Способы представления информации в ЭВМ

Вся информация в компьютере представляется в двоичном виде.

Наименьшая единица памяти называется бит, который может принимать значения 0 и 1. Бит – основной строительный блок памяти, АЛУ и ЦПУ.

Наименьшая адресуемая единица памяти и более удобный ее элемент – байт. Байт состоит из 8 бит. Т.к. каждый бит может принять значение 0 и 1, то 8 бит могут представить 256 (2^8) комбинаций из 0 и 1.

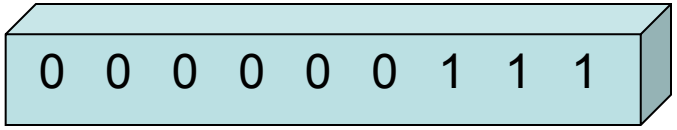
Способы представления информации в ЭВМ(2)

Для удобства обработки, чтения и записи информации байты могут объединяться в слова (2 байта), двойные слова (4 байта) и т.д.

Информация, с которой работает пользователь, бывает числовой, символьной, аудио, видео и т.д.

Для представления числовой информации используются целые и вещественные числа.

Целое число не имеет дробной части (2, -45, 789). Представив целое число в двоичном виде, его нетрудно разместить в памяти. Например, число 7 – это 111.

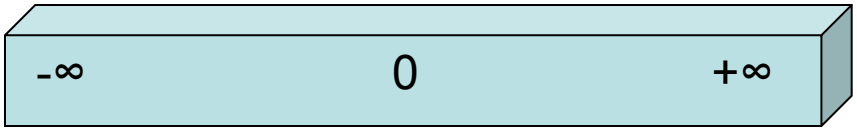


2^2 2^1 2^0

4+2+1 = 7

Способы представления информации в ЭВМ(3)

Целое число



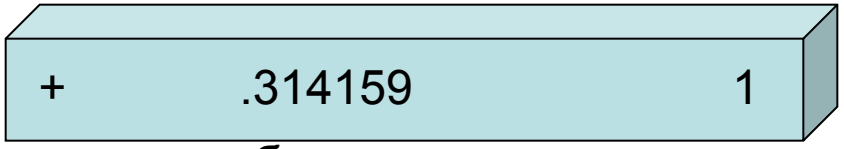
Целые числа могут быть положительными (без знака) и отрицательными (со знаком).

Для хранения знака используется один двоичный разряд (старший). Целые числа являются дискретной информацией и в машине представляются точно.

Вещественные числа – это разновидность аналоговой информации. Включают в себя числа, расположенные между целыми.

В машине такие числа представляются в двоичном виде с определенной точностью. Это связано со схемой размещения и обработки в памяти вещественного числа.

Способы представления информации в ЭВМ(4)



$+ \quad .314159 \quad \times 10^{1} \quad = \quad 3.14159$

Вещественное число представляется в форме числа с плавающей точкой. Формирование представления такого числа состоит в его разбиении на дробную часть и порядок, которые затем размещаются в памяти ($7.5 - 0.75 \times 10^{1}$).

Для размещения чисел в памяти используются двоичные числа и степени двойки вместо степеней десяти. Поэтому точно можно представить только дроби, являющиеся степенями 2.

Однако, такое разбиение дает возможность представить число несколькими способами, например 75×10^{-1}, 7.5×10^{0}, 0.75×10^{1}.

Способы представления информации в ЭВМ(5)

Символьная информация представляется двоичным кодом, который может быть не более 8 двоичных разрядов (1 байт) в соответствии с таблицей кодировки и может содержать коды 256 символов.

Так символ А представляется кодом 65,

символ 0 кодом 48,

символ 9 кодом 58,

символ е кодом 101.

1.4 Программы и алгоритмы

Основное назначение компьютера – обработка информации, для чего необходимо выполнить определенный набор операций - **программу**.

Программа – набор инструкций, описывающих последовательность действий, приводящих к результату.

Программу можно написать на машинном языке, однако это требует высокой квалификации программиста.

