Программирование на языке Си

Литература

1. **Б.И. Березин, С.Б. Березин. Начальный курс С и С++. М.: Диалог МИФИ, 1996.**

2. **А.В. Крячков, И.В. Сухинина , В.К. Томшин Программирование на С и С++. М.: Телеком, 2000.**

3. **Е.Р. Алексеев Учимся программировать на Microsoft Visual C++ и Turbo C++ Explorer. М.: NT Press, 2007.**

4. **В.В. Подбельский, С.С. Фомина. Программирование на языке Си. — М.: Финансы и статистика, 2003.**

5. **Б. Керниган, Д. Ритчи Язык Си. М.: Финансы и статистика, 1985.**

Языки программирования

- **Машинно-ориентированные (низкого уровня)** - каждая команда соответствует одной команде процессора (ассемблер)
- **Языки высокого уровня** – приближены к естественному (английскому) языку, легче воспринимаются человеком, **не зависят от конкретного компьютера**
  - *для обучения*: Бейсик, ЛОГО, Паскаль
  - *профессиональные*: Си, Фортран, Паскаль
  - *для задач искусственного интеллекта*: *Пролог, ЛИСП*
  - *для Интернета*: *JavaScript, Java, Perl, PHP, ASP*

Язык Си

1972-1974 – Б. Керниган, Д. Ритчи

- высокая скорость работы программ
- много возможностей
- стал основой многих современных языков *(C++, C#, Javascript, Java, ActionScript, PHP)*

- много шансов сделать ошибку, которая не обнаруживается автоматически

Простейшая программа

главная (основная) программа всегда имеет имя ***main (_tmain)***

```
main()
{

}
```

начало программы

«тело» программы (основная часть)

конец программы

Что делает эта программа?

Что происходит дальше?

текст программы на Си или Си++

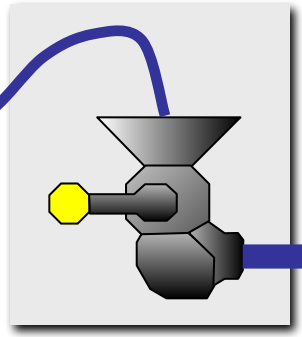
first.cpp

```
main()
{

}
```

исходный файл

транслятор

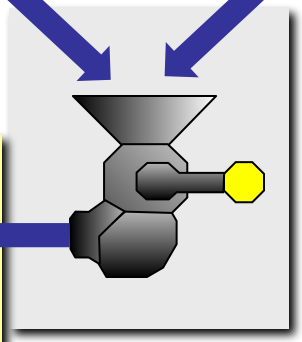


first.o

перевод исходного файла в машинные коды построение *объектного файла*

объектный файл

стандартные функции



редактор связей (компоновка)

first.exe

Компоновщик подключает стандартные функции

файл с расширением ***.exe**, готовая программа

исполняемый файл

- **по исходному файлу можно восстановить остальные**
- **исполняемый файл можно запустить**

Вывод текста на экран

include (= включить)

