1. Установка ЛВС сопряжена с ***дополнительными затратами:***

* на приобретения сетевого оборудования,
* сетевого программного обеспечения,
* прокладку сетевого кабеля,
* обучение персонала,
* оплату специалиста, занимающегося управлением доступом к ресурсам и обслуживанием сети, а также отвечающего за безопасность информации в сети (сохранность и защиту от несанкционированного доступа) – ***сетевого администратора***.

2. Сеть – прекрасная среда для распространения ***компьютерных вирусов***. Поэтому следует принимать особые меры для защиты от вирусов.

***3. Перемещение компьютеров*** в установленной сети сопряжено с определенными трудностями. Хотя этот недостаток не распространяется на ***структурированные сети***, в которых рабочие станции подключаются к заранее размещенным точкам подключения, и ***беспроводные*** *ЛВС* (wireless networks).

**Понятие корпоративной сети**

ЛВС отдельных предприятий, фирм, банков, офисов, корпораций, имеющие пользователей, расположенных в разных помещениях и использующие инфраструктуру глобальной сети Интернет, называют ***корпоративными сетями***или сетями ***интранет***(***intranet***).

**18. Глобальные вычислительные сети.**

**Понятие глобальной сети**

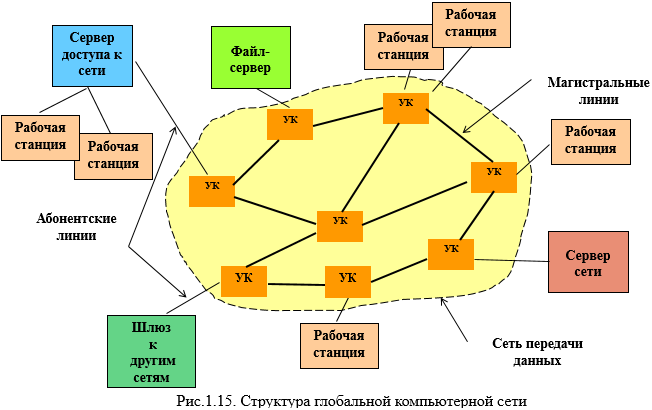
Глобальные сети не отвечают требованиям, предъявляемым к ЛВС. Для них характерно следующее:

* рассчитаны на неограниченное число абонентов;
* часто используют низкокачественные каналы связи и низкую скорость передачи данных: десятки - сотни Кбит/с;
* в глобальных сетях важно не качество связи, а факт ее существования.

Т.о., глобальная сеть – это сеть, охватывающая большую территорию и обладающая вышеуказанными признаками.

Примером глобальной сети является сеть Internet, охватывающая весь земной шар, известная по аббревиатуре www – world wide web – «всемирная паутина».

**Топология глобальных сетей. Сервер сети. Файл-сервер. Серверы доступа.**



В отличие от ЛВС, глобальные сети характеризуются (рис.1.15) достаточно сложной неоднородной топологией.

Основу передающей среды ГВС составляют узлы коммутации (УК), в качестве которых выступают ***связные процессоры***. Узлы коммутации соединены между собой с помощью высокоскоростных каналов передачи данных (магистральных каналов), к которым предъявляются высокие требования безошибочной передачи информации..

В глобальных сетях, как правило, используется несколько выделенных серверов:

* управляет работой сети специальный компьютер – **сервер сети**;
* в больших сетях может работать несколько **файл-серверов**, служащих для хранения больших объемов информации и организации эффективного доступа к ней со стороны рабочих станций;
* глобальные сети предполагают подключение большого числа рабочих станций; для этой цели используются специальные **серверы доступа**, с помощью которых обеспечивается эффективный доступ рабочих станций к компьютерной сети.

Количество и расположение узлов коммутации выбирается так, чтобы при минимальных затратах обеспечить требуемую пропускную способность сети

**19. Топология сетей ЭВМ.**

**Базовые топологии локальных сетей**

Термин топология, или более точно, сетевая топология – это способ организации связей компьютеров в сети. Топология – это стандартный термин, которым пользуются специалисты, когда характеризуют физические или логические связи компьютеров в сети.

Следует различать ***физическую*** и ***логическую*** топологии.

***Физическая*** топология определяет конфигурацию физических связей, т.е. электрических соединений компьютеров в сети.

***Логическая*** топология определяет маршруты передачи данных между компонентами сети и определяется способом работы коммуникационного оборудования.

Далее рассматриваются базовые физические топологии сети.

Структура связей в ЛВС легко прослеживается.

