# ИНФОРМАТИЗАЦИЯ. ИНФОРМАЦИОННОЕ ОБЩЕСТВО

Информатизация – процесс создания, развития и всеобщего применения информационных средств и технологий, обеспечивающих достижения уровня информированности всех членов общества, необходимого и достаточного для кардинального улучшения качества труда и условий жизни в обществе.

*Признаки информационного общества:*

* Большинство работающих (около 80%) занято в информационной сфере, то есть в сфере производства информации и информационных услуг
* Обеспечены техническая, технологическая и правовая возможности доступа любому члену общества практически в любой точке территории и приемлемое время к нужной ему информации
* Информация становится важнейшим стратегическим ресурсом общества

# ИНФОРМАТИКА И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

*Информатика возникла и развивалась как прикладная наука, в функции которой входила разработка:*

* методов и правил рационального проектирования устройств и систем обработки информации
* технологии использования этих устройств и систем для решения научных и практических задач
* методов взаимодействия человека с этими устройствами и системами

Информатика – фундаментальная естественная наука, изучающая структуру и общее свойство информации, а также вопросы, связанные с процессами сбора, хранения, поиска, передачи, переработки, преобразования и использования информации в различных сферах человеческой деятельности.

*Информатика как отрасль народного хозяйства состоит из следующих частей:*

* Производство технических средств обработки и передачи информации
* Обработка информации
* Производство и реализация программных средств и систем

Информационные технологии – совокупность методов и средств получения и использования информации на базе коммуникационной и вычислительной техники и широкого применения математических методов.

Технология – это совокупность знаний о способах и средствах проведения производственных процессов, при которых происходит качественное изменение обрабатываемых объектов.

*Свойства информационных технологий:*

* Предметом обработки являются данные
* Целью процесса является получение информации
* Средствами осуществления процесса являются программные, аппаратные и программно-аппаратные вычисление комплексы
* Критериями оптимальности процесса являются своевременность доставки информации, её надёжность, достоверность и адекватность

# ИНФОРМАЦИЯ И ФОРМЫ ЕЁ ПРЕДСТАВЛЕНИЯ

В рамках науки информация является первичным и неопределяемым понятием, оно предполагает наличие материального носителя информации, источника, приёмника и канала связи между источником и приёмником.

Информация – первоначальные сведения, передаваемые людьми устным, письменным или другим способом.

С середины 20-го века, информация – общенаучное понятие, включающее обмен сведениями между людьми, человеком и автоматом, автоматом и автоматом.

Обмен сигналами в животном и растительном мире, передача признаков от клетки к клетке, от организма к организму – одно из основных понятий кибернетики.

В технике информация – это сведения, являющиеся объектом хранения, передачи и преобразования.

В философии информация – один из атрибутов материи, отражающий её структуру. Сигнал представляет собой любой процесс, несущий информацию.

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

Данные – это информация, представленная в формализованном виде и предназначенная для обработки её техническими средствами.

Сигнал называется непрерывным, если его параметр в заданных пределах может принимать любые промежуточные значения.

Сигнал называется дискретными, если его параметр в заданных пределах может принимать отдельные фиксированные значения.

Аналоговые – непрерывные, цифровые – дискретные.

ЭВМ (электронно-вычислительная машина) – устройство преобразования информации посредством выполнения управляемой программой последовательности операций.

Процесс преобразования информации часто требует представлять буквы одного алфавита средствами другого алфавита. Такое представление называется кодированием. Процесс обратного преобразования информации относительно раннее выполненного кодирования называется декодированием.



Схема передачи информации.

# ИНФОРМАЦИОННЫЕ ПРОЦЕССЫ

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

* визуальную – передаваемую видимыми образами.
* аудиальную – звуками.
* тактильную – ощущениями.
* органолептическую – запахами и вкусами.
* машинную – выдаваемую и воспринимаемую средствами вычислительной техники.

Восприятие информации – процесс преобразования сведений, поступающих в техническую систему или живой организм из внешнего мира в форму, пригодную для дальнейшего использования.

Сбор информации – деятельность субъекта, в ходе которой он получает сведения об интересующем его объекте.

Обмен информацией – процесс, в ходе которого источник информации её передаёт, а получатель – принимает.

Если в передаваемых сообщениях обнаружены ошибки, то организуется повторная передача этой информации.

В результате обмена информацией между источником и получателем устанавливается информационный баланс. В идеальном случае получатель будет располагать той же информацией, что и источник.

Источниками информации могут быть любые объекты реального мира, обладающие определёнными свойствами и способностями. Если объект относится к неживой природе, то он вырабатывает сигналы, непосредственно отражающие его свойства. Если объектом-источником является человек, то вырабатываемые им сигналы могут не только отражать его свойства, но и соответствовать тем знакам, которые человек вырабатывает с целью обмена информацией.

Процесс формирования исходного несистематизированного массива информации называется накоплением информации.

Хранение информации – это процесс поддержания информации в виде, обеспечивающем выдачу данных по запросам конечных пользователей в установленные сроки.

Обработка информации – упорядоченный процесс её преобразования в соответствии с алгоритмом решения задачи.

Важнейшей характеристикой методов обработки являются адекватность обработки информации.

После решения задачи обработки информации результат должен быть выдан конечным пользователям в требуемом виде. Эта операция реализуется в ходе решения задачи выдачи информации.

# СВОЙСТВА ИНФОРМАЦИИ

* Релевантность – способность информации соответствовать нуждам потребителя.
* Полнота – свойство информации исчерпывающие для данного потребителя характеризовать отражаемый объект или процесс. Полнота является важнейшей характеристикой качества информации и определяет достаточность данных для принятия решений и для создания новых данных на основе имеющихся.
* Объективность и субъективность информации – понятие объективности является относительным, более объективной принято считать ту информацию, которой методы её получения и обработки вносят меньше субъективных элементов.
* Достоверность – свойство информации не иметь скрытых ошибок. Данные возникают в момент регистрации сигналов, но наряду с полезным сигналом регистрируются и посторонние сигналы, так называемые шумы. Чем более чётко зарегистрирован полезный сигнал и меньше шумов, тем более достоверна полученная информация.
* Адекватность – свойство информации однозначно соответствовать отображаемому объекту или явлению. Неадекватная информация может образовываться при создании новой информации на основе неполных или недостоверных данных. В то же время, используя для обработки полных и достоверных данных неадекватные методы, можно получить неадекватную информацию.
* Доступность – свойство информации, характеризующее возможность её получения данным потребителем.
* Актуальность – способность информации соответствовать нуждам потребителя в текущий момент времени. Поскольку информационные процессы растянуты во времени, то достоверная и адекватная, но устаревшая информация может приводить к ошибочным решениям.
* Защищённость – свойство, характеризующее невозможность несанкционированного использования или изменения информации.
* Эргономичность – свойство, характеризующее удобство формы или объёма информации с точки зрения данного потребителя.

# ИНФОРМАЦИОННЫЕ СИСТЕМЫ

Система – множество элементов, находящихся в отношениях и связях друг с другом, образующих определённую целостность, единство.

*Свойства системы:* делимость и целостность.

Делимость означает, что систему можно представить, состоящей из относительно самостоятельных частей – подсистем, каждая из которых может рассматриваться как отдельная система. Возможность выделения подсистем или иначе декомпозиция системы упрощает её анализ, разработку, внедрение и эксплуатацию.

Целостность указывает на согласованность цели функционирования всей системы с целями функционирования её подсистем и элементов.

Информационная система – взаимосвязанная совокупность средств, методов и персонала, используемых для сбора, хранения, обработки, поиска и выдачи информации в интересах достижения поставленной цели.

Существуют более узкие толкования этого термина, например имеется ввиду аппаратная часть компьютера, программа, набор технических средств и программ и так далее.

Структуру информационной системы составляет совокупность её отдельных частей (подсистем).

*Структура информационной системы:*

* Информационное обеспечение – система классификации и кодирования информации, система документации, схема информационных поток, циркулирующих в организации, методология построения баз данных.
* Техническое обеспечение – компьютеры различных моделей, устройства сбора, накопления, обработки, передачи и вывода информации, устройства передачи данных и линии связи, оргтехника, эксплуатационные материалы и соответствующая документация на перечисленные устройства.
* Математическое и программное обеспечение – совокупность математических методов, моделей, алгоритмов и программ, используемых в информационной системе.
* Организационное обеспечение – совокупность методов и средств, регламентирующих взаимодействие работников с техническими средствами и между собой в процессе разработки и эксплуатации информационной системы.
* Правовое обеспечение – совокупность правовых норм, определяющих создание, юридический статус и функционирование информационной системы, регламентирующих порядок получения, преобразования и использования информации.

# КЛАССИФИКАЦИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

КЛАССИФИКАЦИЯ ПО ПРИЗНАКУ СТРУКТУРИРОВАННОСТИ ЗАДАЧ

*Различают три типа задач, для которых создаются информационные системы:*

* Структурированные (формализуемые)
* Не структурированные (не формализуемые)
* Частично структурированные

Формальное описание – математическое и алгоритмическое описание решаемой задачи.

Структурированная задача – задача, где известны все её элементы и взаимосвязи между ними, возможна полная автоматизация решения задачи с помощью информационной системы.

Не структурированная задача – задача, в которой невозможно выделить элементы и установить между ними связь, возможности использования информационных систем в этом случае невелики.

Большинство задач являются частично структурированными.

КЛАССИФИКАЦИЯ ПО ФУНКЦИОНАЛЬНОМУ ПРИЗНАКУ

*Функциональный признак определяет назначение системы, её основные цели, задачи и функции:*

* Системы маркетинга – исследование рынка и прогнозирование продаж, управление продажами, рекомендации по производству новой продукции, анализ и установление цены, учёт заказов.
* Производственные системы – планирование объёмов работ и разработка календарных планов, оперативный контроль и управление производством, анализ работы оборудования, участие в формировании заказов поставщикам, управление запасами.
* Финансовые и учётные системы – управление заказами, управление кредитной политикой, разработка финансового плана, финансовый анализ и прогнозирование, бухгалтерский учёт и расчёт заработной платы.
* Системы кадров – анализ и прогнозирование потребности в трудовых ресурсах, ведение архивов, записей о персонале, анализ и планирование подготовки кадров.

КЛАССИФИКАЦИЯ ПО ХАРАКТЕРУ ИСПОЛЬЗОВАНИЯ ИНФОРМАЦИИ

* Информационно-поисковые системы – производят ввод, систематизацию, хранение, выдачу информации по запросу пользователя без сложных преобразований данных.
* Информационно-решающие системы – осуществляют сложную переработку информации. Разделяются на: управляющие информационные системы – вырабатывают информацию, на основании которой человек принимает решения, для них характерно решение расчётных задач и обработка больших массивов данных; советующие информационные системы – вырабатывают информацию,

которая принимается человеком к сведению и не превращается немедленно в серию конкретных действий, характерна обработка знаний, а не данных.

КЛАССИФИКАЦИЯ ПО СФЕРЕ ПРИМЕНЕНИЯ ИНФОРМАЦИИ

* Научные исследования – автоматизация деятельности научных работников, анализ статистической информации, управление экспериментами.
* Автоматизированное проектирование – автоматизация работы инженеров- проектировщиков, разработка новых изделий и технологий их производства, инженерные расчёты, создание графической документации (чертежей, схем, планов), моделирование проектируемых объектов, создание управляющих программ для станков с числовым программным управлением.
* Организационное управление – автоматизация функций управленческого персонала, управление промышленными предприятиями, банками, страховыми компаниями и прочее.
* Управление технологическими процессами – автоматизация различных технологических процессов металлургии, машиностроения, энергетики и других областях.
* Интегрированные информационные системы – автоматизация всех процессов на предприятии от проектирования до сбыта продукции.

# ПРЕДСТАВЛЕНИЕ ЧИСЛОВОЙ ИНФОРМАЦИИ

СИСТЕМЫ СЧИСЛЕНИЯ

Системой счисления называется способ наименования и записи чисел с помощью некоторого алфавита символов, называемых цифрами.

Непозиционными системами называются такие системы счисления, в которых каждый символ сохраняет своё значение независимо от места его положения в числе (например, римская).

Система счисления называется позиционной, если одна и та же цифра имеет различное значение, определяющееся позицией цифры в последовательности цифр, изображающей число. Это значение меняется в однозначной зависимости от позиции, занимаемой цифрой, по некоторому закону.

Количество P различных цифр, употребляемых в позиционной системе, определяет название системы счисления и называется основанием системы счисления.

Любое число N в позиционной системе счисления с основанием P может быть представлено в виде многочлена по степеням P.

𝑁𝑝 = 𝑎𝑘𝑝𝑘 + 𝑎𝑘−1𝑝𝑘−1 + ⋯ + 𝑎1𝑝1 + 𝑎0𝑝0 + 𝑎−1𝑝−1 + ⋯ 0 ≤ 𝑎𝑘 ≤ 𝑝 − 1, 𝑘 > 0, 𝑝 > 0

Число при этом представляется в виде последовательности чисел. Запятая отделяет целую часть числа от дробной части.

𝑁𝑝 = 𝑎𝑘𝑎𝑘−1 … 𝑎1𝑎0, 𝑎−1𝑎−2 …

Наиболее важные системы счисления

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ДВОИЧНАЯ** | **ВОСЬМИРИЧНАЯ** | | **ДЕСЯТИЧНАЯ** | **ШЕСТНАДЦАТИРИЧНАЯ** | |
|  |  | Триады |  |  | Тетрады |
| 0 | 0 | 000 | 0 | 0 | 0000 |
| 1 | 1 | 001 | 1 | 1 | 0001 |
|  | 2 | 010 | 2 | 2 | 0010 |
|  | 3 | 011 | 3 | 3 | 0011 |
|  | 4 | 100 | 4 | 4 | 0100 |
|  | 5 | 101 | 5 | 5 | 0101 |
|  | 6 | 110 | 6 | 6 | 0110 |
|  | 7 | 111 | 7 | 7 | 0111 |
|  |  |  | 8 | 8 | 1000 |
|  |  |  | 9 | 9 | 1001 |
|  |  |  |  | A | 1010 |
|  |  |  |  | B | 1011 |
|  |  |  |  | C | 1100 |
|  |  |  |  | D | 1101 |
|  |  |  |  | E | 1110 |
|  |  |  |  | F | 1111 |

# ПЕРЕВОД ЧИСЕЛ ИЗ ОДНОЙ СИСТЕМЫ СЧИСЛЕНИЯ В ДРУГУЮ

ПЕРЕВОД ИЗ ДЕСЯТИЧНОЙ СИСТЕМЫ В ДРУГУЮ. ПЕРЕВОД ЦЕЛЫХ ЧИСЕЛ

Дано целое положительное число С, требуется определить его коэффициенты в разложении по степеням двойки. Разделим обе части равенства (1) на два.