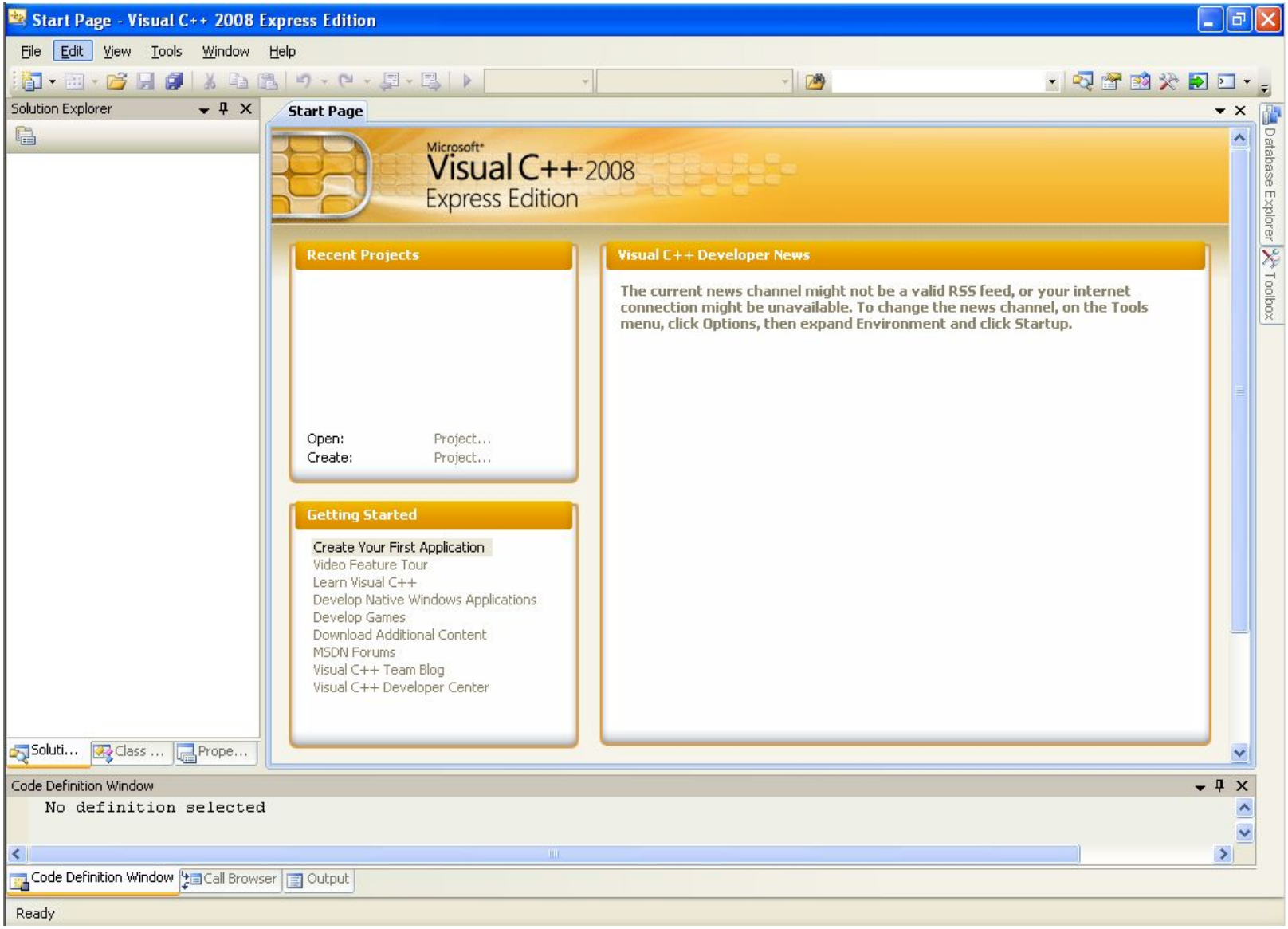
файл ***stdio.h***: описание стандартных функций ввода и вывода

_tmain( )