Для возможности написания программы пользователем непрограммистом используют специальные языки называемые языками программирования (Бэйсик, Паскаль, С и т.д.).

Программа на языке программирования преобразуется в машинные команды, которые затем выполняются компьютером.

Программы и алгоритмы (2)

Однако, чтобы составить программу, необходимо хорошо представлять себе, что нужно сделать, чтобы решить какую либо задачу.

Алгоритм – это конечная последовательность четко определенных действий, задающая обработку исходных данных с целью получения нужного результата.

1.4.1 **Свойства алгоритмов**

1. Массовость (обеспечение функций алгоритма для большой совокупности данных)
2. Дискретность (возможность представить алгоритм в виде отдельных последовательных шагов)
3. Определенность (каждый шаг алгоритма должен быть четко определен и однозначно понятен)

Свойства алгоритмов(2)

4. Результативность (получение нужного результата)
5. Конечность (выполнение алгоритма за конечное число шагов)

1.4.2 **Способы представления алгоритма**

1. Описательная форма (на естественном языке)
2. Псевдокод (описательная форма с ограниченным числом элементов)
3. Графическая форма (схема алгоритма)
4. Табличная форма (таблицы решений)

Основные конструкции псевдокода

Псевдокод:

1. Следование

…
Действие 1
Действие 2
…

2. Ветвление

…
Если Условие
    то Действие 1
    иначе Действие 2
Все-если
…

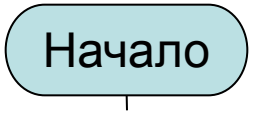
3. Цикл-пока

…
Цикл-пока Условие
    Действие
Все-цикл
…

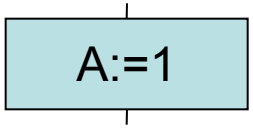
Схемы алгоритмов

Обозначения ГОСТ 19.701 – 90

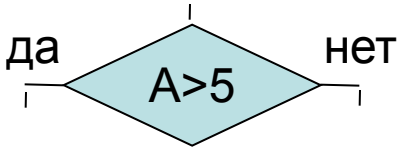
1. Терминатор (начало/конец)



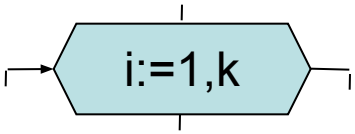
2. Процесс (вычисления)



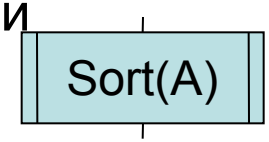
3. Анализ (проверка)



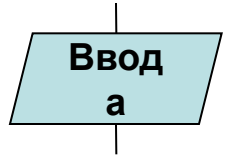
4. Модификатор (автоматическое изменение)



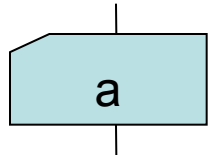
5. Предопределенный процесс (подпрограмма)



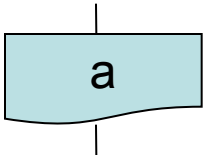
6. Ввод/вывод данных



7. Ввод с перфокарт



8. Вывод на принтер



9. Комментарий

10. Соединитель

Правила выполнения схем алгоритмов

Схемы алгоритмов должны быть выполнены аккуратно, желательно с применением карандаша и линейки или графических редакторов на компьютере.

Стрелки на линиях, идущих сверху вниз и слева направо, т. е. в направлении нашего письма не ставят, чтобы не затенять схему.

Если линия – ломанная, и направление ее хотя бы в одном сегменте не совпадает со стандартными, то стрелка ставится в конце линии, перед блоком, в который она входит.

Если схема не умещается на странице или линии многократно пересекаются, то линии разрывают. При этом один соединитель ставится в месте разрыва, второй – в месте продолжения линии. Оба соединителя помечаются одной и той же буквой или цифрой.

Для простоты чтения схемы ее начало должно быть сверху, а конец – снизу. При этом количество изгибов, пересечений и обратных направлений линий должно быть минимальным.

Таблицы решений

Таблица составляется следующим образом.