𝐶 = 𝑎𝑘2𝑘 + 𝑎𝑘−12𝑘−1 + ⋯ + 𝑎121 + 𝑎020 (1)

𝐶 + 𝑃0 = 𝑎

2𝑘−1 + 𝑎

2𝑘−2 + ⋯ + 𝑎

20 + 𝑎 /2

0 2 𝑘

𝑘−1 1 0

После деления получим частное C0 и остаток P0, который равен либо нулю, либо единице. Из этого равенства очевидно, P0 = a0, то есть цифра, стоящая в младшем разряде двоичного представления числа С, совпадает с остатком от деления С на два, в итоге получим формулу (2).

𝐶0 = 𝑎𝑘2𝑘−1 + 𝑎𝑘−12𝑘−2 + ⋯ + 𝑎120 (2)

𝑃2

𝐶 + = 𝑎

2𝑘−2 + 𝑎

2𝑘−3 + ⋯ + 𝑎

20 + 𝑎 /2

1 2 𝑘

𝑘−1 2 1

Аналогично разделим обе части равенства (2) на двойку. Очевидно, P1 = a1, далее повторяем операцию деления до тех пор, пока не получим частное единица. Оно является единицей старшего разряда числа.

Перевод целых десятичных чисел в восьмеричную, шестнадцатеричную и двоичную системы осуществляется последовательным делением десятичного числа на основание той системы счисления, в которую оно переводится до тех пор, пока не получится частное, меньшее этого основания. Число в новой системе записывается в виде остатков деления, начиная с последнего.

ПЕРЕВОД ПРАВИЛЬНЫХ ДРОБЕЙ

Дано десятичное число D (0 < D < 1), требуется определить коэффициенты в разложении его по степеням двойки.

𝐷 = 𝑎1𝑝−1 + 𝑎2𝑝−2 + ⋯ + 𝑎𝑘𝑝−𝑘 + ⋯ (3)

Умножим части уравнения (3) на два. Результат можно представить в виде суммы целого числа С1 равного нулю или единице и новой дробной части D1.

𝐶1 + 𝐷1 = 𝑎1𝑝0 + 𝑎2𝑝−1 + ⋯ + 𝑎𝑘𝑝−𝑘+1 + ⋯

Очевидно, а1 = С1, а D1 равняется оставшейся части. Умножим D1 на два, получим уравнение следующего вида:

𝐶2 + 𝐷2 = 𝑎2𝑝0 + 𝑎3𝑝−1 + ⋯ + 𝑎𝑘𝑝−𝑘+2 + ⋯

Здесь а2 = С2. Количество знаков в двоичном представлении правильной дроби берётся из соображений требуемой точности.

Для перевода правильной десятичной дроби в другую систему, эту дробь надо последовательно умножать на основание той системы, в которую она переводится, при этом умножаются только дробные части. Дробь в новой системе записывается в виде целых частей произведений, начиная с первого.

Для перевода неправильной десятичной дроби в систему счисления с недесятичным основанием, необходимо отдельно перевести целую часть и отдельно дробную часть.

ПЕРЕВОД В ДЕСЯТИЧНУЮ СИСТЕМУ СЧИСЛЕНИЯ

Перевод чисел в десятичную систему осуществляется путём составления степенного ряда с основанием той системы, из которой число переводится, затем подсчитывается значение суммы.

Для перевода восьмеричного и шестнадцатеричного числа в двоичную форму, достаточно заменить каждую цифру этого числа соответствующим трёхразрядным двоичным числом (триадой) или четырёхразрядным двоичным числом (тетрадой), при этом отбрасывают ненужные нули старших и младших разрядов.

# ПЕРЕВОД ИЗ ДВОИЧНОЙ В ВОСЬМЕРИЧНУЮ И ШЕСТАНДЦАТЕРИЧНУЮ

Для перехода из двоичной к восьмеричной или шестнадцатеричной системе поступают следующим образом: двигаясь от точки влево или вправо разбивают двоичное число на группы по три (четыре) разряда, дополняя, при необходимости, нулями крайнюю левую и правую группы. Затем триаду (тетраду) заменяют соответствующей восьмеричной (шестнадцатеричной) цифрой.

# ПЕРЕВОД ИЗ ШЕСТНАДЦАТЕРИЧНОЙ В ВОСЬМЕРИЧНУЮ

Перевод из шестнадцатеричной в восьмеричную систему и обратно осуществляется через двоичную систему с помощью триад и тетрад.

# ДВОИЧНАЯ АРИФМЕТИКА

|  |  |  |
| --- | --- | --- |
| **СЛОЖЕНИЕ** | **ВЫЧИТАНИЕ** | **УМНОЖЕНИЕ** |
| 0 + 0 = 0 | 0 – 0 = 0 | 0 \* 0 = 0 |
| 0 + 1 = 1 | 1 – 0 = 1 | 0 \* 1 = 0 |
| 1 + 0 = 1 | 1 – 1 = 0 | 1 \* 0 = 0 |
| 1 + 1 = 10 | 10 – 1 = 1 | 1 \* 1 = 1 |

При сложении двоичных чисел в каждом разряде производится сложение цифр слагаемых и перенос из соседнего младшего разряда, если он имеется. При этом необходимо учитывать, что 1 + 1 дают ноль в данном разряде и единицу переноса в следующем.

При вычитании двоичных чисел в данном разряде при необходимости занимается единица из старшего разряда. Эта занимаемая единица равна двум единицам данного разряда.

Умножение двоичных чисел производится по тем же правилам, что и для десятичных, с помощью таблиц двоичного умножения и сложения.

Деление двоичных чисел производится по тем же правилам, что и для десятичных. При этом используются таблицы двоичного умножения и вычитания.

# ФОРМЫ ПРЕДСТАВЛЕНИЯ ЧИСЕЛ В ПАМЯТИ ЭВМ

Под знак числа отводится один разряд (нуль соответствует плюсу, единица соответствует минусу). По способу фиксации положения запятой различают две формы записи числа: запись с фиксированной запятой и запись с плавающей запятой. При записи числа с фиксированной запятой отводится постоянное число разрядов для целой и дробной частей числа.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|  | 2n-1 |  |  |  |  | 21 | 20 | 2-1 |  |  |  |  |  |  | 2-m |
| Знак | Целая часть (n разрядов) | | | | | | | Дробная часть (m разрядов) | | | | | | | |

К достоинствам использования чисел с фиксированной запятой относится простота выполнения арифметических операций и высокая точность изображения числа. Недостатком – не большой диапазон представления чисел.

Для представления числа с плавающей запятой используется так называемая полулогарифмическая форма записи числа.

N = ±mq±p

Здесь q – основание системы счисления, p – порядок числа, m – мантисса числа. Если мантисса числа находится в диапазоне следующего вида: 1 ≤ |𝑚| ≤ 1, говорят о

𝑞

нормализованной форме записи числа.

Для представления числа в памяти ЭВМ в этом случае выделяют группы разрядов для изображения знака числа, знака порядка, порядка числа и мантиссы. Число изображается в нормализованной форме.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Знак m | Знак p | Порядок (4 разряда) | | | | Мантисса (10 разрядов) | | | | | | | | | |

Для записи чисел также используют 32-х разрядный формат (машинное слово) и 64-х разрядный формат (двойное слово). В частности, при представлении числа с плавающей запятой в формате машинного слова порядок занимает 7 разрядов, мантисса – 23. При представлении в формате двойного слова (представление с двойной точностью) под порядок отводится 7 разрядов, под мантиссу – 55. Числа с плавающей запятой позволяют увеличить диапазон обрабатываемых чисел, но при этом точность изображения чисел определяется разрядами мантиссы и уменьшается по сравнению с числами с фиксированной запятой.

# КОДИРОВАНИЕ ЧИСЕЛ

В ЭВМ в целях упрощения выполнения арифметических операций применяют специальные коды для представления чисел. Использование кодов позволяет свести операцию вычитания чисел к арифметическому сложению кодов этих чисел, что на аппаратном уровне гораздо легче реализовать. Применяются прямой, обратный и дополнительный коды.

Прямой код используется для представления отрицательных чисел в памяти ЭВМ, а также при умножении и делении.

Обратный и дополнительный коды используются для замены операции вычитания операцией сложения, что упрощает устройство арифметического блока ЭВМ.

Прямой код двоичного числа совпадает по изображению с записью самого числа. Значение знакового разряда для положительных чисел равно нулю, а отрицательных – единица.

Обратный код для положительного числа совпадает с прямым кодом. Для отрицательного числа все цифры числа заменяются на противоположные (1 на 0, 0 на 1), а в знаковый разряд заносится единица.

Дополнительный код положительного числа совпадает с прямым кодом. Для отрицательного числа дополнительный код образуется путём получения обратного кода и добавлением к младшему разряду единицы.

*Пример:* +1101 прямой код 0 0001101, обратный код 0 0001101, дополнительный код 0

0001101; -1101 прямой код 1 0001101, обратный код 1 1110010, дополнительный код

1 1110011.

# МОДИФИЦИРОВАННЫЕ ОБРАТНЫЙ И ДОПОЛНИТЕЛЬНЫЙ КОД

При переполнении разрядной сетки происходит перенос единицы в знаковый разряд. В этом случае положительное число получившиеся в результате арифметической операции может восприниматься как отрицательное, так как в знаковом разряде у нас появляется единица. Для обнаружения переполнения разрядной сетки вводятся модифицированные коды.

Модифицированный обратный код – под знак числа отводится не один, а два разряда. Форма записи чисел в модифицированном обратном коде выглядит следующим образом:

* Для положительного числа: X = X1X2X3…Xa; Xм = 00 X1X2X3…Xa;

об

* Для отрицательного числа: X = X1X2X3…Xa; Xм = 11 X1X2X3…Xa;

об

Любая другая комбинация (01, 10), получившаяся в знаке числа, означает, что случилось переполнение разрядной сетки. Сложение чисел в модифицированном обратном коде происходит также, как и в обычном обратном коде.

Х = 00 101011, Y = 00 110100; X + Y = 01 011111

В ЭВМ в процессе работы оба знаковых разряда сравниваются. В случае появления признака переполнения вычисления останавливаются.

Модифицированный дополнительный код – также рассматриваются два знаковых разряда, в остальном совпадает с обычным дополнительным кодом. Выглядит следующим образом:

* Для положительного числа: X = X1X2X3…Xa; Xмдоп = 00 X1X2X3…Xa;
* Для отрицательного числа: X = X1X2X3…Xa; Xмдоп = 11 X1X2X3…Xa;

*Пример:* даны два числа X = 101001 и Y = -11010. Сложить их в дополнительном и модифицированном дополнительном кодах.

*Решение:*

1. Переведём Х и Y в дополнительный и модифицированный дополнительный код: Обычная запись: X = +0101001, Y = -0011010

Обратный код: X = 0,0101001, Y = 1,1100101

Дополнительный код: X = 0,0101001, Y = 1,1100101 Модифицированный обратный код: X = 00,101001, Y = 11,100101 Модифицированный дополнительный код: X = 00,101001, Y = 11,100110

1. Выполним сложение:

(X + Y)доп = 0,0001111 (X + Y)мдоп = 00,001111

# АРИФМЕТИЧЕСКИЕ ДЕЙСТВИЯ НАД ЧИСЛАМИ С ПЛАВАЮЩЕЙ ТОЧКОЙ

При сложении чисел с плавающей точкой сначала уравниваются порядки слагаемых, а затем складываются мантиссы. Уравнивание порядков заключается в том, что меньший порядок числа увеличивается до большего, при этом, соответственно, изменяется и мантисса. Мантиссы складываются в одном из модифицированных кодов. Порядком суммы является больший порядок.

Если после сложения результат оказывается ненормализованным, то его нормализуют, изменяя и порядок.

*Пример:* представить числа X = 910 и Y = -3710 в виде нормализованных двоичных чисел с плавающей точкой и сложить. Результат проверить, переведя сумму в десятичную систему.

*Решение:*

1. Переведём числа в двоичную систему счисления и нормализуем: X = 910 = 10012 = (0.1001 \* 10100)2

Y = -3710 = -1001012 = (-0.100101 \* 10110)2

1. Уравниваем порядки чисел:

X = 0.1001 \* 10100 = 0.001001 \* 10110

Y = -0.100101 \* 10110

1. Переводим мантиссы чисел в модифицированный дополнительный код и складываем их:

mxдоп = 00.001001

myдоп = 11.011011

mx+yдоп = 11.100100

1. Переводим результат в прямой код: mx+yпр = 11.011100
2. Представим результат в виде числа с плавающей точкой и нормализуем его: X + Y = -0.0111 \* 10110 = -0.111 \* 10101
3. Таким образом результат сложения: X + Y = (-0.111 \* 10101)2 = -2810

При умножении двух чисел с плавающей точкой их мантиссы перемножаются, а порядки складываются. В арифметико-логическом устройстве ЭВМ операция умножения реализуется в виде следующей последовательности действий:

1. Мантиссы, представленные в прямом коде, перемножаются, при этом установка десятичной точки происходит, как и при нормальном умножении, путём подсчёта количества разрядов после точки в обоих множителях.
2. Порядки складываются с применением обратного или дополнительного кода.
3. Знаки чисел обрабатываются специальной логикой (исключающий элемент).

*Пример:* представить числа X = 510 и Y = -0.37510 в виде нормализованных двоичных чисел с плавающей точкой и перемножить. Результат проверить, переведя произведение в десятичную систему.

*Решение:*

1. Переведём числа в двоичную систему счисления и нормализуем: X = 510 = 1012 = (0.101 \* 1011)2

Y = -0.37510 = -0.0112 = (-0.11 \* 10-1)2

1. Запишем мантиссы в восьмиразрядном прямом коде: mxпр = 0,1010000

myпр = 1,1100000

1. Перемножим мантиссы и определим знак произведения. Таким образом мантисса произведения, записанная в прямом коде, имеет вид:

(mx \* my)пр = 1,0111100

1. Сложим порядки:

pX + pY = 11 + (-1) = 10

1. Таким образом, результат умножения:

X \* Y = (-0.01111 \* 1010)2 = (-0.1111 \* 101)2 = -1.87510

При выполнении операции деления чисел с плавающей точкой, мантиссу делимого делят на мантиссу делителя, а из порядка делимого вычитают порядок делителя. В арифметико-логическом устройстве ЭВМ при выполнении операции деления, деление мантисс и вычитания порядков осуществляется с использованием обратного или дополнительного кодов. Знаки чисел обрабатываются с помощью исключающего ИЛИ.

# АРХИТЕКТУРА ЭВМ

Под архитектурой ЭВМ понимается совокупность общих принципов организации аппаратно-программных средств и их характеристик, определяющая функциональные возможности ЭВМ при решении соответствующих классов задач.

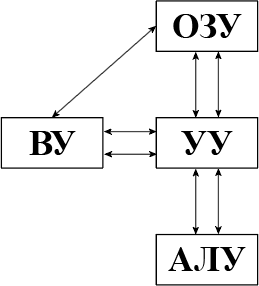
*Компонентами архитектуры являются:*

* Вычислительные и логические возможности (системы команд в формате данных, быстродействие).
* Аппаратные средства (структура ЭВМ, организация памяти, организация ввода/вывода, принципы управления).
* Программное обеспечение (операционная система, языки программирования, прикладное программное обеспечение).