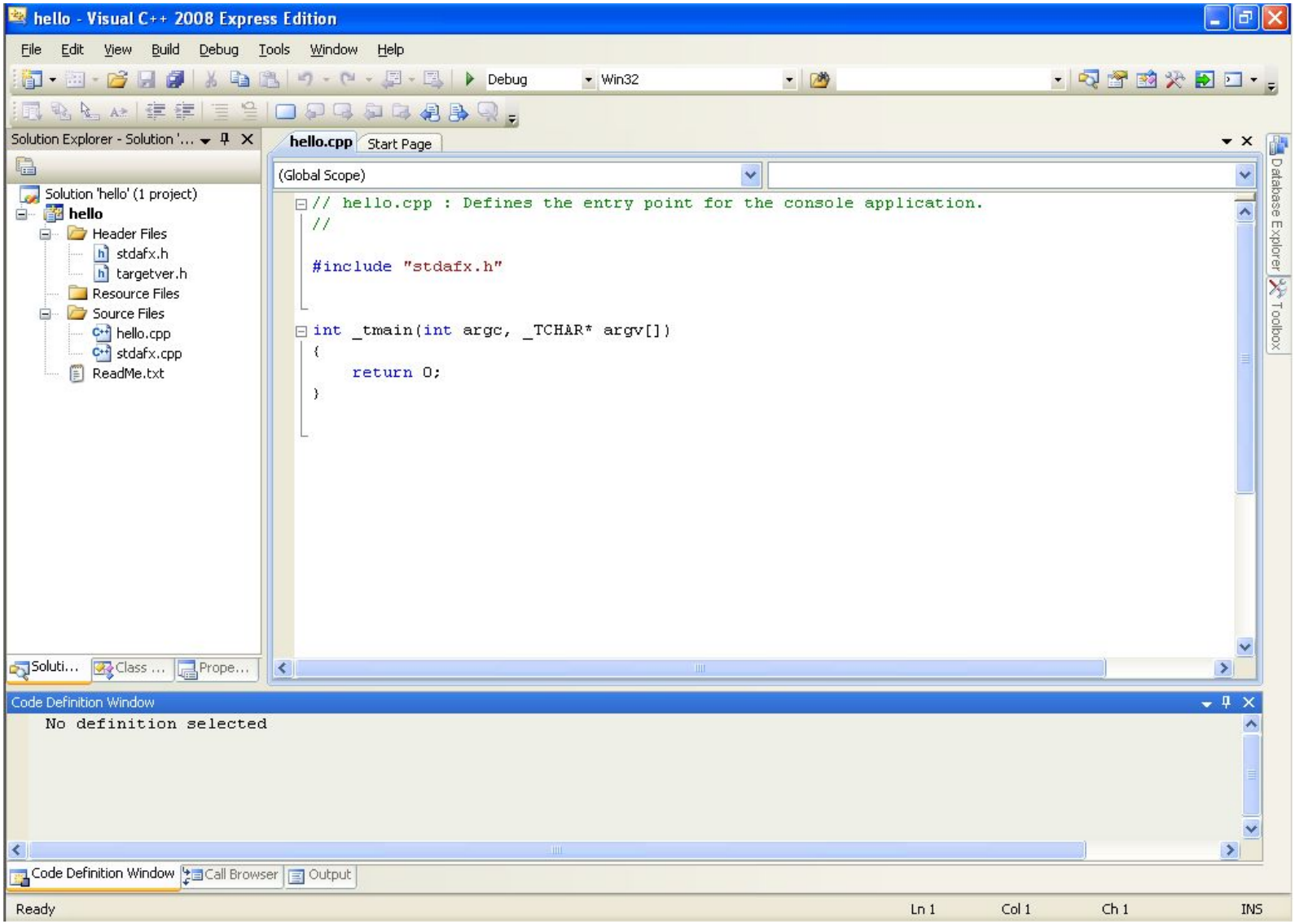
```
#include <stdio.h>
main()
{
printf("Привет!");
}
```

вызов стандартной функции ***printf*** (= *print format*) (форматный вывод)