В столбик выписываются все условия, от которых зависят дальнейшие вычисления, а по горизонтали - все случаи для вычислений.

На пересечении каждого столбца и строки ставят букву Y, если для данного решения данное условие должно выполняться, букву N, если данное условие обязательно должно не выполняться, и прочерк, если исход сравнения не важен.

Например, для алгоритма вычисления корней квадратного уравнения можно составить следующую таблицу:

Таблицы решений(2)

	Нет корней	$x = -b / 2a$	$x=\pm(-b\sqrt{D}) / 2a$
$D < 0$	Y	N	N
$D = 0$	N	Y	N
$D > 0$	N	N	Y

Иногда, составленная таблица может иметь довольно сложный вид. Рассмотрим, например, таблицу:

	P1	P2	P3	P4
Условие 1	Y	-	N	Y
Условие 2	N	Y	N	N
Условие 3	Y	-	-	N

Таблицы решений(3)

Если строго придерживаться заданного порядка проверки условий, то получится довольно сложный алгоритм и его построение вызывает определенные трудности.

Но этот алгоритм можно значительно упростить, если в таблице поменять местами проверяемые условия, а также для удобства построения алгоритма поменять местами столбцы таблицы. Если преобразовать таблицу следующим образом:

	P1	P4	P3	P2
Условие 2	N	N	N	Y
Условие 1	Y	Y	N	-
Условие 3	Y	N	-	-

Часть 2. Основы алгоритмизации и процедурное программирование
Введение. Этапы создания ПО

1. **Постановка задачи** – неформальное описание задачи
2. **Анализ и уточнение требований** – формальная постановка задачи и выбор метода решения
3. **Проектирование** – разработка структуры ПО, выбор структур данных, разработка алгоритмов, определение особенностей взаимодействия с программной средой
4. **Реализация** – составление программ, тестирование и отладка
5. **Модификация** – выпуск новых версий

Пример разработки программы

1. **Постановка задачи**: Разработать программу, которая определяет наибольший общий делитель двух целых чисел.
2. **Анализ и уточнение требований:**

   1) *Функциональные требования*

   исходные данные: a, b – натуральные числа; 0 < a, b < **?** ;

   результат: x – натуральное число, такое, что

$x = \max \{y_i \,/\, i = \overline{1,n}\}$, где $((a \bmod y_i) = 0) \,\&\, (b \bmod y_i) = 0)$

   Метод решения:

   а) найти делители Y = { y_i } и определить x = max {Y};

   б) метод Евклида

Пример 1:

a	b
24	18
6	18
6	12
6 =	6

Пример 2:

a	b
3	4
3	1
2	1
1 =	1

Пример разработки программы (2)

2) *Эксплуатационные требования*:

а) процессор – не ниже Pentium;

б) операционная система – Windows XP (консольный режим);

в) предусмотреть запрос на ввод данных с клавиатуры;

г) результаты вывести на экран дисплея.

3) *Технологические требования*:

а) язык программирования: C++;

б) среда программирования: Microsoft Visual Studio .Net 2003;

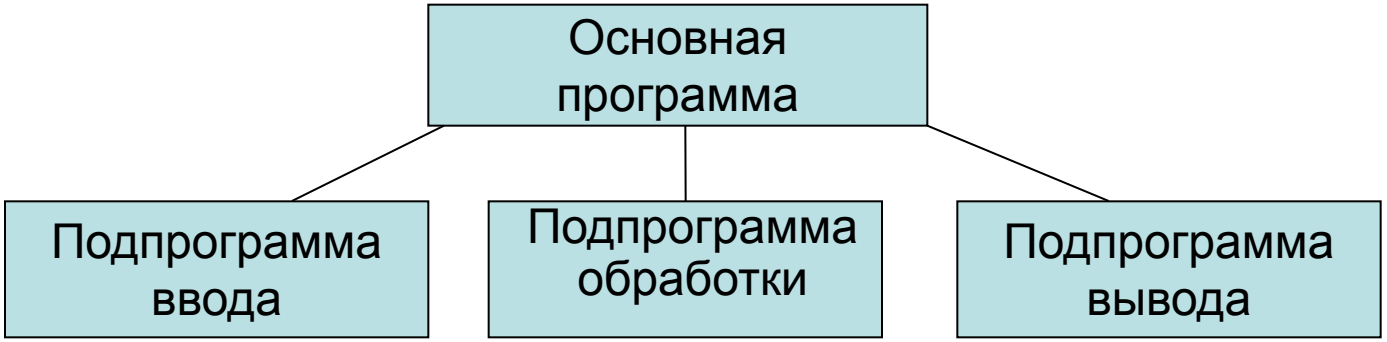
в) технология: структурный подход.