# АППАРАТНЫЕ СРЕДСТВА

*Согласно Фон Нейману ЭВМ должна содержать следующие устройства:*

* Арифметико-логическое устройство (АЛУ), выполняющее арифметические и логические операции.
* Устройство управления (УУ), организующее процесс выполнения программ.
* Запоминающее устройство для хранения программ и данных (ОЗУ).
* Внешнее устройство (ВУ) для ввода и вывода информации.



Процессор объединяет АЛУ и УУ и включает также небольшой объём памяти, называемой кэшем.

Память разделяют на внутреннюю и внешнюю. Внутренняя память состоит из ПЗУ и оперативной памяти (ОЗУ), она составляет большую часть и служит для приёма, хранения и выдачи информации, является энергозависимым, т.е. при выключении питания вся информация теряется.

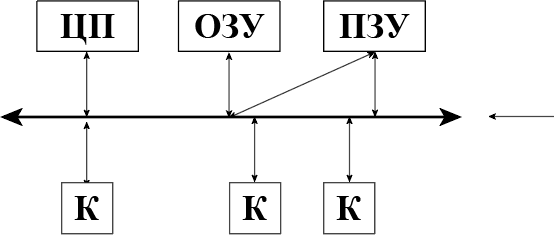
Вторая часть внутренней памяти – постоянное запоминающее устройство (ПЗУ) служит для хранения и выдачи информации. В ПЗУ хранятся часто используемые программы и данные, например, программы тестирования, некоторые программы операционной системы и прочее. При выключении питания, содержимое ПЗУ сохраняется.

Внешнее запоминающее устройство (ВЗУ) используется для долговременного хранения больших объёмов информации.

Устройства ввода/вывода или внешние (периферийные) устройства служат для ввода информации ЭВМ или вывода из неё. К ним относятся мониторы, клавиатуры, мыши, принтеры, сканеры и прочее.

Контроллеры служат для управления внешними устройствами и согласования их с системным интерфейсом.

Системный интерфейс – это конструктивная часть ЭВМ, предназначенная для взаимодействия её устройств и обмена информацией между ними. В персональных компьютерах в качестве системного интерфейса используется системная шина – физическая совокупность проводов.



*В основе открытой архитектуры лежат два основополагающих принципа:*

* Принцип магистральности. Под магистральностью в вычислительной технике подразумевают группу линий передачи информации. Принцип заключается в том, что все модули компьютера подключены к одной и той же магистрали. Для обеспечения принципа магистральности устанавливается единое правило для передачи сигналов, а также правило диалога между отдельными периферийными устройствами.
* Принцип совместимости сверху-вниз. Он означает, что каждая новая модель блока компьютера должна поддерживать функции предыдущей модели.

# ОПЕРАЦИОННАЯ СИСТЕМА

Операционная система – это совокупность программных средств, осуществляющих управление ресурсами ЭВМ, запуск прикладных программ и их взаимодействие с внешними устройствами и другими программами, а также обеспечивающих диалог пользователя с компьютером.

Ресурсом является любой компонент ЭВМ и предоставляемые им возможности:

* Центральный процессор (ЦП)
* Оперативная или внешняя память (ОЗУ или ВП)
* Внешнее устройство (ВУ)
* Программа
* Прочее

Операционная система загружается при включении компьютера. Она предоставляет пользователю удобный способ общения (интерфейс) с вычислительной системой.

Интерфейс может быть программным или пользовательским.

Программный интерфейс – это совокупность средств, обеспечивающих взаимодействие устройств и программ в рамках вычислительной системы.

Пользовательский интерфейс – это программные и аппаратные средства взаимодействия пользователя с программой или ЭВМ.

# ИНТЕГРАЦИЯ ДАННЫХ

Операционная система, в частности Windows, обеспечивает интеграцию в одном документе разных видов данных (текстовых, таблиц, рисунков и прочее) созданных при помощи различных приложений.

Интеграция обеспечивается за счёт передачи данных между программами через буфер обмена, а также с помощью средств связывания и внедрения объектов.

Буфер обмена является областью памяти, используемой для передачи данных. Данные помещаются в буфер обмена, а затем вставляются в нужный документ. При использовании буфера обмена может обеспечиваться конвертирование данных из формата документа источника в формат документа получателя.

Различают два вида объектов: внедрённые и связанные.

Объект является внедрённым, если после его размещения в документе получателя он не зависит от документа источника, то есть изменение этого объекта в документе источнике не приводит к изменениям в документе получателя.

При связывании документ получателя содержит лишь ссылку на источник и при удалении или перемещении источника объект удаляется и у получателя. Но в этом случае документ-получатель имеет меньший размер.

# ЗНАКОВЫЕ СИСТЕМЫ

Под знаком будем понимать материальный предмет, событие или явление, которые служат заместителем другого предмета и используются вместо него в процессе получения, хранения и передачи информации. Знаки являются видом сигнала.

В любом знаке следует различать две стороны: имеется объект, выступающий в роли знака, то есть некоторое материальное выражение; в то же время за знаком скрывается некий объект, замещаемый этим знаком, эта сторона знака называется значение. Существует и третий элемент, связанный с тем, что знаки определяют свойства замещаемого объекта.

Чтобы определить природу знаков вводится понятие денотата и концепта.

Денотата – это объект, обозначаемый знаком. Свойства денотата, выраженное знаком, называется концептом.

*Пример:*

Число π. 3,14 – денотат, отношение длины окружности к радиусу – концепт.

Существуют объекты, которые замещают другие объекты, но в полном смысле знаками считаться не могут. Например, квитанция может выступать в роли заменителя объекта, но при этом никаких свойств объекта она не выражает.

# СИМВОЛЫ И ДИАКРИТИКИ

По степени выраженности смысла знаки делятся на символы и диакритики. Форма символа подчинена стремлению максимально отобразить его значение (геральдические знаки, пиктограммы в общественных местах и прочее).

Диакритики не имеют прямой связи между своей формой и значением (знаки арифметических операций, знаки препинания и прочее).

Чаще всего мы пользуемся не отдельными знаками, а их группами. Знаки, относящиеся к какой-либо одной области, связанные друг с другом и организованные в чёткую структуру, образуют знаковую систему.

Люди используют и внесистемные знаки, однако такие знаки имеют ограниченное применение.

# ЯЗЫК КАК ЗНАКОВАЯ СИСТЕМА

Языки, используемые людьми для общения, называются естественными.

*Языки являются знаковыми системами по следующему признаку:*

* Каждый естественный язык может выступать в двух формах: устной и письменной. Звучащие и написанные слова обозначают что-то и поэтому являются знаками.
* Слова в языке существуют не изолированно, а используются в определённой системе – знаковой системе языка.

Некоторые слова содержат прямую информацию по назначению и особенностях того, что они обозначают. Обычно это составные слова. Слова являются диакритическими знаками.

Из всех знаковых систем естественные языки выделяются тем, что они универсальны, все остальные знаковые системы имеют ограниченное применение. Универсальность естественных языков проявляется в том, что эти знаковые системы позволяют выразить всё то, для чего используются другие знаковые системы.

Искусственные языки – это языки, придуманные людьми с определённой целью.

Естественные языки играют важную роль, потому что обладают рядом полезных свойств, в частности, естественный язык обладает избыточностью. Избыточность проявляется в использовании лишних слов, без которых содержание было бы понятно, а также в грамматических связях.

Другое свойство естественного языка – многозначность. Многие слова могут использоваться, обозначая разные смыслы. Мы понимаем текст, потому что можем оценить, какое из возможных значений слова может быть использовано в данном фрагменте. Многозначность является следствием универсальности естественных языков.

Интерпретация – связь, между словом и предметом, который обозначает слово.

Искусственные языки конструируются с определённой целью – в науке, как и в компьютерных технологиях, искусственные языки служат для формализованного описания объектов и явлений. Эти описания используются для получения новых знаний об исследуемых объектах или явлениях.

СВОЙСТВА ИСКУССТВЕННЫХ ЯЗЫКОВ

* Отсутствует многозначность, свойственная естественным языкам.
* Практически отсутствует возможность перефразирования, то есть нельзя некоторое содержание представить в разных формах.
* Они ориентированы в основном на письменную форму представления.
* В них отсутствует избыточность, свойственная естественным языкам.

Формальный язык – искусственный язык, характеризующийся точными правилами построения выражений и их понимание.

# СЕМИОТИКА

Семиотика – наука, изучающая способы передачи информации, свойства знаков и знаковых систем.

*Семиотика изучает вопросы связи:*

* Знаков друг с другом
* Знаков и замещаемых ими объектов внешнего мира
* Знаков с субъектами, использующими их как инструмент коммутации.

Соответственно, имеется три раздела семиотики:

Синтактика – изучает внутреннее устройство знаковых систем, в частности правила построения сложных знаков из простых.

Семантика – изучает соотношение между знаками и денотатами.

Прагматика – изучает отношение между знаками и людьми, которые их употребляют.

# ФОРМАЛИЗАЦИЯ

Точное описание трёх аспектов семиотики:

Прагматика в значительной степени связанна с психологией и социологией, и поэтому математическими методами практически не описывается.

Семантика изучается в разделе информатики, который называется искусственный интеллект. В некоторых случаях может быть формализована.

Синтактика описывается, в основном, набором правил и поэтому может быть достаточно просто формализована.

# ПРЕДСТАВЛЕНИЕ ТЕКСТОВОЙ ИНФОРМАЦИИ

Таблицы, в которых указывается, какому знаку соответствует какой числовой код, называются таблицами кодировок (в частности, ASCII).

Это система кодирования содержит 2 таблицы кодирования: базовую и расширенную. Базовая таблица содержит коды с 0 по 127, а расширенная с 128 по 255.

Коды с 0 по 31 – управляющие коды. Служат для управления выводом информации.

Коды с 32 по 127 – коды символов английского алфавита, знаков препинания, цифр, арифметических действий и других символов.

Коды с 128 по 255 – символы национальных алфавитов.

Неудобство этой системы в том, что под каждый язык требуется отдельная кодировочная таблица.

Другой стандарт – UNICODE. Здесь для кодирования символов используются 16 разрядов. Соответственно, в одной таблице можно закодировать 65К символов. Недостаток в том, что такой текст занимает вдвое больше места.

ТЕКСТОВЫЕ ФАЙЛЫ

Состоят из строк символов. Каждая строка заканчивается двумя специальными символами: возврат каретки и новая строка. Они не видны при редактировании и просмотре. В основном текстовые файлы имеют расширение \*.txt.

Файлы документы хранят данные в формализованном виде. В документе кроме текста хранятся сведения о делении документа на абзацы, характеристики абзацев, характеристики шрифтов и так далее. Для хранения информации о форматировании используются невидимые символы.

Такие файлы можно просматривать и редактировать только в том редакторе, в котором они созданы. Например, файлы с расширением \*.doc.

ШРИФТЫ

Разделяют растровые шрифты и масштабируемые. Растровые шрифты задаются при помощи точечных изображений – растров. Масштабируемые задаются при помощи описаний контуров символов.

Принцип WYSIWIG – соответствие символов на экране и на печати.

Post Script – шрифты фирмы Adobe Systems, который используют специальный язык описания Post Script являются масштабируемыми шрифтами.

True Type – другой формат масштабируемых шрифтов, разработанный Apple и Microsoft.

Файлы шрифтов: \*.fon – растровые, \*.ttf - масштабируемые.

Различают шрифты с засечками и шрифты без засечек. Шрифты с засечками используются для основного текста, поскольку лучше воспринимаются при чтении. А шрифты без засечек используются для выделения (заголовком и прочего).

Различают также пропорциональные шрифты и непропорциональные.

Пропорциональные шрифты – ширина отдельных символов и расстояние между соседними символами меняется так, чтобы сопряжение символов было наиболее благоприятно для чтения.

Непропорциональные шрифты – каждый символ, вместе с окаймляющими его интервалами, имеет строго определённую длину.

Имеются специальные шрифты, например шрифт Symbol. Он содержит в себе математические символы и греческие буквы.

# ТЕКСТОВЫЕ РЕДАКТОРЫ

Текстовый редактор – программа, предназначенная для обработки текстовой информации.

*Функции:*

* Запись текстов в файл.
* Вставка, удаление, замена символов, строк, фрагментов текста.
* Проверка орфографии и грамматики.
* Оформление текста различными шрифтами.
* Выравнивание текста.
* Разбиение текста на страницы.
* Поиск и замена слов и выражений.
* Включение в текст простых иллюстраций.
* Печать текста.

ТИПЫ ТЕКСТОВЫХ РЕДАКТОРОВ

* Редакторы текста. Предназначены для создания и редактирования текстовых файлов/текстовых программ. (Блокнот)
* Редакторы документов (Текстовые процессоры). Предназначены для работы с документами, структурно-состоящими из абзацев, страниц, вложенных документов и прочих элементов. Позволяют не только вводить текста, но и форматировать его, т.е. оформлять. В структуру документа может входить информация о форматировании, таблицы и графические изображения, создаваемые в других приложениях. (Microsoft Word)
* Издательские системы. Используются для подготовки больших сложных документов (книг, журналов, газет и прочего). Здесь работа состоит из двух этапов. Первый: предварительная подготовка материалов. Используются редакторы документов. Второй: вёрстка текста. Используются издательские системы. Размещение текста по страницам, вставка рисунков, использование различных шрифтов. По сравнению с редакторами документов имеют больше возможностей. (Adobe inDesign)
* Редакторы научных текстов. Обеспечивают подготовку и редактирование научных текстов, содержащих большое количество математических формул, графиков, специальных символов. (LaTeX, TeX)

# MICROSOFT EXCEL

Компьютерные программы, предназначенные для хранения и обработки данных, представленных в табличном виде, называют электронными таблицами (редакторы таблиц, табличные процессоры).

*Функции электронных таблиц:*

* Создание и редактирование электронных таблиц.
* Оформление и печать электронных таблиц.
* Создание многотабличных документов, объединённых формулами.
* Построение диаграмм, их модификация и решение экономических задач графическими методами.
* Обработка результата эксперимента.
* Проведение поиска оптимальных значений параметров и прочее.

СТРУКТУРА ДОКУМЕНТА EXCEL

Таблицы состоят из столбцов и строк. Элементы данных записываются на их пересечениях. Такое пересечение, создающее место для записи данных, называется ячейкой таблицы.

Столбцы могут быть озаглавлены латинскими буквами и двухбуквенными комбинациями. Строки могут нумероваться цифрами. Ячейки обозначаются по номеру столбца и строки, на пересечении которых они находятся (A1, DE234).

Особенностью электронных таблиц является возможность использования формул для описания связи между значениями различных ячеек. Расчёт по заданным формулам выполняется автоматически. Изменение содержимого какой-либо ячейки приводит к пересчёту значений всех ячеек, которые с ней связаны формульными отношениями и тем самым к обновлению всей таблицы в соответствии с изменившимся данными. Адреса ячеек используются для записи формул, определяющих взаимосвязь между значениями, расположенными в разных ячейках.

На данные, расположенные в соседних ячейках, можно ссылаться в формулах как на единое целое. Такую группу ячеек называют диапазон (А1:С15).