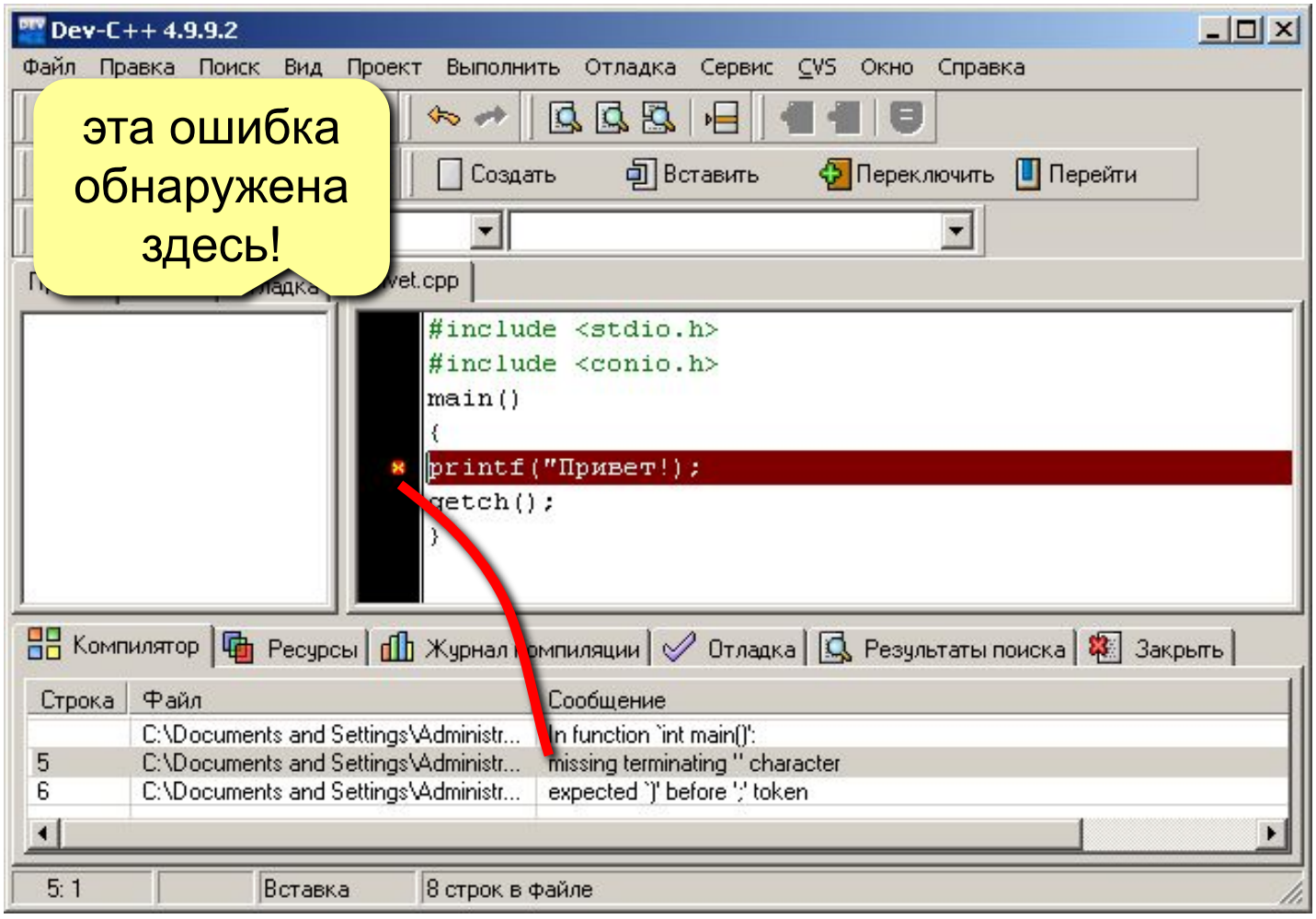
этот текст будет на экране


Start Page - Visual C++ 2008 Express Edition
File Edit View Tools Window Help
Solution Explorer
Start Page
Microsoft
Visual C++ 2008
Express Edition
Recent Projects
Open: Project...
Create: Project...
Visual C++ Developer News
The current news channel might not be a valid RSS feed, or your internet connection might be unavailable. To change the news channel, on the Tools menu, click Options, then expand Environment and click Startup.
Getting Started
Create Your First Application
Video Feature Tour
Learn Visual C++
Develop Native Windows Applications
Develop Games
Download Additional Content
MSDN Forums
Visual C++ Team Blog
Visual C++ Developer Center
Soluti...
Class ...
Prope...
Database Explorer
Toolbox
Code Definition Window
No definition selected
Code Definition Window
Call Browser
Output
Ready

Основное окно ИС



Где ошибки?