Обычно выделяют три базовых (основных) топологии сети:

* ***Шинная*** (**bus**), при которой все рабочие станции подключаются к общей разделяемой (совместно используемой - share) передающей среде - магистрали (шине) с помощью *сетевых адаптеров* (устройств сопряжения компьютера с линией связи); конструктивно адаптер представляет собой плату - карту, встраиваемую в компьютер. Аналогичным образом к шине подключаются и другие сетевые устройства (рис.1.2).
* ***Звездообразная*** (**star**) или ***радиальная***, при которой к одному центральному узлу присоединяются все остальные компьютеры, причем каждый из них использует свою отдельную линию связи (рис.1.3). Сообщения от передающей станции поступают на адаптеры всех рабочих станций, однако воспринимаются только адаптером той станции, которой они адресованы.
* ***Кольцевая*** (**ring**), при которой компьютеры объединены в замкнутое однонаправленное кольцо (рис.1.4).

**20. Основные типы сетевых устройств.**

К основным типам сетевых утройств относят:

* повторители,
* концентраторы,
* мосты,
* коммутаторы,
* маршрутизаторы.

Это оборудование применяется для структуризации сети - разбиения сети на сегменты. Оборудование такого рода также называют ***коммуникационным***, имея в виду, что с помощью него отдельные сегменты сети взаимодействуют между собой.

Различают физическую и логическую структуризацию сети.

**Повторители**

Простейшее из коммуникационных устройств — ***повторитель (repeater)*** *—* используется для физического соединения различных сегментов кабеля локальной сети с целью увеличения общей длины сети. Повторитель передает сигналы, приходящие из одного сегмента сети, в другие ее сегменты (рис. 1).

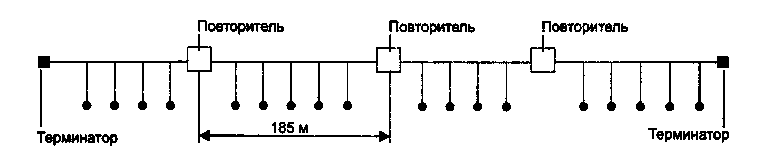


Рис. 1. Повторитель позволяет увеличить длину сети Ethernet

Повторитель позволяет преодолеть ограничения на длину линий связи за счет улучшения качества передаваемого сигнала — восстановления его мощности и формы (рис.2).

C:\Users\Лена\AppData\Local\Microsoft\Windows\INetCache\Content.Word\28.png

Рис. 2. Повторитель позволяет восстановить форму сигнала



**Концентраторы**

* Повторитель, который имеет несколько портов и соединяет несколько физических сегментов, называют ***концентратором*** *(concentrator)* или *хабом (hub).* Эти названия (hub — основа, центр деятельности) отражают тот факт, что в данном устройстве сосредоточиваются все связи между сегментами сети.
* Концентраторы характерны практически для всех базовых технологий локаль­ных сетей — Ethernet, ArcNet, Token Ring, FDDI, Fast Ethernet, Gigabit Ethernet, l00VG-AnyLAN.
* В работе концентраторов любых технологий много общего — они повторяют сигналы, пришедшие с одного из своих портов, на других своих портах.
* Концентраторы могут отличаться тем, на каких портах повторяются входные сигналы.
* Концентратор Ethernet повторяет входные сигналы на всех своих портах, кроме того, с которого сигналы поступают (рис.3,а)*.*
* Концентратор Token Ring (рис.3,б) повторяет входные сигналы, поступающие с некоторого порта, только на одном порту - на том, к которому подключен

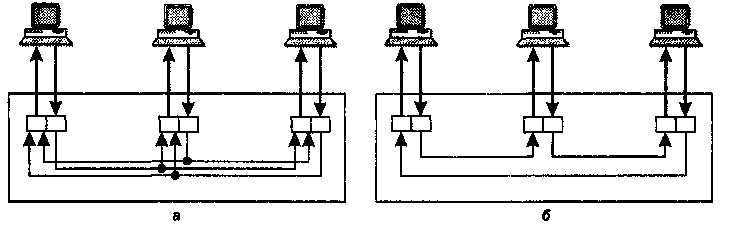


Рис. 3. Концентраторы различных технологий:

а) физическая (звезда) и логическая (шина) топологии не совпадают,   
б) физическая (звезда) и логическая топологии (кольцо) не совпадают



**Коммутаторы**

* *Коммутатор (switch, switching hub)* по принципу обработки кадров не отличается от моста. Основное его отличие от моста состоит в том, что он является своего рода коммуникационным мультипроцессором, так как каждый его порт оснащен специализированным процессором, который обрабатывает кадры по алгоритму моста независимо от процессоров других портов. За счет этого общая производительность коммутатора обычно намного выше производительности традиционного моста, имеющего один процессорный блок. Можно сказать, что коммутаторы — это мосты, которые обрабатывают кадры в параллельном режиме.
* Ограничения, связанные с применением мостов и коммутаторов — по топологии связей, а также ряд других, — привели к тому, что в ряду коммуникационных устройств появился еще один тип оборудования — *маршрутизатор).*