Одна из ячеек всегда является активной.

Документ Excel называется рабочей книгой. Рабочая книга представляет собой набор рабочих листов. Каждый рабочий лист имеет табличную структуру и название и может содержать несколько таблиц.

ТИПЫ ДАННЫХ

Различают основные и производные данные. Данные, которые нельзя определить по другим ячейкам таблицы называют основными. Данные, зависящие от значений других ячеек таблицы, называют производными.

*В Excel ячейка может содержать следующие типы данных:*

* Текст – любая последовательность символов. Данные текстового типа используются для заголовков таблиц, заголовков строк и столбцов, а также для комментариев.
* Число (числовая константа).
* Формула – выражение, состоящее из числовых величин и арифметических операций. Кроме числовых величин, в формулу могут входить в качестве аргументов адреса ячеек, функций и другие формулы. В ячейке, в которой находится формула, виден только результат вычислений. Саму формулу можно увидеть в строке формул, когда данная ячейка становится активной.
* Функция – запрограммированная формула. Позволяет проводить часто встречающиеся последовательности вычислений.
* Дата. Может быть представлена в различных форматах. С датами можно выполнять различные арифметические и логические операции.

АБСОЛЮТНЫЕ И ОТНОСИТЕЛЬНЫЕ АДРЕСА ЯЧЕЕК

По умолчанию адреса ячеек в формулах рассматриваются как относительные. Это означает, что при копировании формулы адреса в ссылках автоматически изменяются в соответствии с относительным расположением исходной ячейки и создаваемой копии.

При абсолютной адресации адреса ячеек при копировании не изменяются. Элементы номера ячейки, использующие абсолютную адресацию, предваряются символом $.

*Четыре способа адресации:*

* Относительный: А1
* Абсолютный: $A$1
* Абсолютный по первому номеру и относительный по второму: $A1
* Относительный по первому номеру и абсолютный по второму: A$1

Для изменения адресации при редактировании формулы необходимо выделить ссылку на ячейку и нажать клавишу F4.

# ПРЕДСТАВЛЕНИЕ ГРАФИЧЕСКОЙ ИНФОРМАЦИИ

Компьютерная графика – область информатики, изучающая методы и средства создания и обработки изображений с помощью программно-аппаратных вычислительных комплексов. Изображения при этом могут быть представлены как на экране монитора, так и на различных внешних носителях. В зависимости от способа формирования изображения разделяют растровую, векторную и фрактальную графику. Отдельной областью считается трёхмерная графика, изучающая приёмы и методы построения объёмных моделей в трёхмерном пространстве.

В зависимости от особенностей цветового представления разделяют чёрно-белую и цветную графику. По области применения разделяют инженерную графику, научную графику, веб-графику, компьютерную полиграфию и другие разделы.

# ОСОБЕННОСТИ ЧЕЛОВЕЧЕСКОГО ЗРЕНИЯ. ЯРКОСТНАЯ И ЦВЕТОВАЯ ИНФОРМАЦИЯ. ЦВЕТОВАЯ ТЕМПЕРАТУРА.

Красный цвет – 720-620 нм. Фиолетовый цвет – 430-380 нм.

Кроме длины волны световой луч характеризуется определённой энергией. Энергия определяет интенсивность (яркость) светового луча. С уменьшением длины волны увеличивается энергия светового луча.

Чем выше цветовая температура источника света, тем больше излучается коротковолновых лучей. При понижении её, в излучении преобладают длинноволновые лучи (красный цвет).

# РАСТРОВАЯ ГРАФИКА

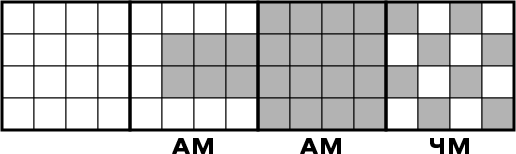
Растровое изображение формируется из отдельных точек. Важнейшим параметром, характеризующим растровое изображение, является разрешение. Оно характеризует количество точек, приходящиеся на единицу длины. При этом различают разрешение оригинала, разрешение экранного изображения и разрешение печатного изображения.

Разрешение оригинала измеряется в точках на дюйм (DPI). Разрешение экранного изображения определяется количеством точек по горизонтали и вертикали (точку в этом случае называют пиксель). Разрешение печатного изображения – размер точки растрового изображения зависит от используемого метода и параметров наложения растра оригинала. При наложении растра на оригинал накладывается сетка линий, ячейки которых образуют элемент растра.

Частота сетки растра измеряется числом линий на дюйм (LPI) и называется линиатурой. Степень заполнения ячейки растра определяет интенсивность тона/светлоту данного элемента изображения.

Человеческое зрение воспринимает не более 256 уровней интенсивности. В полутоновых изображениях используется от 150 до 256 уровней интенсивности тона, меньшее их число ухудшает качество изображения. Соответствующий размер ячейки растра будет 16х16.

Для наложения растра используются методы с частотной модуляцией и амплитудной модуляцией. Амплитудная модуляция – здесь ячейки заполняются рядом друг с другом. Чем больше ячеек заполнено – тем выше интенсивность. В случае частотной модуляции ячейки занимаются равномерно.



Между разрешением оригинала (DPI) и линиатурой растра (LPI) и градацией (N) имеется следующая связь:

dpi 2

N = ( lpi) + 1

dpi lpi =

√M - 1

Глубина цвета (цветовое разрешение) – количество цветов, используемых для представления каждой точки изображения.

При печати многоцветных изображений растры различных цветов накладываются друг на друга.

*Достоинством* растровых изображений является высокая точность передачи оригинала.

*Недостатки:*

* Большой объём памяти, необходимый для хранения изображения.
* Невозможность увеличения, для рассмотрения деталей. Увеличение изображения приводит к увеличению отдельных точек, т.е. проявляется эффект, называемый пикселизацией.

# ВЕКТОРНАЯ ГРАФИКА

В отличие от растровой графики, где базовым элементом является точка, базовым элементом векторной графики является линия. Линия описывается математически как единый объект и изображение таким образом занимает гораздо меньше памяти.

Линия – элементарный объект векторной графики.

*Свойства линии:*

* Форма.
* Толщина.
* Цвет.
* Начертание.

*Замечание:* замкнутые линии имеют свойства заполнения. Охватываемое ими пространство может быть заполнено другими объектами (текстурами, картами) или выбранным цветом.

Текстура – повторяющееся изображение. Карта – заготовленное растровое изображение.

*Замечание:* простейшая незамкнутая линия ограничена двумя точками, именуемыми узлами. Параметры узлов определяют форму линии. Все прочие объекты векторной графики состоят из линий (например квадрат – 4 линии, куб – 12 линий).

# МАТЕМАТИЧЕСКИЕ ОСНОВЫ ВЕКТОРНОЙ ГРАФИКИ

Объекты векторной графики хранятся в памяти в виде наборов параметров.

Точка представляется на плоскости двумя числами (x, y), указывающими её положение относительно начала координат.

Прямая линия описывается уравнением 𝑦 = 𝑘𝑥 + 𝑏. Для описания бесконечной прямой в известной системе координат достаточно указать параметры k и b.

Отрезок прямой – необходимо описание ещё двух параметров, например координат

(𝑥1, 𝑥2) конца отрезка.

Кривые второго порядка – линии, уравнения которых содержат степени не выше второй. Кривая второго порядка не имеет точек перегиба.

В общем виде они задаются так:

𝑥2 + 𝑎1𝑦2 + 𝑎2𝑥𝑦 + 𝑎3𝑥 + 𝑎4𝑦 + 𝑎5 = 0

То есть необходимо указать пять параметров. Для описания отрезка кривой необходимо ещё два параметра.

Кривая третьего порядка – уравнения, описывающие кривые, содержат степени не выше третьей. Кривые третьего порядка содержат точки перегиба. Эта особенность позволяет сделать кривые третьего порядка основой отображения природных объектов.

В общем случае кривая третьего порядка описывается следующим уравнением:

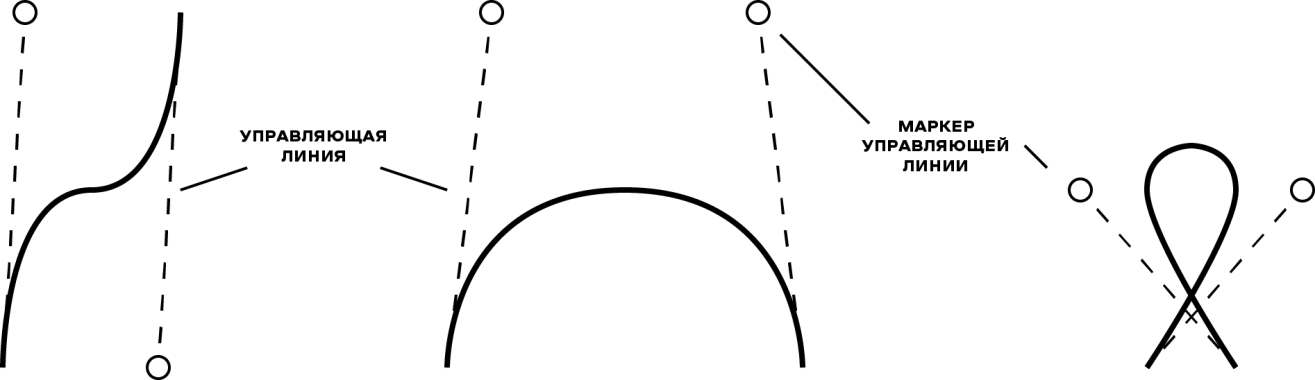
𝑥3 + 𝑎1𝑦3 + 𝑎2𝑥2𝑦 + 𝑎3𝑥𝑦2 + 𝑎4𝑥2 + 𝑎5𝑦2 + 𝑎6𝑥𝑦 + 𝑎7𝑥 + 𝑎8𝑦 + 𝑎9 = 0

Таким образом, для описания необходимо девять параметров. Для описания отрезка кривой потребуется ещё два параметра.

КРИВЫЕ БЕЗЬЕ

В векторных редакторах используют не любые кривые третьего порядка, а кривые Безье

* частный случай кривых третьего порядка. Для описания отрезков данных кривых необходимо не одиннадцать параметров, а только восемь. Отрезок кривой Безье описывается двумя касательными, проведёнными к отрезку линии в точках её концов. На форму линии влияют угол наклона касательной и длина её отрезка.



*Достоинства векторной графики:*

* + Небольшой объём данных, необходимый для описания изображения.
  + Возможность масштабирования рисунка.

*Недостатки:*

* + Сложность задания изображений.

Таким образом, растровую графику используют для создания художественных изображений. Векторную – для создания несложных рисунков, чертежей, а также в областях, где существенно свойство масштабируемости изображения (в картографии, в конструкторских системах автоматизируемого проектирования (SAPR), в автоматизированных системах архитектурного проектирования).

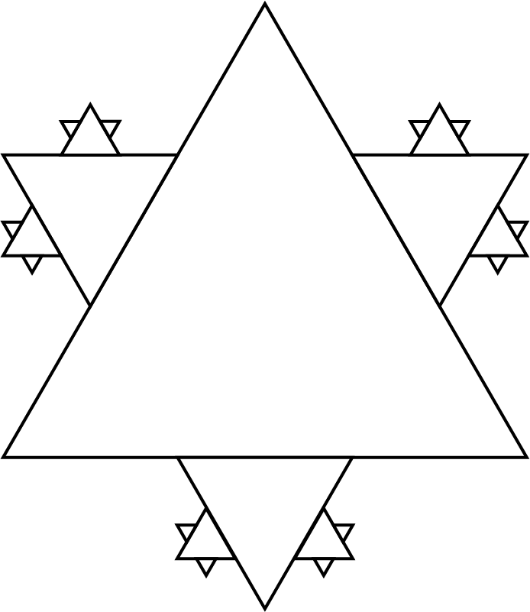
*Замечание:* изображение на экране монитора или при печати является растровым, для отображения векторной графики используются специальные вычисления каждой точки изображения. В этом смысле говорят о вычисляемой графике.

# ФРАКТАЛЬНАЯ ГРАФИКА

Фрактальная графика является вычисляемой, как и векторная, но базовым элементом фрактальной графики является математическая формула, то есть объект как таковой отсутствует и изображение строится исключительно по уравнению или системе уравнений. Изменив коэффициенты в уравнении, можно получить другую картину, то есть указывается не объект, а способ его формирования.

Фрактальный треугольник

1. Построить равносторонний треугольник со стороной a.
2. Разделить каждую его сторону на три равных отрезка.
3. На среднем отрезке стороны построить равносторонний треугольник со стороной 1/3a.
4. На других со стороной 1/9а.
5. С полученными треугольниками повторить те же операции.



Фрактальными свойствами обладают многие объекты живой и неживой природы. В частности, рост кристаллов и растений строится по фрактальной схеме. Таким образом по отдельному элементу объекта с помощью соответствующих математических методов проследить свойства всего объекта.

Фрактальная графика используется для представления различных природных объектов и для автоматической генерации необычных иллюстраций.

# ПРЕДСТАВЛЕНИЕ ЦВЕТА

Световой поток формируется из лучей, представляющими собой комбинацию трёх основных спектральных цветов: красного, зелёного и синего.

Для излучающих объектов характерна аддитивное цветовоспроизведение. Цветовые излучения суммируются (например, изображение на мониторе). Для отражающих объектов характерно субтрактивное цветовоспроизведение. Цветовые излучения вычитаются (например, полиграфическая печать).

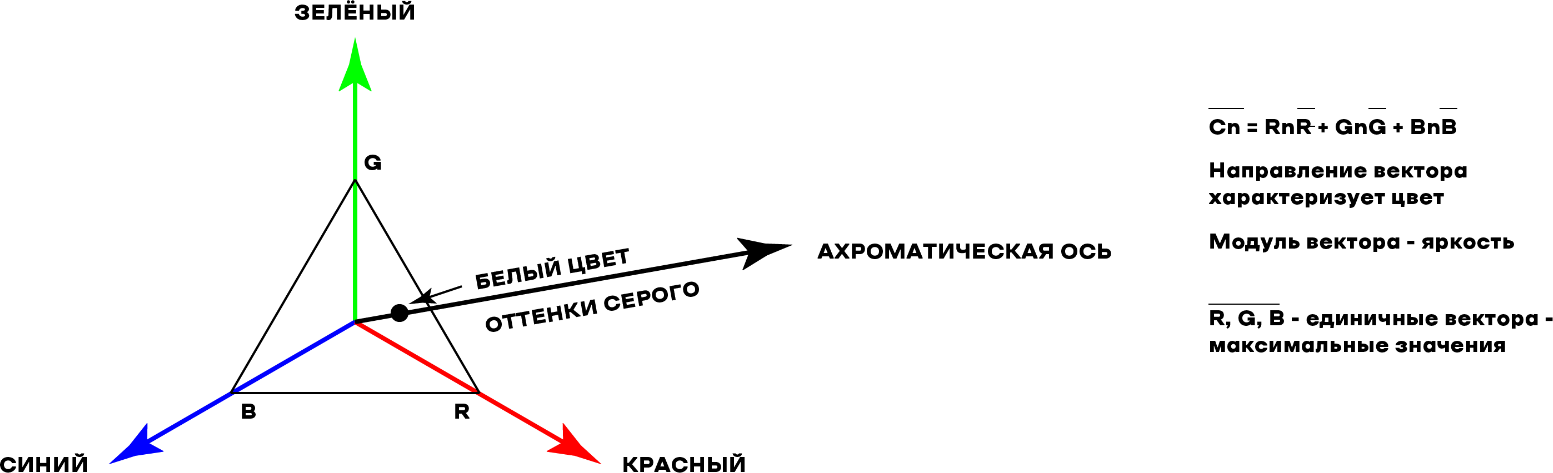
Светлота – величина, определяющая различия между зрительными ощущениями, вызываемыми двумя смежными одноцветными поверхностями.