Ошибка может быть в конце предыдущей строки!

Наиболее «популярные» ошибки

xxx.h: No such file or directory	не найден заголовочный файл 'xxx.h' (неверно указано его имя, он удален или т.п.)
'xxx‘ undeclared (first use this function)	функция или переменная 'xxx' неизвестна
missing terminating " character	не закрыты кавычки "
expected ;	нет точки с запятой в конце оператора **в предыдущей строке**
expected }	не закрыта фигурная скобка

Ждем нажатия любой клавиши

```
#include <stdio.h>
#include <conio.h>
main()
{
printf("Привет!"); // вывод на экран
getch();   /* ждать нажатия клавиши */
}
```

файл ***conio.h***: описание функций для работы с клавиатурой и монитором

комментарий до конца строки

ждать нажатия на любую клавишу

комментарий между `/*` и `*/`

Переход на новую строку

```
#include <stdio.h>
#include <conio.h>
main()
{
printf("Привет,\n Вася!");
getch();
}
```

последовательность

\n (код 10)

переход на новую строку

на экране:

```
Привет,
Вася!
```

Тема 1 Арифметические и логические основы ЭВМ.

1.Определения

Электронная вычислительная машина (ЭВМ) - устройство, способное хранить и выполнять программы.

Никлаус Вирт (разработчика языка Паскаль)

> **Алгоритмы + структуры данных = программы**

Структуры данных - исходные данные, промежуточные и конечные результаты.

Алгоритмы - указания о том, какие действия и в какой последовательности необходимо применять к данным для получения требуемого конечного результата.

2.Арифметические основы

2.1. Система счисления

- совокупность приемов и правил для записи чисел цифровыми знаками.

– совокупность правил и приемов для обозначения и именования чисел

Различают непозиционные и позиционные системы счисления.

- ***В непозиционной системе счисления* значение знака (символа) не зависит от его положения в числе. Пример - римская система счисления.**

Например, VIII, XIX

- ***Позиционная система счисления* - система, в которой значение цифры числа определяется ее положением (позицией) в числе.**

Например,

101010, 200, 2000

Непозиционная – это система счисления, в которой количественный эквивалент числа (значение каждого символа) не зависит от его положения в коде числа, т.е. количественный эквивалент числа определяется конфигурацией символов.

Пример: римские числа: I, II, III, IV, V, VI.

X – 10

L – 50

C- 100

D-500

M - 1000

Позиционная система счисления – система, в которой количественный эквивалент, значение символа, зависят от его места (позиции) в коде числа.

Базис (основание) – количество символов, которые используются для изображения числа в разрядах данной системы счисления.

Любое действительное число «*A*» записанное в однородной позиционной системе счисления может быть представлено в виде суммы степенного ряда по базису «q»

$$A_{(q)} = a_n \cdot q^n + \ldots + a_1 \cdot q^1 + a_0 \cdot q^0 + a_{-1} \cdot q^{-1} + \ldots + a_{-m} \cdot q^{-m}$$

$$= a_n \ldots a_1 a_0 , a_{-1} \cdots a_{-m} = \sum_{i=-m}^{n} a_i \cdot q^i \qquad (1)$$

- **<u>Однородная система счисления</u> – система счисления только с одним базисом.**
- **<u>Неоднородная (смешанная) система счисления</u> – несколько базисов**

  (например, измерение времени):
- **<u>Унарная система счисления</u>** – в которой для записи чисел применяется только один вид знаков.

  Например, счетные палочки.