**Коммутаторы**

* Работают на канальном уровне!
* В отличие от концентратора, коммутатор передает данные только непосредственно получателю
* Если сеть занята, кадр не теряется, а ожидает в буфере коммутатора. Таким образом, не возникает коллизий.
* Управление коммутатором может осуществляться посредством   
  веб-интерфейса



**Маршрутизаторы**

* Маршрутизатор (router)более надежно и более эффективно, чем мосты, изолируют трафик отдельных частей сети друг от друга. Маршрутизаторы образуют логические сег­менты посредством явной адресации, поскольку используют не «плоские» аппарат­ные, а составные числовые адреса. В этих адресах имеется поле номера сети, так что все компьютеры, у которых значение этого поля одинаково, принадлежат к одному сегменту, называемому в данном случае *подсетью (subnet).*
* Кроме локализации трафика маршрутизаторы выполняют еще много других полезных функций. Так, маршрутизаторы могут работать в сети с замкнутыми контурами, при этом они осуществляют выбор наиболее рационального маршрута из нескольких возможных.
* Сеть, представленная на рис. 1.7, отличается от своей предшественницы (см. рис. 1.6 тем, что между подсетями отделов 1 и 2 проложена дополнительная связь, которая может использоваться как для повышения производительности сети, так и для повышения ее надежности.
* Другой очень важной функцией маршрутизаторов является их способность связывать в единую сеть подсети, построенные с использованием разных сетевых технологий, например Ethernet и Х.25.

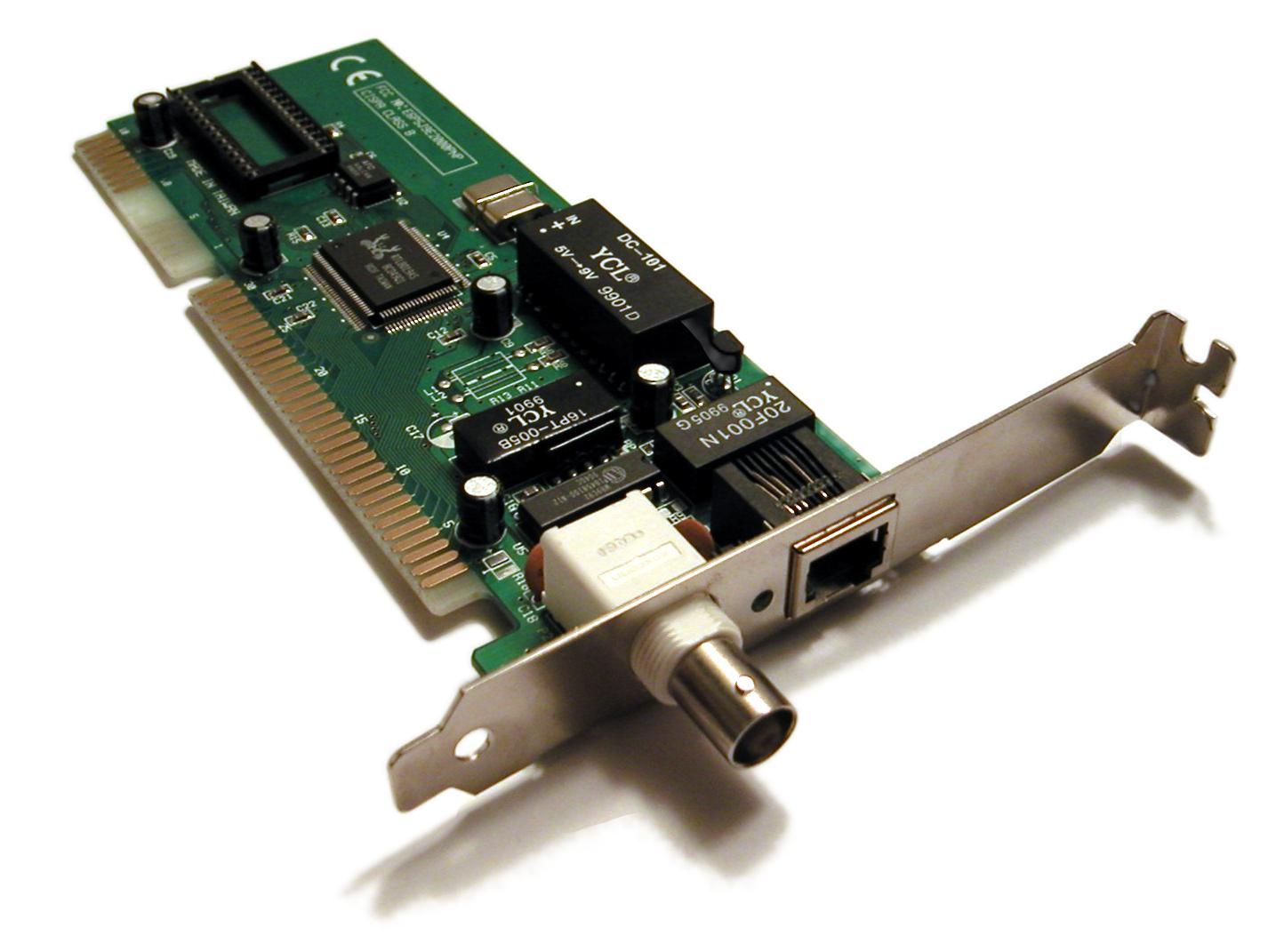
**Маршрутизаторы (router)** представляют собой устройства объединенных сетей, которые пересылают пакеты между сетями на основе адресов третьего уровня. Маршрутизаторы способны выбирать наилучший путь в сети для передаваемых данных.