Насыщенность – показывает, насколько данный цвет отличается от монохромного излучения того же цветового тона.

*Ахроматические цвета (белый, серый, чёрный)* характеризуются только светлотой.

*Хроматические цвета* имеют параметры насыщенности, светосилы (светлоты) и цветового тона.

Любой цвет можно представить точкой или вектором в трёхмерной системе координат, образующей цветовое пространство.



# КАЧЕСТВО ИЗОБРАЖЕНИЯ. РАЗРЕШЕНИЕ ПО ЭЛЕМЕНТАМ. КРИТЕРИЙ НАЙКВИСТА.

Помимо увеличения числа градаций серого цвета, качество изображения можно повысить за счёт увеличения числа точек в модели изображения. Размер области изображения, которое моделируется одной точкой определяет разрешение по элементам. Минимальный уровень разрешения по элементам изображения определяется тонкостью деталей изображения, которые надо воспроизвести. Минимальное разрешение определяет критерий Найквиста: чтобы результаты измерений были лишены искажений, число замеров должно по крайней мере в двое превосходить число учитываемых деталей. К приложению об оцифровке изображений это означает, что на самую малую делать должно приходиться как минимум две точки.

Если это правило будет нарушено, то появляются дефекты изображения, например, муар. Он выражается в появлении серых пятен, нечёткости контуров.

# ОТРАЖЁННЫЙ СВЕТ И ВОСПРИЯТИЕ ЦВЕТООСВЕЩЁННОСТИ ОБЪЕКТА

Идеально белой является поверхность, которая полностью отражает волны из всего видимого спектра. Идеально чёрная поверхность, наоборот, поглощает все волны видимого спектра. Серая поверхность равномерно отражает световые волны любой длины.

Когда мы воспроизводим цветные печатные документы наше восприятие основано на отражённом свете. В этом случае в качестве основных берутся цвета, представляющие комбинацию двух из трёх цветов RGB. Тогда при освещении краситель будет поглощать третий цвет. Так, например, голубой поглощает красный, пурпурный – зелёный, жёлтый – синий. Если поместить рядом две точки, например пурпурную и

жёлтую, то одна поглощает зелёный, вторая – синий, а отражена будет красная составляющая. Чёрный цвет получается в результате смешения голубого, пурпурного и жёлтого.

# ЦВЕТОВЫЕ МОДЕЛИ

Цветовое разрешение (глубина цвета) определяет метод кодирования цветовой информации.

|  |  |  |
| --- | --- | --- |
| КОЛ-ВО БИТОВ | КОЛ-ВО ЦВЕТОВ | НАЗВАНИЕ |
| 2 бита | 4 цвета | Индексная |
| 8 битов | 256 цветов | Фиксированная |
| 16 бит | 65536 цветов | High Color |
| 24 бит | 16,5 млн. цветов | True Color |

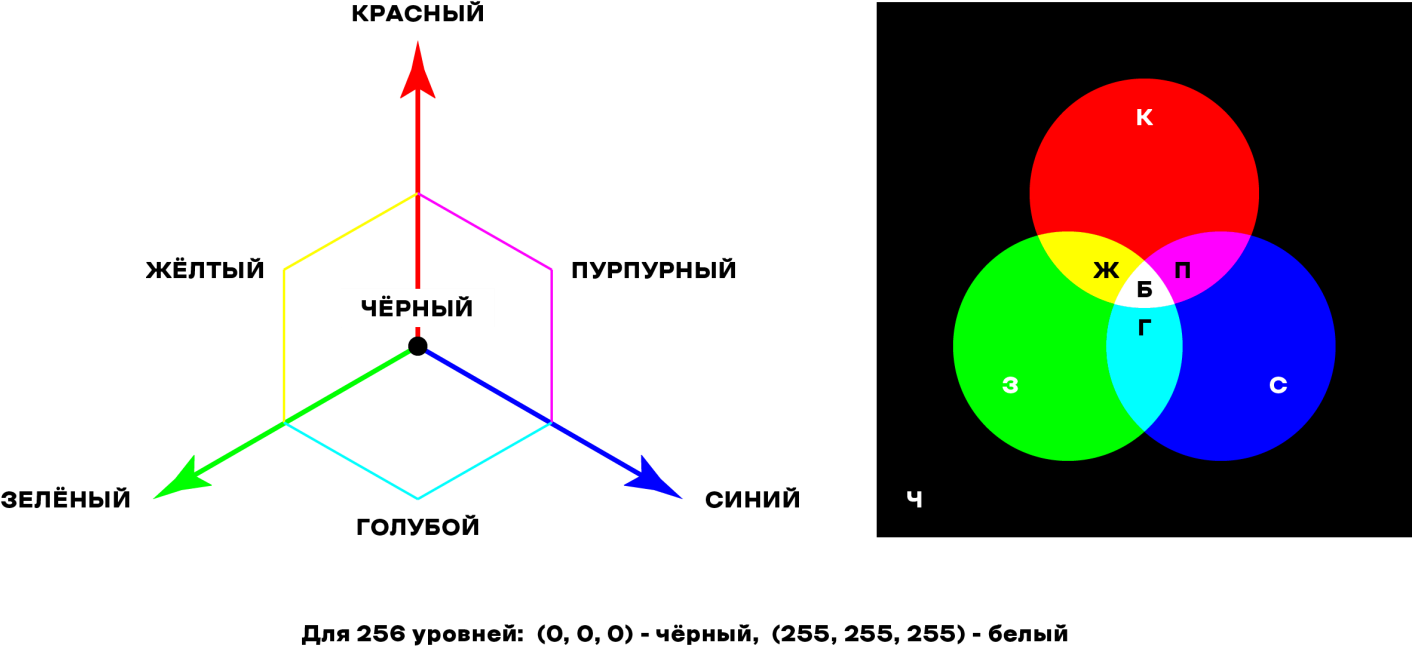
Большинство цветовых оттенков образуются смешиванием основных цветов.

Способ разделения цветового оттенка на составляющие компоненты называется цветовой моделью.

# МОДЕЛЬ RGB

При наложении цветов яркость суммарного цвета увеличивается. Графически эту модель можно представить в виде трёхмерного пространства.

Модель RGB служит основой при создании и обработке компьютерной графики, предназначенной для электронного воспроизведения.

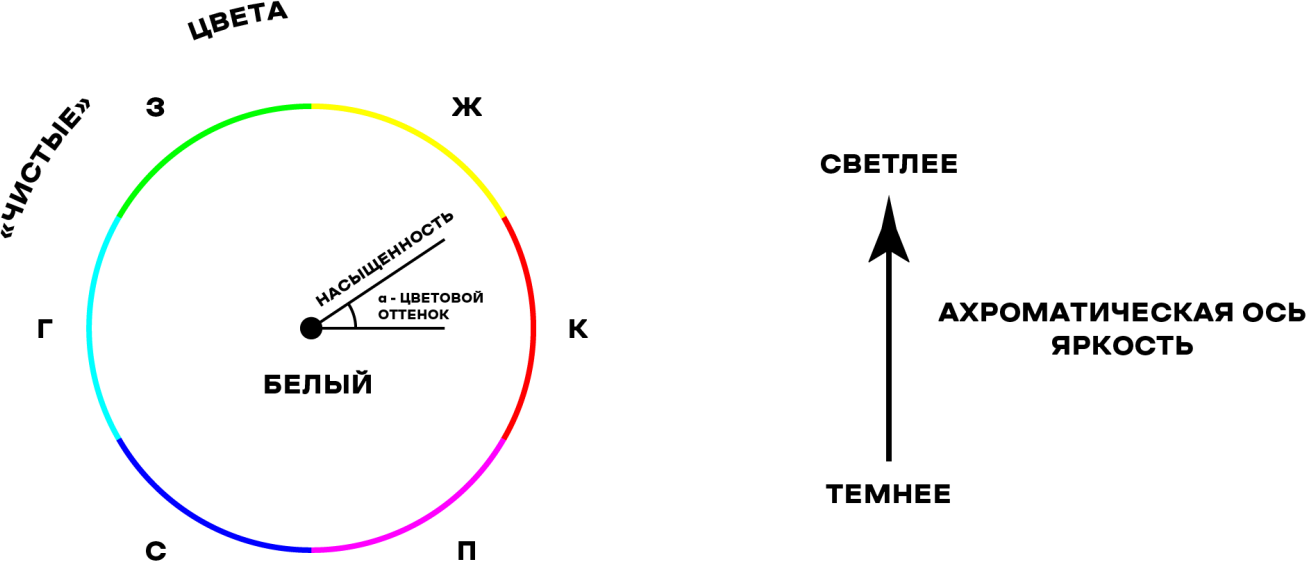


# МОДЕЛЬ HSB

HSB (Hue Saturation Brightness) максимально учитывает особенности человеческого зрения. Цвет здесь представляется в виде следующих *характеристик*: Hue – тон, Saturation – насыщенность, Brightness – яркость.

Данная модель используется для создания различных художественных изображений с имитацией работы и инструментария художника.

После создания изображения его можно преобразовать в другую цветовую модель в зависимости от предполагаемого способа публикации.



# МОДЕЛЬ CMYK, ЦВЕТОДЕЛЕНИЕ

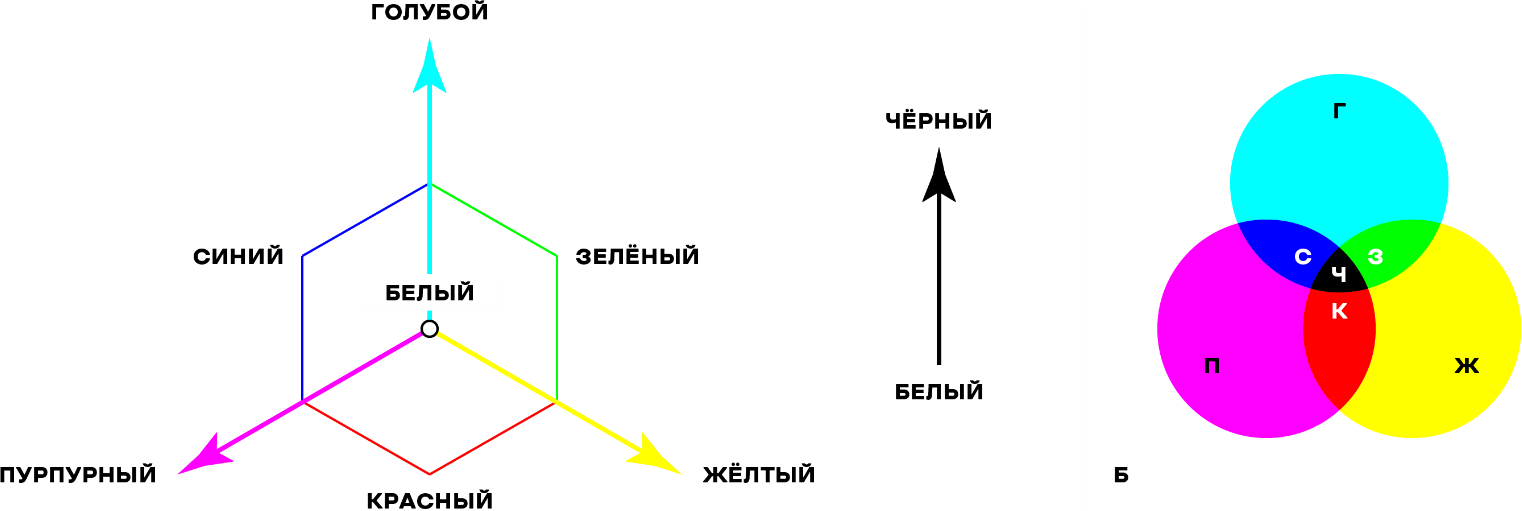
CMYK (Cyan Magenta Yellow blacK) – субтрактивная модель, используется при печати.

CMYK – это дополнительные цвета, получаемые вычитанием основных из белого. Голубой (белый – красный = зелёный + синий), Пурпурный (белый – зелёный = красный

+ синий), Жёлтый (белый – синий = красный + зелёный). Голубой, пурпурный, жёлтый

* дополнительные цвета, они дополняют основные до белого.

Наложением дополнительных цветов нельзя получить на практике чистый чёрный цвет, поэтому в данной модели используется ещё четвёртый цвет – чёрный.



*Замечание:* для печати на полиграфическом оборудовании, цветовое изображение нужно разделить на составляющие, соответствующие цветовой модели цвета. Этот процесс называется цветоделением. В итоге получают четыре отдельных изображения, содержащих одноцветное представление каждого компонента в оригинале. Затем, в типографии, печатают многоцветное изображение, получаемое наложением цветов CMYK.

# ЦВЕТОВАЯ ПАЛИТРА

Цветовая палитра – таблица данных, в которой хранится информация о том, каким цветом закодирован тот или иной цвет. Эта таблица создаётся и хранится вместе с графическим файлом.

Цветовые палитры создаются на основе цветовых моделей. Но если цветовой моделью теоретически можно воспроизвести любой из описываемых её цветов, то цветовая палитра содержит ограниченный набор цветов.

# 24-Х РАЗРЯДНЫЙ СПОСОБ КОДИРОВАНИЯ, TRUE COLOR

Используется 16,5 миллионов цветов. Каждый оттенок кодируется 256 цветами. Цветовая палитра в этом случае не нужна.

# ИНДЕКСНАЯ ПАЛИТРА

Используется 256 цветов (8 бит). В этом случае каждый цветовой оттенок представлен одним числом – индексом цвета. Цвет разыскивается по индексу сопроводительной цветовой палитры, приложенной к файлу. Могут использоваться различные палитры с различными наборами цветов.

# ФИКСИРОВАННАЯ ПАЛИТРА

Здесь для хранения информации о размере изображения используется 16 бит (65536 цветов). Каждый цвет выражается двухбайтным кодом. Однако, прикладывать такую таблицу к файлу нецелесообразно из-за её большой величины, поэтому используется единая фиксированная палитра. Она не прикладывается к файлу, поскольку в любой графическом файле, имеющем 16-ти разрядное кодирование цвета, один и тот же код всегда выражает один и тот же цвет.