Десятичная ПСС:

q=10

123,456 =

$= \mathbf{1 * 10^{+2} + 2 * 10^{+1} + 3 * 10^{0} + 4 * 10^{-1} + 5 * 10^{-2} + 6 * 10^{-3}}$

Двоичная ПСС: q=2

$$A_{(2)} = 1101{,}001_{(2)}$$

В вычислительной технике, в основном, используются

- ***двоичная,***
- ***восьмеричная и***
- ***шестнадцатеричная*** **системы счисления**

(восьмеричная и шестнадцатеричная - для более компактной записи двоичных кодов).

В двоичной системе счисления для записи числа в сокращенной форме используются цифры О и 1,

- **в восьмеричной - О, 1,2, ..., 7,**
- **в десятичной - О, 1,2, ..., 9**
- **в шестнадцатеричной - О, 1,2, ..., 9, A,B, C, D, E, F.**

Перевод чисел из 2-ичной, 8-ричной или 16-ричной систем счисления в 10-чную **систему легко выполняется с помощью развернутой формы записи числа (1).**

Например,

$$A_{(2)} = 1101{,}001_{(2)}$$

$$A = 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^0 + 1 \cdot 2^{-3} = 13{,}125$$

- Алгоритм перевода из одной системы счисления в другую.

3. Представление информации (данных) в ЭВМ.

Структурные единицы информации

Мин. объем информации – 1 бит:

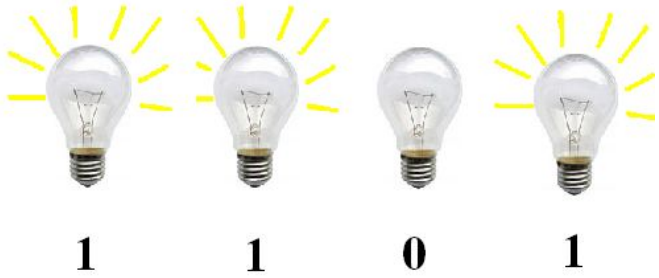
1bit (**bi**nary digi**t** - однозначное число- цифра)

-мин единица информации, представленная одним двоичным элементом (биполярным элементом - триггером).

Основным логическим элементом - **электронный триггер** – 2-х устойчивый (биполярный) элемент.

Например, условно – лампочка или один двоичный разряд (0 или 1).

4 разряда=4 бита



Можно представить все целые (натур.) числа от **0** до **15 (= 2^4 - 1).**

10-тичное	2-ичное
0	0
1	1
2	10
3	11
4	100
…	…
14	1110
15	1111

С помощью 5 битов – числа до

$$2^5 - 1 = 31_{10} = 11111_2$$

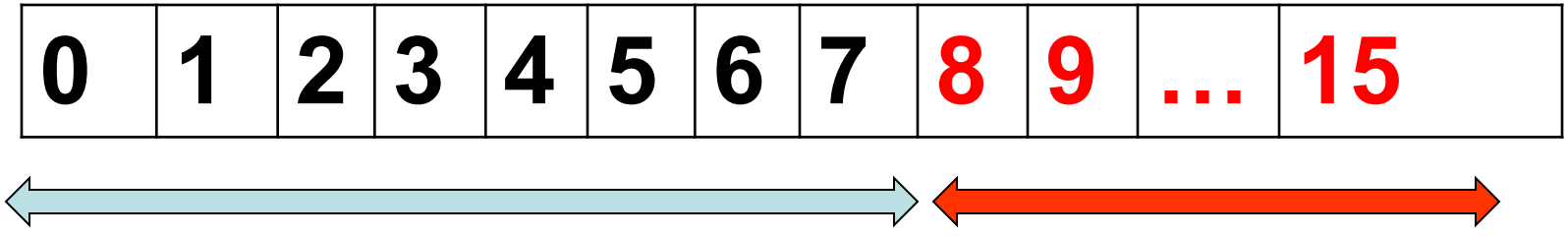
Если **n** битов - числа до

$$(2^n - 1).$$

Чтобы хранить более емкую информацию, двоичные элементы объединяются в группы.