Пример разработки программы(3)

3. Проектирование

Виды проектной документации:

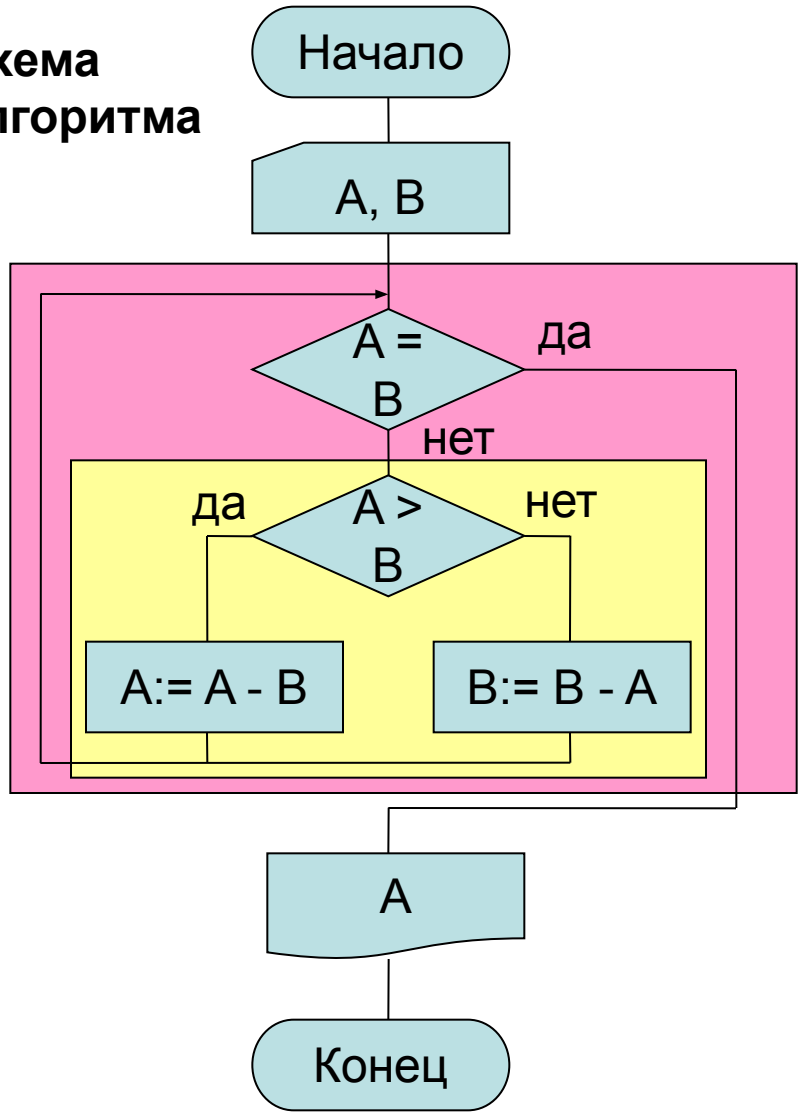
Структурная схема ПО – показывает взаимодействие по управлению основной программы и подпрограмм.



Алгоритм основной программы и подпрограмм в соответствии с выбранным способом представления

Пример разработки программы (4)

Схема алгоритма



Алгоритм на псевдокоде

Начало

Ввести A,B

Цикл-пока A ≠ B

Если A > B

то A := A – B

иначе B := B – A

Все-если

Все-цикл

Вывести A

Конец

4. Реализация программы, ее тестирование и отладка.