### Сетевые карты

Устройства, которые связывают конечного пользователя с сетью, называются также **оконечными узлами или станциями (host)**. Примером таких устройств является обычный персональный компьютер или **рабочая станция** (мощный компьютер, выполняющий определенные функции, требующие большой вычислительной мощности. Например, обработка видео, моделирование физических процессов и т.д.). Для работы в сети каждый **хост** оснащен **платой сетевого интерфейса (Network Interface Card — NIC)**, также называемой **сетевым адаптером**. Как правило, такие устройства могут функционировать и без компьютерной сети.

Сетевой адаптер представляет собой печатную плату, которая вставляется в слот на материнской плате компьютера, или внешнее устройство. Каждый адаптер NIC имеет уникальный код, называемый MAC-адресом. Этот адрес используется для организации работы этих устройств в сети. Сетевые устройства обеспечивают транспортировку данных, которые необходимо передавать между устройствами конечного пользователя. Они удлиняют и объединяют кабельные соединения, преобразуют данные из одного формата в другой и управляют передачей данных. Примерами устройств, выполняющих перечисленные функции, являются **повторители, концентраторы, мосты, коммутаторы и маршрутизаторы**.

[](http://itandlife.ru/wp-content/uploads/2011/09/NIC.jpg)

Мосты

**Мост (bridge)** представляет собой устройство второго уровня, предназначенное для создания двух или более сегментов локальной сети LAN, каждый из которых является отдельным коллизионным доменом. Иными словами, мосты предназначены для более рационального использования полосы пропускания. Целью моста является фильтрация потоков данных в LAN-сети с тем, чтобы локализовать внутрисегментную передачу данных и вместе с тем сохранить возможность связи с другими  
частями (сегментами) LAN-сети для перенаправления туда потоков данных. Каждое сетевое устройство имеет связанный с NIC-картой уникальный MAC-адрес. Мост  
собирает информацию о том, на какой его стороне (порте) находится конкретный MAC-адрес, и принимает решение о пересылке данных на основании соответствующего списка MAC-адресов. Мосты осуществляют фильтрацию потоков данных на основе только MAC-адресов узлов. По этой причине они могут быстро пересылать данные любых протоколов сетевого уровня. На решение о пересылке не влияет тип используемого протокола сетевого уровня, вследствие этого мосты принимают решение только о том, пересылать или не пересылать фрейм, и это решение основывается лишь на MAC-адресе получателя. Ниже приведены наиболее важные свойства мостов.



**21. Основные технологии WWW.**

*Всемирная паутина*  (англ. World Wide Web) — глобальное информационное пространство, основанное на физической инфраструктуре Интернета и протоколе передачи данных HTTP. Всемирная паутина вызвала настоящую революцию в информационных технологиях и бум в развитии Интернета. Часто, говоря об Интернете, имеют в виду именно Всемирную паутину. Для обозначения Всемирной паутины также используют слово веб (англ. web) и аббревиатуру «WWW».

*Всемирную паутину образуют миллионы веб-серверов сети Интернет, расположенные по всему миру. Веб-сервер* - программа, запускаемая на подключённом к сети компьютере. В простейшем виде такая программа получает по сети HTTP-запрос на определённый ресурс, находит соответствующий файл на локальном жёстком диске и отправляет его по сети запросившему компьютеру. Более сложные веб-серверы способны динамически формировать ресурсы в ответ на HTTP-запрос.

Для просмотра информации, полученной от веб-сервера, на клиентском компьютере применяется специальная программа-клиент — *веб-бра́узер (web-browser). Основная функция веб-браузера — отображение гипертекста.*

Всемирная паутина неразрывно связана с понятиями гипертекста*.* Гипертекст (Hypertext) – это документ или система документов с перекрестными ссылками (*гиперссылками*). По такому документу можно перемещаться не последовательно, а, активируя гиперссылки, переходить по ним к связанным со ссылками текстам или файлам.