# ФОРМАТЫ ГРАФИЧЕСКИХ ФАЙЛОВ

TIFF (.tiff) – предназначен для хранения растровых изображений высокого качества. Цветовой охват – от чёрно-белого до 32-х разрядной модели CMYK. Для уменьшения размера файла используется встроенный алгоритм сжатия.

Photoshop Document (.psd) – используется в программе Adobe Photoshop. Предназначен для хранения растровой информации, используется 48-ми разрядное кодирование цвета, цветоделение, различные цветовые модели. Недостаток: большой объём файла.

JPEG (.jpg) – предназначен для хранения растровых изображений. Позволяет регулировать соотношение между степенью сжатия файла и качеством изображения.

GIF (.gif) – хранение растровых изображений с фиксированным (256) количеством цветов. Обеспечивает высокую степень сжатия, чем обусловлена его распространённость в интернет-публикациях.

EPS (.eps) – универсальный формат описания векторных и растровых изображений на языке PostScript. В файле могут одновременно храниться растровые, векторные изображения и текст.

Windows Bitmap (.bmp) – формат хранения растровых изображений в операционной системе Windows.

Windows Metafile (.wmf) – формат хранения векторных изображений в операционной системе Windows.

PDF (.pdf) – формат описания документов фирмы Adobe. Предназначен для хранения документа целиком: текста и графики. Обеспечивает аппаратную независимость изображения, то есть вывод изображения на любые устройства. Обеспечивает компактность файлов при высоком качестве иллюстраций.

# СРЕДСТВА РАБОТЫ С КОМПЬЮТЕРНОЙ ГРАФИКОЙ

Средства работы с растровой графикой: Paint, Adobe Photoshop. Средства работы с векторной графикой: Adobe Illustrator, Corel Draw.

# ПРОГРАММЫ ТРАССИРОВЩИКИ

Специальные программы преобразования растровых изображений в векторные: Adobe Streamline, Corel Trace.

# ПРЕДСТАВЛЕНИЕ ВИДЕОИНФОРМАЦИИ

Под видеоинформацией будем понимать группу сменяющих друг друга изображений, в данном случае, называемые кадрами. Чтобы уменьшить этот объём информации используют *следующие приёмы*:

* Для отдельного кадра фильма требования качества могут быть ниже, чем для отдельного фотоизображения, поэтому возможно сохранение кадров с потерей информации.
* Существенно сократить объём хранимой информации можно за счёт того, что сохраняется не весь новый кадр, а только то, чем он отличается от предыдущего.

# ПРЕДСТАВЛЕНИЕ ЗВУКОВОЙ ИНФОРМАЦИИ

При записи звуковой информации роль нашего слуха моделируется с помощью микрофона. В результате, под воздействием звуковой волны вырабатывается электрический сигнал, непрерывно изменяющийся во времени. Чтобы представить этот аналоговый сигнал в цифровой форме, проводят его измерения с определённой частотой. Чем выше частота измерения, тем лучше качество дискретизации. Обратная сторона: увеличивается поток информации, увеличивается объём информации, а следовательно, требуется больше места. Поэтому частота дискретизации должна быть

минимально допустимой. Минимальная частота определяется из теории Найквиста. Наш слух воспринимает колебания до 20кГц, соответственно частота дискретизации должна быть не меньше 40 кГц. Стандартно используют частоту 44.1кГц.

Выбирая такую минимально допустимую частоту дискретизации, мы будем получать звуковые файлы большого размера, поэтому для хранения звуковой информации применяются различные методы сжатия. Кроме того, специальным образом кодируются паузы: в начале паузы записывается некоторый её признак и продолжительность.

# АЛГОРИТМ

Алгоритм – это точное предписание, которое определяет процесс, ведущий от исходных данных к требуемому конечному результату.

Если процесс, определяемый алгоритмом, заканчивается получением результатов, то говорят, что соответствующий алгоритм применим к рассматриваемой совокупности исходных данных. В противном случае говорят, что алгоритм не применим к совокупности исходных данных. Любой применимый алгоритм обладает следующими основными свойствами:

* Результативность – означает возможность получения результата после выполнения конечного количества операций.
* Определённость – состоит в совпадении получаемых результатов, независимо от пользователя и применяемых технических средств.
* Массовость – заключается в возможности применения алгоритма к целому классу однотипных задач, различающихся конкретными значениями исходных данных.

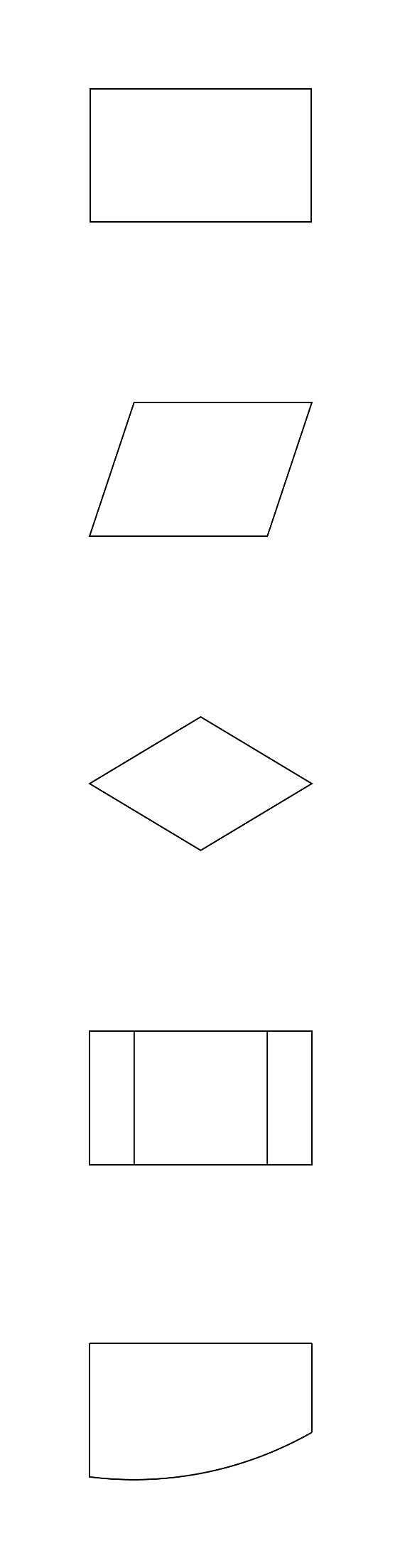
Для описания алгоритма необходимо описать следующие его *элементы*: наборы объектов, составляющих совокупность возможных исходных данных промежуточных и конечных результатов; правило начала; правило переработки информации, то есть описания последовательности действий; правило окончания; правило извлечения результатов.

Программа для ЭВМ – представляет собой описание алгоритма и данных на некотором языке программирования, предназначенная для последующего автоматического выполнения.

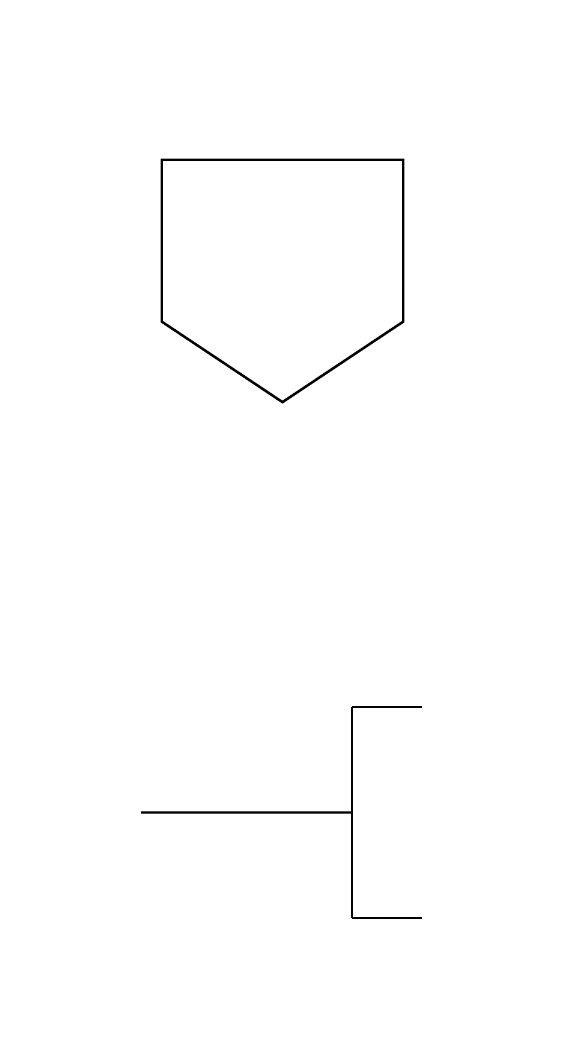
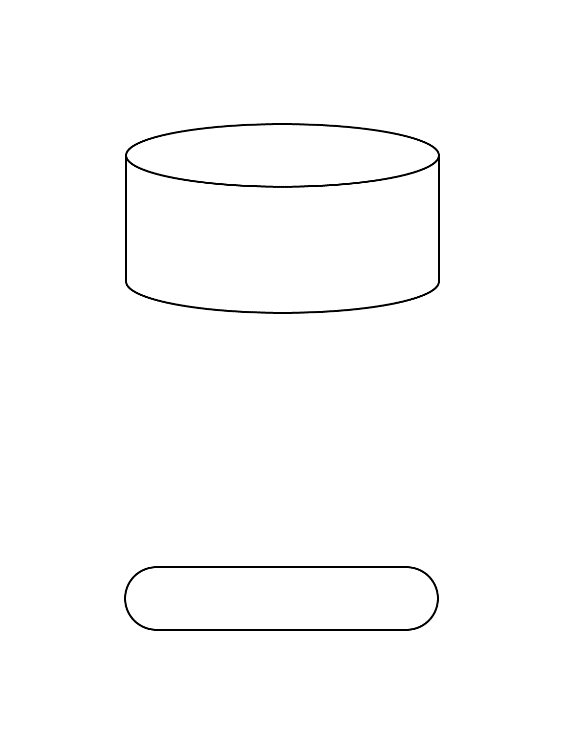
# СПОСОБЫ ОПИСАНИЯ АЛГОРИТМА

При блок схемном описании алгоритм обозначается геометрическими фигурами (блоками), связанными по управлению линиями со стрелками. Оформление алгоритмов программ описаны в ESPD.

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Обозначение** | **Функция** |
| **Процесс** |  | Выполнение операции или группы операций, в результате которых изменяется значение, форма представления или расположение данных. |
| **Ввод-вывод** |  | Преобразование данных в форму, пригодную для обработки (ввод) или отображения результатов обработки (вывод). |
| **Решение** |  | Выбор направления выполнения алгоритма в зависимости от некоторых переменных условий. |
| **Предопределённый процесс** |  | Использование ранее созданных и отдельно написанных программ (подпрограмм). |
| **Документ** |  | Вывод данных на бумажный носитель. |



|  |  |  |
| --- | --- | --- |
| **Магнитный диск** |  | Ввод-вывод данных, носителем которых служит магнитный диск. |
| **Пуск-остановка** |  | Начало, конец, прерывание процесса обработки данных. |
| **Соединитель** |  | Указание связи между прерванными линиями, соединяющими блоки. |
| **Межстраничный соединитель** |  | Указание связи между прерванными линиями, соединяющими блоки, расположенные на разных листах. |
| **Комментарий** |  | Связь между элементом схемы и пояснением. |



|  |  |  |
| --- | --- | --- |
| **Предопределённое действие**  **Цикл с параметром**  **Модификация** |  | Например: организация цикла с параметром. |
| **Цикл** |  | Начало и конец цикла. |

# ТИПЫ АЛГОРИТМОВ

Алгоритм линейной структуры – алгоритм, в котором блоки выполняются последовательно, друг за другом, в порядке, заданной схемой.

Алгоритм разветвляющейся структуры – алгоритм, в котором в зависимости от выполнения некоторого логического условия вычислительный процесс осуществляется по одной или другой ветви.

Алгоритм циклической структуры – многократно повторяемый участки вычислительного процесса, называются циклами. Использование циклов позволяет существенно сократить объём схемы алгоритма и длину соответствующей программы. Различают циклы с заданным и неизвестным числом повторений. Для организации *цикла необходимо*:

* Задать перед циклом начальное значение переменной, изменяющейся в цикле, так называемый, параметр системы.
* Изменять переменную перед каждым новым повторением цикла.
* Проверять условие окончания или повторения цикла.
* Управлять циклом, то есть переходить к началу, если он не закончен, или выходить из него по окончании.

Пункты 2-4 выполняются многократно.

# СВОЙСТВА АЛГОРИТМА

* Конечность – алгоритм состоит из отдельных элементарных шагов или действий, причём множество различных шагов, из которых составлен алгоритм, конечно. Алгоритм должен заканчиваться за конечное число шагов. Если строится бесконечный, сходящийся к искомому решению, процесс, то он обрывается на некотором шаге и полученное значение принимается за приближённое решение рассматриваемой задачи. Точность приближения зависит от числа шагов.
* Элементарность – каждый шаг алгоритма должен быть простым, чтобы устройство, выполняющее операции, могло выполнить его одним действием.
* Дискретность – процесс решения задачи представляется конечной последовательностью шагов, и каждый шаг алгоритма выполняется за конечное время.
* Эффективность – одну и ту же задачу можно решить по-разному и, соответственно, за разное время и с различными затратами памяти. Желательно, чтобы алгоритм состоял из минимального числа шагов и при этом решение удовлетворяло бы условию точности и требовало минимальных затрат ресурсов.

Выделяет следующие основные *способы записи алгоритмов*:

* Вербальный – то есть озвученный на обычном человеческом языке.
* Символьный – алгоритм описывается с помощью набора символов.
* Графический – алгоритм описывается с помощью набора графических изображений.

Если исполнителем будет *человек*, то запись алгоритма может быть не полностью формализована, на первое место выдвигаются понятность и наглядность. В этом случае используют словесную форму записи или блок-схему.

Для записи алгоритмов, предназначенных для исполнителей *автоматов*, необходима формализация. В таких случаях применяют формальные специальные языки. Преимущества формального способа записи состоит в том, что он даёт возможность изучать алгоритмы как математические объекты.

# МЕТОДЫ РАЗРАБОТКИ АЛГОРИТМОВ

Ключевым подходом в алгоритмизации является сведение задачи к подзадаче. Это преобразование может состоять из нескольких этапов, на которых единственный шаг разбивается на несколько более простых, но ещё не элементарных. Эти более простые шаги соответствуют подзадачам, совокупное решение которых приводит к решению исходной задачи.

Любая задача может быть сформулирована как функция преобразования исходных данных в выходные данные (f(x) = y). Например, х – число, вектор, текст, база данных, рисунок и прочее. То же самое и по выходным данным: функция может быть суммой чисел, тригонометрической функцией, корнем уравнения, переводом текста и прочее.

*Разбивая задачу на подзадачи*, можно:

* Разбивать исходные и выходные данные на части или упрощать их.
* Производить декомпозицию функции, то есть превращать её в суперпозицию более простых.