Схема процесса подготовки программы

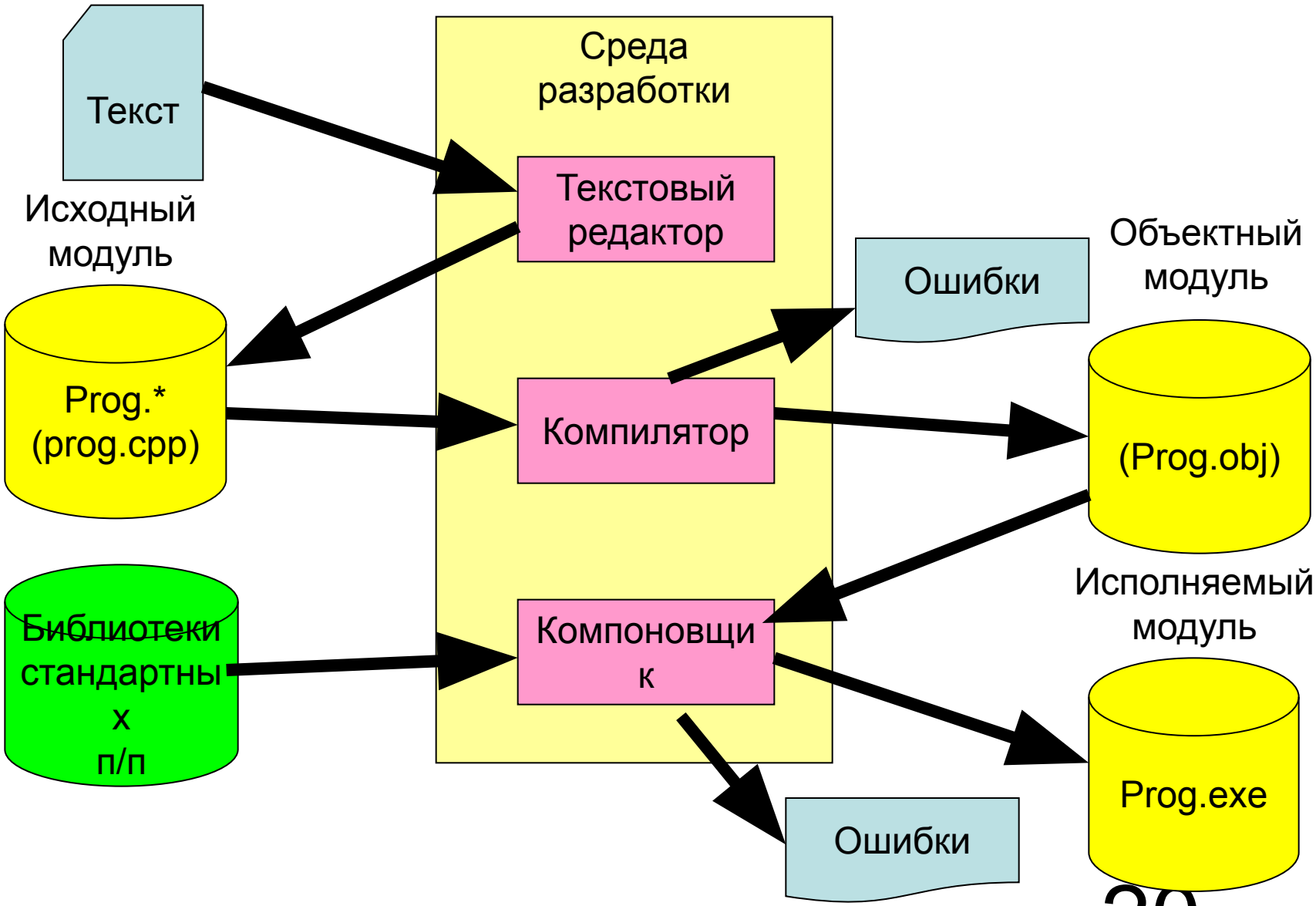


Схема процесса отладки и выполнения