Для создания, хранения и отображения гипертекста используется язык *HTML (англ. HyperText Markup Language)*, язык *разметки* гипертекста. Работа по разметке гипертекста называется вёрсткой, специалисты по разметке называются *веб-мастерами*.

*HTML-файл является самым распространённым ресурсом Всемирной паутины.* HTML-файл доступный веб-серверу, называют «*веб-страницей*». Набор связанных тематикой, дизайном или владельцем веб-страниц образует *веб-сайт*.

*Гиперссылки помогают пользователям Всемирной паутины легко перемещаться между страницами одного сайта и между различными сайтами.*

**22. Технология PHP. Встраивание PHP в HTML.**

*PHP — скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений.* В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков, применяющихся для создания динамических веб-сайтов.

В области веб-программирования, в частности серверной части, *PHP — один из популярных сценарных языков.*

*Популярность в области построения веб-сайтов определяется наличием большого набора встроенных средств для разработки веб-приложений. Основные из них:*

- автоматическое извлечение POST и GET-параметров, а также переменных окружения веб-сервера в предопределённые массивы;

- взаимодействие с большим количеством различных систем управления базами данных (MySQL, Oracle (OCI8), Oracle и др.);

- автоматизированная отправка HTTP-заголовков;

- работа с HTTP-авторизацией;

- работа с cookies и сессиями;

- работа с локальными и удалёнными файлами, сокетами;

- обработка файлов, загружаемых на сервер;

- работа с XForms.

*Существуют три основных области, где используется PHP:*

* cоздание скриптов для выполнения на стороне сервера;
* cоздание скриптов для выполнения в [командной строке](http://www.php.su/prepare/?commandline);
* cоздание приложений GUI, выполняющихся на стороне клиента.

Для работы программы не требуется описывать какие-либо переменные, используемые модули и т. п. Любая программа может начинаться непосредственно с оператора PHP.

Простейшая программа Hello world на PHP выглядит следующим образом:

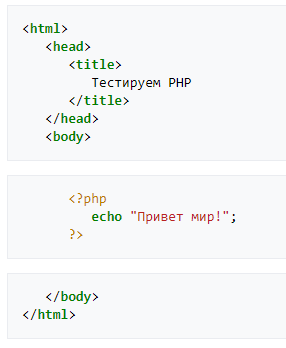


Также возможен более короткий вариант вывода строки:



Открывающий тег вида <?= используется для сокращённой записи конструкций используемых для вывода строки.

PHP исполняет код, находящийся внутри ограничителей, таких как <?php ?>. Всё, что находится вне ограничителей, выводится без изменений. В основном это используется для вставки PHP-кода в HTML-документ, например, так:



Помимо ограничителей <?php ?>, допускается использование сокращённого варианта <? ?>. Кроме того, до версии 7.0 допускалось использование ограничителей языка программирования ASP <% %> и <script language="php"> </script>. Работа сокращённых конструкций определяется в конфигурационном файле php.ini.

Имена переменных начинаются с символа $, тип переменной объявлять не нужно. Имена переменных и констант чувствительны к регистру символов. Имена классов, методов классов и функций к регистру символов не чувствительны. Переменные обрабатываются в строках, заключённых в двойные кавычки, и heredoc-строках (строках, созданных при помощи оператора <<<). Переменные в строках, заключённых в одинарные кавычки, не обрабатываются.

PHP рассматривает переход на новую строку как пробел, так же как HTML и другие языки со свободным форматом. Инструкции разделяются с помощью точки с запятой (;), за исключением некоторых случаев, после объявления конструкции if/else и циклов.

Переменные в функцию можно передавать как по значению, так и по ссылке (используется знак &).

PHP поддерживает три типа комментариев: в стиле языка Си (ограниченные /\* \*/), C++ (начинающиеся с // и идущие до конца строки) и оболочки UNIX (с # до конца строки).

**23. Каскадные таблицы стилей CSS.**

***Каскадные (многоуровневые) таблицы стилей***- *cascading style sheets (CSS)* - это мощный стандарт на основе текстового формата, определяющий представление данных в броузере.

*Если формат HTML предоставляет информацию о составе документа, то таблицы стилей сообщают как он должен выглядеть.* Таким образом каскадные таблицы стилей дают возможность хранить содержимое отдельно от его представления.

*Стиль включает все типы элементов дизайна:* шрифт, фон, текст, цвета ссылок, поля и расположение объектов на странице.

*CSS разрабатывались так, чтобы обеспечить больший уровень контроля над размещением текста и графики.*

Каскадные таблицы стилей обеспечивают должный уровень единства оформления, организации и контроля во время разработки узла, который является недостижимым с помощью одного только HTML.