Под разбиением данных понимается разделение структуры данных на части, например, разделение текста на предложения. Под упрощением данных понимаются такие ситуации, когда х – это число и его нельзя разбить на части, но его можно разложить (x

= x1 + x2), так что результаты f(x1) и f(x2) отыскиваются проще, чем f(x). При разработке алгоритма может использоваться отдельно первый подход, отдельно второй или оба подхода совместно.

Разложение задачи на последовательность разнородных подзадач состоит в следующем: обычно выделяется относительно небольшое число подзадач. Результаты решения первой подзадачи становятся исходными данными для второй подзадачи и так далее.

Частным случаем является разложение задачи на последовательность однородных подзадач, то есть некая задача P сводится к N экземплярам более простой задачи R или к простой задаче U, объединяющей N решений.

Однородность подзадач позволяет значительно сократить длину текста алгоритма за счёт применения операторов повторения.

Рекурсия – это сведение задачи к самой себе. Как и в предыдущем случае, задача сводится к более простой, но эта более простая задача имеет ту же формулировку, что и исходная, с той разницей, что решаться она должна для более простых исходных данных.

Метод последовательных приближений – в начале, каким-либо образом угадывается значение x0, близкое к решению. Задача P нахождения решения, сводится к многократному решению задачи R улучшения решения. Предполагается, что решение находится не точно, а с некоторым приближением. Требует отыскать не точное решение y, а любое решение, отличающееся от y не более чем на некоторую величину e, то есть приближённое решение (например задача о вычислении корня уравнения).

Основной проблемой является построение задачи R по исходной задаче P, доказательство факта сходимости процесса к искомому решению и обеспечение достаточно высокой скорости сходимости.

Решение обратной задачи – иногда обратная задача, то есть задача, соответствующая некой функции f(y) = x, которая решается значительно более просто, чем исходная задача. Тогда имеющийся алгоритм решения обратной задачи R можно, иногда, использовать для построения алгоритма решения прямой задачи P.

Метод полного перебора – берётся некоторое подмножество исходных данных и непосредственной проверкой (решение задачи) проверяется удовлетворяет ли это подмножество поставленному условию. Поскольку количество подмножеств имеет конечное число, то потенциально можно перебрать все подмножества и найти решение.

Метод полного перебора применим в тех случаях, когда искомое решение y = f(x) принадлежит некоторой конечной области и может быть найдена простая функция для проверки качества выбранного решения. Таким образом задача P вычисления функции f заменяется многократным решением задачи R вычисления новой функции.

Эвристические методы разработки алгоритмов – под эвристическими понимаются методы, правильность которых не доказана. Они выглядят правдоподобными и в большинстве случае должны давать верное решение. Не удаётся построить контрпример, демонстрирующий ошибочность метода, но не удаётся и доказать математическими средствами правильность метода. Тем не менее практика использования эвристических методов даёт положительный результат.

Динамическое программирование – в наиболее общей форме так называют процесс пошагового решения задач, когда на каждом шаге выбирается одно значение из множества допустимых на этом шаге, причём такое, которое оптимизирует заданную цель. Часто не удаётся разбить задачу на небольшое число подзадач, объединённое решение которых позволяет получить исходное решение задачи. Можно попытаться разделить задачу на столько задач, сколько необходимо, затем каждую из них на ещё более мелкие и так далее. Иногда проще создать таблицу решения всех подзадач, независимо от того, нужна она или нет. Заполнение таблиц – решение подзадач для получения решения определённой задачи в теории алгоритмов получила название динамическое программирование. Формы алгоритмов динамического программирования могут быть разным, общими могут быть лишь заполненные таблицы и порядок заполнения их элементов.

# КЛАССИФИКАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Программное обеспечение (ПО) – совокупность программных и документальных средств для создания и эксплуатации систем обработки данных средствами вычислительной техники.

Программное обеспечение делят на системное, прикладное и системно-прикладное.

Системное ПО – организует процесс обработки информации в компьютере и обеспечивает рабочую среду для прикладных программ. Тесно связано с аппаратными средствами.

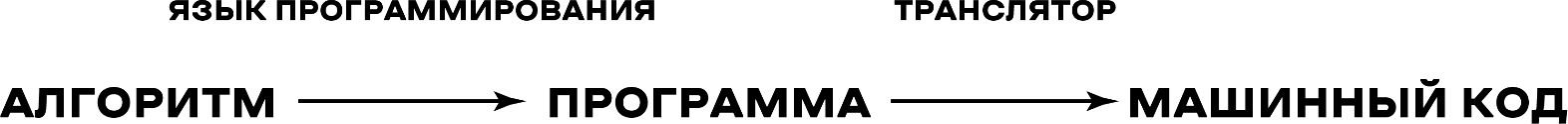
В состав системного ПО входят операционные системы, сервисные программы, программы технического обслуживания.

Прикладное ПО – предназначено для решения конкретных задач пользователя. Прикладное ПО работает под управлением системного ПО, в частности операционных систем.

# ЯЗЫКИ ПРОГРАММИРОВАНИЯ

Машинный код – команды, представляющие собой последовательности нулей и единиц.

Транслятор – специальное служебное приложение, переводящее текст программы с алгоритмического языка в машинный код.



Языки программирования (алгоритмические языки) – специальные искусственные языки, предназначенные для написания компьютерных программ.

Их отличает ограниченное количество слов, понятных транслятору, и строгие правила записи команд. Совокупность этих требований образует синтаксис языка. Смысл каждой программы и других конструкций языка образуют его семантику.

*Замечание.* Таким образом, возможны два рода ошибок при написании программы: синтаксические, выявляемые на этапе трансляции программы, и семантические (логические), возникающие при правильно написанных, но не отвечающих алгоритму командах языка. Логические ошибки выявляются на этапе тестирования программы.

Процесс поиска ошибок в программе называется тестированием. Процесс устранения ошибок в программе называется отладкой.

# КОМПИЛЯТОРЫ И ИНТЕРПРЕТАТОРЫ

Различают два вида трансляторов: компиляторы и интерпретаторы.

Компилятор – программа, переводящая текст, написанный на алгоритмическом языке как единое целое, в машинный код. В результате получается компактная и эффективная программа, которую можно использовать отдельно от системы программирования и исходного текста.

*Достоинства:* компактность и эффективность программы. Возможность использования программы отдельно от системы программирования и исходного текста.

*Недостатки:* трудоёмкость трансляции языков программирования, ориентированных на обработку данных сложной структуры, часто заранее неизвестной или динамически меняющийся в процессе работы программы.

Интерпретатор – программа, сразу выполняющая очередную команду языка, указанную в исходном тексте. При этом оператор может транслировать некоторые промежуточные представления или машинный код.

*Достоинства:* возможность контроля и регулирования процесса выполнения программы, что удобно, например, при её отладке.

*Недостатки:* низкая скорость выполнения программы. Необходимость наличия интерпретатора при переносе программы на другую машину.

# УРОВНИ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

Язык программирования, ориентированный на конкретный тип процессора, называется языком низкого уровня. Уровень языка программирования означает его близость к машинному коду (командам процессора, представленным в виде нулей и единиц).

Язык Ассемблера – язык самого низкого уровня, представляет каждую команду машинного кода, но не в виде чисел, а с помощью символьных условных обозначений

* мнемоник.

Ассемблер – вспомогательная программа в составе операционной системы для автоматического перевода исходной программы на машинный язык. Один из видов транслятора.

Однозначное преобразования одной машинной инструкции в одну команду ассемблера называется транслитерацией.

*Замечание.* Поскольку наборы инструкций различаются для разных моделей процессора, то конкретной компьютерной архитектуре соответствует свой язык ассемблера и написанная на нём программа может быть использована только в этой среде.

*Достоинства:* эффективность и компактность программ.

*Недостатки:* необходимо хорошо понимать устройство компьютера. Трудность отладки больших программ. Невозможность переноса программы на компьютер с другим типом процессора.

*Область применения:* небольшие системные приложения, драйверы устройств, модули стыковки с нестандартным оборудованием, стандартные библиотеки, реализующие алгоритмы обработки изображений и прочего.

Язык программирования высокого уровня – не учитывает особенности конкретной архитектуры процессора, однако он ближе и понятней человеку.

*Достоинства:* программы на уровне исходных текстов переносимы на другие платформы, для которых создан транслятор этого языка. Удобство разработки больших приложений.

# ЯЗЫКИ ПРОГРАММИРОВАНИЯ ВЫСОКОГО УРОВНЯ

Fortran – создан в 50-е годы, ориентирован на высокоэффективное выполнение численных расчётов. Имеется большое количество различных библиотек, используемых до сих пор.

Pascal – создан в 70-е годы, имеет жёсткую структуру программы, изначально создавался как учебный язык, однако успешно применяется и для создания крупных проектов.

Basic – создан в 60-е годы в качестве учебного, прост в изучении и весьма популярен.

C – создан в 70-е годы при разработке операционной системы Unix, изначально был ориентирован на создание системных программ, свойственна некоторая низкоуровневость.

C++ - создан в 80-м году, объектно-ориентированное расширение языка C. Требует от разработчиков высокого уровня профессиональной подготовки.

Java – создан компанией Sun в начале 90-х годов на основе языка C++. Призван упростить разработку приложений за счёт исключения низкоуровневых возможностей. Главная особенность – компиляция не в машинный код, а в платформенно-независимый байт-код. Байт-код может выполняться с помощью интерпретатора – виртуальной Java- машины, версии которой имеются для любой платформы. Таким образом программы на Java можно переносить на уровни двоичного байт-кода.

Для работы с базами данных в своё время был создан структурированный язык запросов (SQL), позволяющий выполнять их эффективную обработку. Системы управления базами данных (СУБД) поддерживают язык SQL, а также свой уникальный язык, ориентированный на особенности этой СУБД и непереносимый на другие системы.

В Интернете используются скрипт-языки – интерпретируемые языки, интерпретаторы для которых распространяются бесплатно, а сами программы в исходных текстах.

Язык HTML – создан в начале 90-х годов, язык разметки гипертекста. Предназначен для создания веб-страниц, содержащих текст, графику, звук, анимацию, гипертекстовые ссылки.

Perl – разработан в 80-х годах, предназначен для эффективной обработки больших текстовых файлов.

# СИСТЕМЫ ПРОГРАММИРОВАНИЯ

Работа с программой на алгоритмическом языке состоит из следующих шагов:

* Создание исходного текста, для чего требуется текстовый редактор, желательно специализированный.
* Перевод исходного текста программы в машинный код с помощью компилятора. Проверяются синтаксические ошибки. На этом этапе, как правило, создаётся промежуточный объектный код – двоичный файл (.obg).
* Большая программа может содержаться в нескольких файлах, для каждого из которых создаётся объектный код.
* Объединение этих объектных кодов и добавление к ним машинных кодов подпрограмм, реализующих различные стандартные функции. Такие программы содержатся в библиотеках – специальных файлах (.lib), поставляемых вместе с компилятором.
* Объединение объектных модулей и машинного кода стандартных функций выполняет редактор связей. В результате формируется работоспособное приложение – исполнимый код (.exe, .com), ориентированный на конкретную платформу, то есть на конкретную платформу – на конкретную операционную систему и конкретный процессор.

# ПАСКАЛЬ. ТЕКСТ ПРОГРАММЫ. АЛФАВИТ ЯЗЫКА.

Текст программы представляет собой последовательность строк, состоящих из символов, образующих алфавит языка. Строки программы завершаются специальными символами (управляющими символами).

*Алфавит*:

* Заглавные и строчные латинские буквы, символ «\_». (Символ «\_» считается буквой).
* Цифры от нуля до девяти.
* 22 специальных символа: + - \* / = > < . , ; : @ ‘ ( ) [ ] { } # $ ^. Используются для конструирования знака операций, выражений, комментариев, а также как синтаксические разделители.

# ЛЕКСИЧЕСКАЯ СТРУКТУРА ЯЗЫКА

Символы алфавита языка используются для построения базовых элементов языка – лексем.

Лексема – минимальная единица языка, имеющая самостоятельный смысл. В Паскале имеются *следующие классы* лексем:

* Служебные (зарезервированные слова) – ограниченная группа слов (55, например: array, const, begin и другие), построенная из букв. Смысл служебного слова фиксирован в языке. Служебные слова нельзя использовать в качестве имён, вводимых программистом.
* Идентификаторы (имена) – водятся для обозначения в программе переменных, констант, типов, меток, процедур и функций. Формируются из букв, цифр и знаков подчёркивания. Первой в идентификаторе должна стоять буква. Идентификаторы вводятся в программу посредством описаний. Замечание: заглавные и строчные буквы в языке не различаются.
* Изображения – обозначают числа, символьные строки и некоторые другие значения.
* Знаки операций – формируются из одного или нескольких специальных символов и предназначены для задания действий по преобразованию данных и вычислению значений.
* Разделители – формируются из специальных символов (например: «;» «:»)
* Комментарии – фрагменты пояснительного характера, представляют собой последовательность символов, заключённые в фигурные скобки или может использоваться комбинация (\* \*).
* Директивы компилятора – инструкции к компилятору, предписывающие тот или иной режим работы программы (например: «$B - »)
* Пробел – используется для разделения лексем в тех случаях, когда их слитное написание может изменить смысл программы.

*Структура программы*:

* Заголовок программы.
* Раздел описаний – описание данных, которыми оперирует программа.
* Раздел операторов программы – описание последовательности действий.
* Точка – программа на Паскале заканчивается точкой.
* Комментарии.

Описание данных должны предшествовать описанию действий и должны содержать упоминания всех объектов, используемых в действиях. Действия программы называют операторами.

В разделе описаний могут быть описаны *объекты пяти классов*:

* Метки – посредством меток можно указать те операторы, на которые будет передано управление из других точек программы.
* Константы – задают программе идентификаторы, являющиеся синонимами некоторых значений.
* Типы – предназначены для задания конкретных множеств значений. Указанные множества обозначаются именами и в дальнейшем служат для описания переменных.
* Переменные – совокупность данных, с которыми производятся действия. Переменная обозначается идентификатором. С каждой переменной связывается тип, определяющий множество допустимых значений этой переменной и набор допустимых операций.
* Процедуры и функции – соответствующее описание определяет часть программы как отдельную синтаксическую единицу и сопоставляет ей имя. В последствии действия, сосредоточенные в процедуре или функции, могут быть выполнено в процессе указания её имени.

Раздел операторов – начинается со служебного слова begin, содержит выражения, оканчивающиеся знаком «;», заканчивается служебным словом end.

# ТИПЫ ДАННЫХ

Тип данных – множество значений, которые может принимать переменная и как следствие, множество операций, допустимых над данной переменной. На основе стандартных типов могут быть сконструированы типы произвольной структуры и сложности.

*Система типов* в Паскале:

* Простые – базовые типы. Здесь различают скалярные (делятся на стандартные и перечислимые) и ограниченные (формируются из простых путём сужения их области допустимых значений).
* Составные – по определённым правилам строятся из простых. Здесь различают регулярные, комбинированные, файловые, множественные, строковые и объекты.
* Ссылочные – образуются из любых других типов.
* Процедурные – позволяют обращаться к подпрограммам, как к переменным.