***CSS предполагает 3 типа таблиц стилей - встроенные, внедренные (внутренние) и связанные (внешние).***

***Методы и синтаксис***

Существует ряд методов, с помощью которых таблицы стилей могут применяться к документу HTML. Синтаксис соответствует реальной структуре информации, содержащейся внутри таблицы стиля.

***Методы***

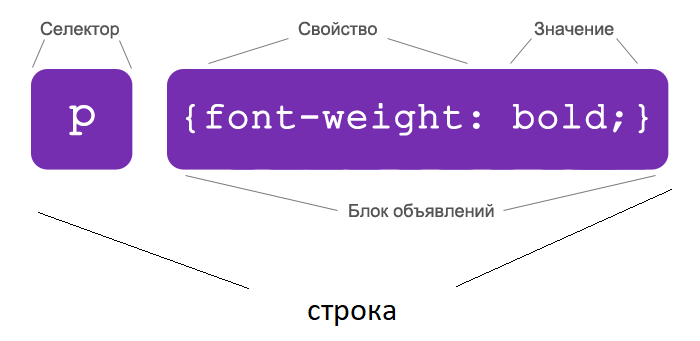
Существует три метода для применения таблицы стилей к документу HTML:

* ***Встроенный (Inline).*** Этот метод позволяет взять любой тег HTML и добавить к нему стиль. Использование встроенного метода предоставляет максимальный контроль над всеми свойствами Web-страницы. Предположим, вы хотите задать внешний вид отдельного абзаца. Вы можете просто добавить атрибут style к тегу абзаца, и броузер отобразит этот абзац с помощью параметров стиля, добавленного в код.
* ***Внедренный (Embedded).*** Внедрение позволяет контролировать всю страницу HTML. При использовании тега <style>, помещенного внутри раздела <head> страницы HTML, в код вставляются детализированные атрибуты стиля, которые будут применяться ко всей странице.
* ***Связанный (Linked или External).*** Связанная таблица стилей - мощный инструмент, который позволяет создавать образцы стилей ***(master styles)***, которые можно затем применять ко всему узлу. Основной документ таблицы стилей (расширение .css) создается Web-дизайнером. Этот документ содержит стили, которые будут едиными для всего Web-узла (неважно, содержит одну страницу или тысячи страниц). Любая страница, связанная с этим документом, будет использовать указанные стили.

***Синтаксис таблицы стилей***

Таблицы стилей строятся в соответствии с определенным порядком (синтаксисом), в противном случае они не могут нормально работать.  
Синтаксис всех методов, используемых для применения стилей к документа HTML, практически одинаков. Таблицы стилей составляются из определенных частей. Эти части включают следующие элементы:

* ***Селектор (Selector).*** Селектор является элементом, к которому будут применяться назначаемые вами атрибуты. Это может быть просто тег типа заголовка **(H1)** или абзаца **(Р)**. Таблицы стилей позволяют использовать различные объекты, включая классы, которые будут кратко обсуждаться далее.
* ***Свойство (Property).*** Свойство определяет указатель. Например, если в качестве указателя выбран абзац, вы можете включить свойства, определяющие этот указатель. В свойства входят такие элементы, как поля, шрифты и фоновые изображения. В таблицах стилей существует много свойств, которые можно использовать для того, чтобы определить указатель.
* ***Значение (Value).*** Значения определяют свойства. Предположим, что у вас есть заголовок первого уровня **H1** (указатель) и вы включаете свойство **type family**(семейство шрифта). Шрифт, который на самом деле будет применен к указанному фрагменту, задается значением этого свойства.
* ***Блок объявлений (Declaration).*** Свойства и значения объединяются, образуя описание.
* ***Строка (Rule).*** Селектор и блок объявлений образуют строку



**24. Система доменных имен DNS.**

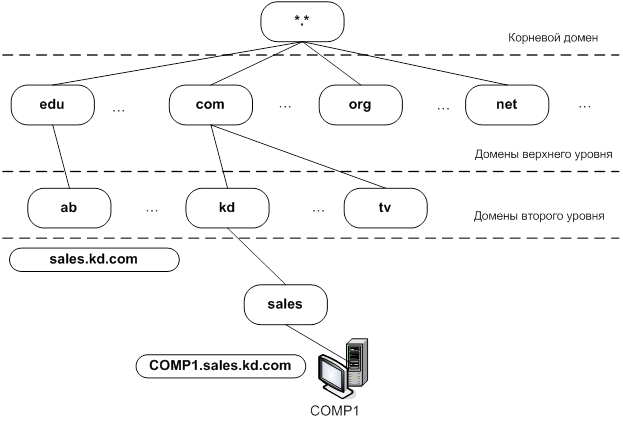
*DNS (Domain Name System  — система доменных имён)* — распределённая система преобразования имени хоста (компьютера или другого сетевого устройства) в IP адрес. DNS работает в сетях TCP/IP. Как частный случай, DNS может хранить и обрабатывать и обратные запросы, определения имени хоста по его IP (PTR-записи).

***Корневой домен обозначается точкой («.»).*** Он представляет собой верхний уровень DNS, остальное пространство имен располагается ниже.

***На следующем уровне под корневым доменом располагаются домены первого уровня***, включая несколько основных (generic) доменных имен (com, edu, mil, net, org и т. д.), около двухсот сокращений названий стран, несколько доменов, которые были введены позже (biz, info, pro и т. д.).

***Под доменами верхнего уровня расположены домены второго уровня***, которые обычно относятся к названиям компаний и должны быть зарегистрированы властями Интернета.

***Поддомены обычно относятся к отделам или подразделениям в пределах компании.*** Эти поддомены регистрируются и управляются с DNS-серверов, которые содержат информацию о доменах второго уровня.



Поскольку DNS использует иерархическое пространство имен, то достаточно просто сконфигурировать его как распределенную базу данных. Прежде чем в Интернете была реализована доменная система имен, вся информация, необходимая для разрешения имен, хранилась в единственном файле. *Поскольку количество хостов в Интернете очень сильно увеличилось, то управление одним файлом стало непрактичным*. Была разработана система DNS, использующая распределенную базу данных. Применение распределенной базы данных означает, что информация DNS хранится на многих компьютерах во всем мире (в случае Интернета) и повсюду в корпоративной сети (в случае внутренней сети). Каждый DNS-сервер обслуживает только одну маленькую часть базы данных DNS. Вся база данных разделена на зонные файлы на основе имен доменов. Зонные файлы распределены между несколькими серверами. К примеру, существует около дюжины серверов, которые содержат зонные файлы для корневого домена. Они хранят информацию о DNS-cepверах, которые несут зонную информацию для доменов высшего уровня. Корневые серверы не содержат всю информацию о доменах высшего уровня, но они знают, какие серверы имеют эту информацию.

*DNS-серверы, хранящие информацию о доменах высшего уровня, содержат также информацию о том, на каких серверах находятся зонные файлы для доменов следующего уровня*. Например, сервер может содержать зонные файлы для домена com, то есть этот сервер знает обо всех доменах второго уровня, которые зарегистрированы с доменом com, но он может не знать отдельные детали о домене второго уровня. Сервер домена высшего уровня знает, какой компьютер на следующем уровне содержит детали, касающиеся домена второго уровня, и так продолжается до самого низа пространства имен DNS. Сервер, ответственный за домен com, может иметь домен kd, зарегистрированный как домен второго уровня. Этот сервер может передавать любые запросы на информацию о домене kd на сервер, который содержит зонные файлы для kd.com.

Использование метода распределенной базы данных означает, что никакому серверу в Интернете не требуется иметь всю информацию DNS. Большинство серверов хранят информацию о некоторой части дерева, но когда приходит запрос, который они не могут выполнить, им известно, какой DNS-сервер хранит необходимую информацию. DNS-серверы используют делегированные записи, ретрансляторы (forwarders) и корневые ссылки для определения того, какой DNS-сервер имеет необходимую информацию.

**25. Графика в web-дизайне.**

## Векторная и растровая

Компьютерные изображения можно разделить на две группы:

* Растровая графика
* Векторная графика

Растровые изображения - это изображения, состоящие из прямоугольного набора точек, например, фотографии, отсканированные рисунки и т.д. Для работы с растровой графикой я предлагаю использовать **Adobe Photoshop**. (Что бы перейти на мой учебник по Adobe Photoshop 5.5 [[нажмите здесь](http://economist.rudn.ru/files/web-studio/docs/misc/www.cl.spb.ru/chtushev/design/aps/intro.htm)])

В векторной графике изображение состоит не из точек, а из набора линий, дуг, прямоугольников и т.п. Это очень удобно, такие изображения легко редактировать. Каждую фигуру можно передвинуть, изменить, развернуть, не затронув других. Кстати, в большинство векторных изображений можно включать растровые куски. Для работы с векторной графикой я предлагаю использовать **Adobe Illustrator**. (Что бы перейти на мой учебник по Adobe Illustrator 8.0 [[нажмите здесь](http://economist.rudn.ru/files/web-studio/docs/misc/www.cl.spb.ru/chtushev/design/ia/intro.htm)])

Если хотите нарисовать что-то своими руками, а не обработать готовое изображение, то обычно лучше использовать векторное изображение. Потом это изображение можно растрировать и доработать его в растровом формате.

Работая с растровой графикой, можно добиться многих вещей, которые пока не доступны в векторной. Например, разные спецэффекты. Засунуть фотографии  или подобные им изображения в векторный формат вам тоже не удастся.