# СТАНДАРТНЫЕ СКАЛЯРНЫЕ ТИПЫ

Целые типы – обозначают множество целых чисел в различных диапазонах.

|  |  |  |
| --- | --- | --- |
| **Название** | **Диапазон** | **Размерность** |
| Shortint | -128 … 127 | 1 байт |
| Integer | -32768 … 32767 | 2 байта |
| Longint | -2147483648 … 2147483647 | 4 байта |
| Byte | 0 … 255 | 1 байт |
| Word | 0 … 65535 | 2 байта |

Операции: +, -, \*, / (даёт вещественный результат, остаток – целый), div (деление нацело, с отображением дробной части), mod (вещественный остаток от целочисленного деления).

Вещественный тип – определяет множество вещественных значений в различных диапазонах.

|  |  |  |  |
| --- | --- | --- | --- |
| **Название** | **Диапазон** | **Размер** | **Размер мантиссы** |
| Real | 2.9E-39 … 1.7E38 | 11-12 байтов | 6 байтов |
| Single | 1.5E-45 … 3.4E38 | 7-8 байтов | 4 байта |
| Double | 5E-324 … 1.7E308 | 15-16 байтов | 8 байтов |
| Extended | 3.4E-4932 … 1.1E4932 | 19-20 байтов | 10 байтов |
| Comp | -2Е64+1 … 2Е63-1 | 10-20 байтов | 2 байта |

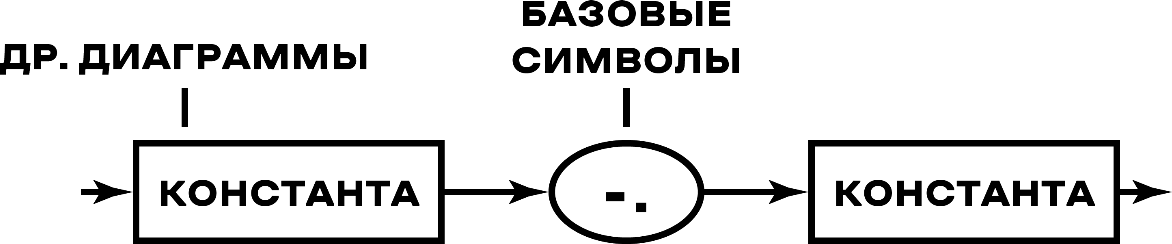
Операции: +, -, \*, /. Результат – вещественное число, если хотя бы один операнд – вещественное.

Вещественные значения могут изображаться в форме с фиксированной точкой и в форме с плавающей точкой.

Символьный тип – значениями являются символы из множества ASCII. Каждый символ таблицы ASCII имеет свой номер, соответственно можно обратиться к символу непосредственно или по его номеру. Над этим типом определены операции сравнения. Эти операции дают тот же результат, что и над соответствующими целочисленными кодами символов.

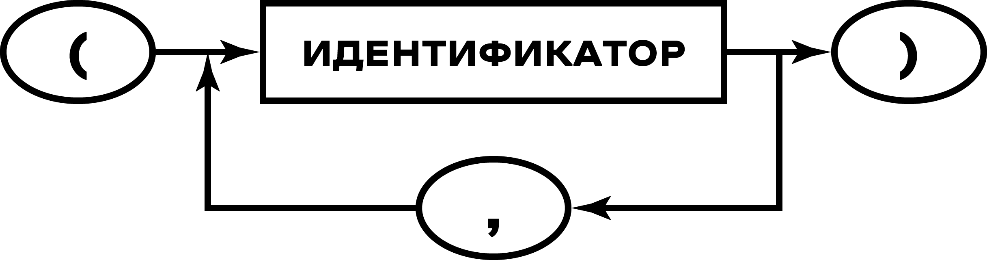
Булевский тип – этот тип составляют две логические константы True (истина) и False (ложь). Операции: операции сравнения, при этом считается, что ложь меньше, чем истина, и логические операции: логическое умножение (and), логическое сложение (or), исключающее логическое сложение (xor) и отрицание (not).

Ограниченный тип – на основе имеющихся стандартных скалярных типов программист может определить новые типы. Одним из способов организации является ограничение допустимого диапазона значений. Синтаксическая диаграмма следующая.



Ограниченные типы следует использовать в программе в случаях, когда заранее известно, что значение некоторой переменной лежат внутри некоторого диапазона.

Перечислимые типы – в Паскале можно определить новый тип путём явного перечисления всех возможных его значений, причём каждое значение определяется только именем и является перечислимым. Синтаксическая диаграмма следующая.

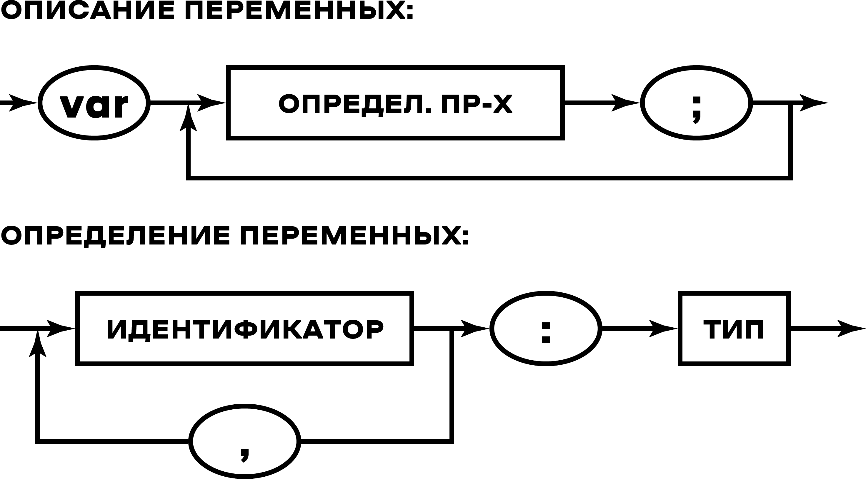


*Замечание.* Над перечислимыми типами определены операции сравнения. Считается, что значение перечислимого типа, указанном в списке, в порядке возрастания.

*Замечание.* Имена из списка, перечислимого типа, считаются константами соответствующего типа. Эти идентификаторы должны быть уникальны в пределах блока. Блоком считается программа или подпрограмма и вложенные в неё программы. Недопустимы описания двух и более перечислимых типов с совпадающими константами.

# ОПИСАНИЕ ПЕРЕМЕННЫХ, КОНСТАНТ И ТИПОВ ПЕРЕМЕННЫЕ

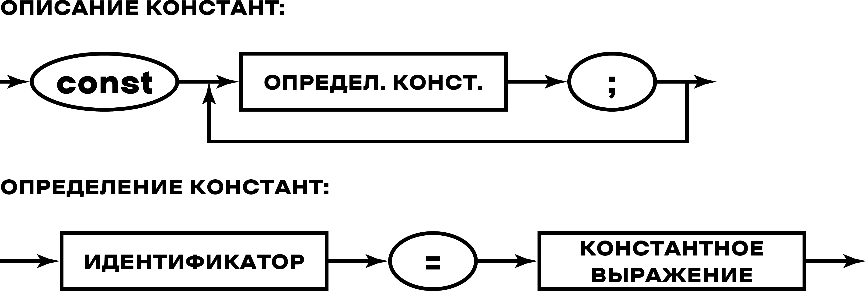
Описание переменной начинается с служебного слова var, далее определение переменной должно содержать два элемента: имя переменной и её тип. Имя переменной задаётся посредством идентификатора. Идентификатор переменной и её тип разделяются символом двоеточия. Определение завершается точкой с запятой.



*Замечание.* Каждая переменная, описанная в блоке, должна упоминаться не более одного раза.

# КОНСТАНТЫ

Описание констант начинается со служебного слова const, за которым располагается последовательность определений константы.

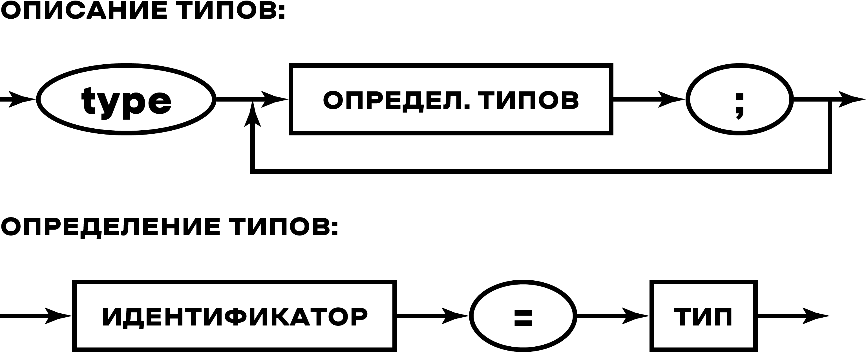


*Замечание.* Тип константы определяется по её значению.

Константное выражение может строиться из чисел, строк, идентификаторов констант и некоторых стандартных функций (например abs).

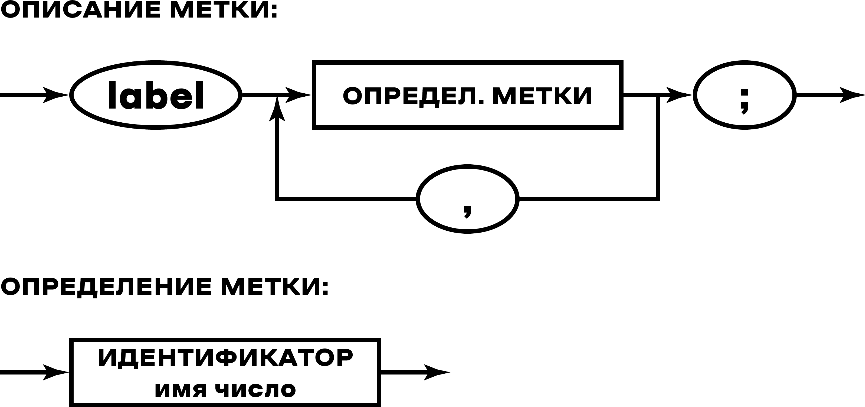
# ТИПЫ

Новый тип может быть указан явно при описании переменной, либо определён отдельно, то есть типу может быть сопоставлено некоторое имя, которое будет использоваться в дальнейшем для описания переменных. Второй способ более предпочтителен, поскольку способствует большей ясности программы.



# МЕТКИ

Метка представляет собой либо идентификатор, либо целое число и может использоваться совместно с оператором перехода label. Она помечает оператор, на который осуществляется переход управления. Метка отделяется от этого оператора двоеточием.



# ПРЕОБРАЗОВАНИЯ И СОВМЕСТИМОСТЬ ТИПОВ

Стандартные функции преобразования типов

|  |  |  |
| --- | --- | --- |
| **Функция** | **Аргумент** | **Тип результата** |
| Odd | Целый | Выдаёт true, если аргумент нечётный и false, если аргумент чётный |
| Trunc | Вещественный | Выдаёт целую часть аргумента (дробная часть отбрасывается) |
| Round | Вещественный | Выдаёт аргумент, округлённый до ближайшего целого |
| Ord | Дискретный | Результат – целый порядковый номер (нумерация с 0) значения аргумента в данном  дискретном типе |
| Chr | 0…255 | Выдаёт символьное значение с заданным в аргументе порядковым номером |

# ЭКВИВАЛЕНТНОСТЬ ТИПОВ

Два типа T1 и T2 являются эквивалентными, если выполняется одно из двух условий: T1 и T2 представляют собой один и тот же идентификатор типа; тип T2 описан с использованием T1 с помощью равенства следующего или цепочки равенств.

*Замечание.* Эквивалентность типов требуется при передаче параметров при вызове функций и процедур. Типы формальных и фактических параметров должны быть эквивалентны.

# СОВМЕСТИМОСТЬ ТИПОВ

Выполнение операций в выражениях и операций сравнения требуют, чтобы операнды имели совместимые типы. *Совместимость двух типов* имеет место, если выполняется хотя бы одно из следующих условий:

* Два типа эквивалентны.
* Оба типа являются либо целыми, либо оба типа являются вещественными.
* Один тип является ограниченным типом, причём его базовым типом является другой тип. (type T1 = shortint; T2 = 1…100;)
* Оба типа являются ограниченными, причём их базовым типом является один и тот же тип. (type T1 = 10…100; T2 = 1…5;)
* Один тип является строковым, а другой тип либо строковым, либо символьным типом.

# СОВМЕСТИМОСТЬ ПО ПРИСВАИВАНИЮ

Оператор присваивания считается корректным, если тип переменной в левой части совместим по присваиванию с типом выражения в правой части.

Два типа являются *совместимыми по присваиванию*, если выполняется хотя бы одно из следующих условий:

* Оба типа эквивалентны.
* Оба типа являются совместимыми дискретными типами и значения типа правой части попадает в диапазон возможных значений левой части.
* Оба типа являются вещественными типами и значения типа правой части попадают в диапазон возможных значений типа левой части.
* Тип левой части является вещественным, а тип правой части является целым типом.
* Тип левой части является строковым типом, а тип правой части – либо строковым типом, либо символьным типом.

Если не выполняется ни одно из этих условий при компиляции выдаётся сообщение об ошибке.

# ОПЕРАТОРЫ

Операторы – синтаксические конструкции, предназначенные для записи алгоритмических действий по преобразованию данных и для задания порядка выполнения других действий.

В Паскале имеются следующие операторы: оператор присваивания, оператор процедуры, оператор перехода, составной оператор, оператор варианта, оператор цикла с предусловием, оператор цикла с постусловием, оператор цикла с параметром, пустой оператор.

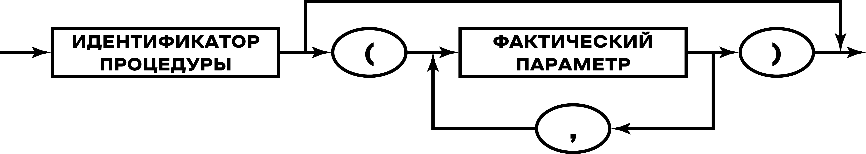
# ОПЕРАТОР ПРИСВАИВАНИЯ

Оператор присваивания предназначен для вычисления нового значения некоторой переменной, а также для определения значения, возвращаемого функцией.



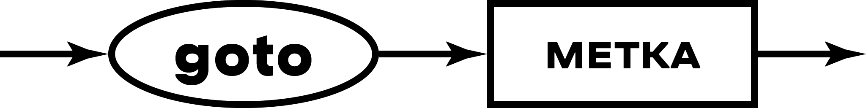
# ОПЕРАТОР ПРОЦЕДУРЫ

Задаёт значение операторов определённой процедуры. После завершения выполнения процедуры, управление передаётся на оператор, следующий за оператором процедуры.



# ОПЕРАТОР ПЕРЕХОДА

Оператор безусловного перехода вызывает передачу управления оператору, которому преследует метка.



*Замечание.* Метка, указанная в операторе перехода, должна помечать оператор, находящийся в том же блоке, что и сам оператор перехода. Таким образом не допускаются переходы из процедур и функций или внутрь их.