Перевести векторный рисунок в растровый очень просто. В векторных редакторах можно сохранить рисунки в растровых форматах, выбрав такой формат в меню **Save As...**, или импортировать его в векторный формат, что, на самом деле, одно и тоже. А вот сделать из растрового рисунка векторный, практически нельзя. Попробуйте описать лицо человека в виде набора линий, кривых, кругов. Естественно, вам это не удастся, а если и удастся, то это будет похоже не на фотографию, а на рисунок, сделанный от руки.

К сожалению, пока в Интернет на Веб-страницах используется только растровая графика (в основном GIF и JPEG форматы). В связи с этим, если вы нарисовали хорошую картинку в векторном формате, то вам придется перевести ее в растровый. Ну что поделаешь, мне это тоже не нравится.

## GIF и JPEG форматы

Как я уже сказал, в Интернете для картинок в основном используются 2 формата. Это файлы \*.GIF и \*.JPG. Все они сжимают (как бы архивируют) изображение. Но каждый приспособлен к определенной цели. Какой из них выбрать? Это очень важно, так как от формата зависит размер файла и его качество (часто приходится делать тяжелый выбор между размером файла и качеством изображения). Помните, как долго загружаются некоторые картинки из плохо продуманных сайтов? Иногда по несколько минут! И что вам хотелось тогда сделать: разбить клавиатуру, сбросить монитор на пол или просто покинуть этот ....... сайт? Конечно, чем меньше размер картинки, тем меньше размер файла, тем быстрее она грузится. Но иногда можно засунуть очень крупную картинку в очень мелкий файл. Рассмотрим форматы файлов:

**Формат GIF**  (расширение **.gif**)

* Может содержать изображение до 256 цветов. Но число цветов можно уменьшить для уменьшения размера файла.
* Хорошо сжимает контрастные и несильно заполненные изображения с малым числом цветов. Например, чертежи, рисунки.
* Может содержать мультипликацию, т.е. несколько картинок, которые выдаются через указанные промежутки времени. Для редактирования таких многокадровых GIF-ов можно использовать программу Adobe Image Ready. (Чтобы перейти на мой учебник по Adobe Image Ready 2.0 [[нажмите здесь](http://economist.rudn.ru/files/web-studio/docs/misc/www.cl.spb.ru/chtushev/design/air/intro.htm)])
* Может содержать "прозрачные" цвета, т.е. сквозь рисунок может просвечивать задний план.

**Формат JPEG** (расширение **.jpg** или **.jpeg**)

* Хорошо сжимает изображение с плавными цветовыми переходами. Например: портреты, пейзажи, фотографии.
* Можно установить сжатие с заданной потерей качества. Таким образом, можно выбирать соотношение: размер файла / качество.

Давайте рассмотрим это на примерах:

|  |  |
| --- | --- |
| **GIF** | **JPEG** |
| Рисунок с резкими цветовыми переходами и границами: | |
| Хороший выбор  Размер 3.5 КБ | Плохой выбор  Размер 6.5 КБ Размазанность на резких цветовых переходах |
| Слабо заполненный рисунок с резкими границами | |
| Хорошее качество  Размер 2.5 КБ | Все-таки добились хорошего качаства, но жутко проиграли в размере.  Размер 14 КБ Для такого рисунка можно добиться хорошего качества и в формате JPEG, но это сильно увеличит его размер |
| А вот для фотографий, обычно выгоднее использовать JPEG-формат: | |
| Плохой выбор  Размер 21 КБ Размер GIF-файла сильно зависит от числа разрешенных цветов, а их уменьшение может привести к появлению полос на плавных переходах. Иногда может даже не хватить 256 цветов. | Хороший выбор  Размер 9 КБ |
| Чем жертвуем при уменьшении размера файлов: | |
| Для уменьшения размера, жертвуем числом цветов  Размер 10 КБ Качество регулируется путем задания числа цветов | При уменьшении размера возрастает размытость фотографии  Размер 4.5 КБ Здесь при регулировании качества регулируется размытие изображения. |
| Многокадровые изображения (Анимированные GIF-ы) | |
| Пример анимации  Размер 8 КБ | JPEG - так не умеет |

Упрощенный вывод можно сделать следующий: GIF-лучше для картинок с ограниченным числом цветов (например, для логотипов), а JPEG-для картинок с неограниченным числом цветов и плавными переходами (например, для фотографий). Чем сложнее изображение, тем больше размер файла.

При подготовке графики для Интернет иногда бывает даже не ясно, в каком формате лучше сохранять рисунок, и приходиться экспериментировать, задавая разные форматы и параметры файла. Какие возможности предоставляет Adobe Photoshop 5.5 для оптимального сохранения графики, вы можете прочитать [[здесь](http://economist.rudn.ru/files/web-studio/docs/misc/www.cl.spb.ru/chtushev/design/aps/saveforweb.htm)].