Поле – последовательность битов, имеющих определенный смысл.

- 1 Байт – 1Byte – поле длиной 8 бит.
- Байт, как правило, - мин. (неделимая) единица информации, с которой оперирует ЭВМ.
- Адрес бита (байта) – порядковый номер. **Нумерация начинается с 0.**



Байты U - в более крупные единицы памяти, называемые **ячейками.**

Для каждого класса ЭВМ определена характерная длина ячейки.

Такая ячейка наз. **машинным словом-**

4 B = 32 bit

полуслово 2 B = 16 bit

двойное слово 8 B = 64 bit

Станд. размер ячейки – разрядность машины.

<table>
<tr><th colspan="8">Схема форматов</th></tr>
<tr><td>B0</td><td>B1</td><td>B2</td><td>B3</td><td>B4</td><td>B5</td><td>B6</td><td>B7</td></tr>
<tr><td colspan="2">полуслово</td><td colspan="2"></td><td colspan="2"></td><td colspan="2"></td></tr>
<tr><td colspan="4">слово</td><td colspan="4"></td></tr>
<tr><td colspan="8">двойное слово</td></tr>
</table>

Последовательность маш.слов U **в более крупные структурные ед-цы:**

- **Страницы –** фиксированный размер
- **Сегменты -** переменный размер

1 К[Byte] =1024 B [yte] = 2^{10} bit

1M[Byte] =1024*1024 B [yte] = 2^{20} bit

1G[Byte] =1024*1024*1024 B [yte] = 2^{30} bit

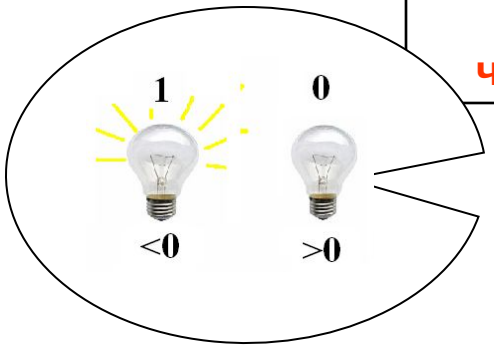
и т.д.

4.Представление данных

4.1. Целые числовые данные

Вариант 1 (полуслово **2B=16 b)**

0b	1b	2b	…	14	15b
Sign числа					

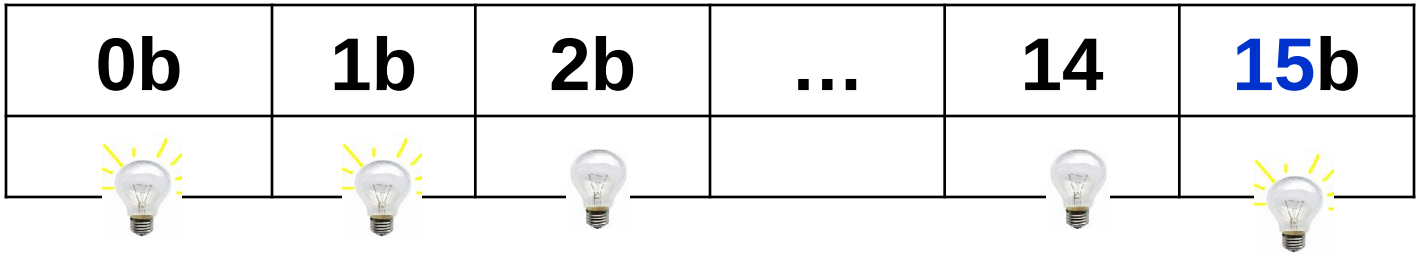


Max по модулю число
$2^{15}-1$ = 32 768

- 32767 …0… 32768

```
short int A; // Объявление переменной с именем
A=2;   // Инициализация переменной
short int A=2; //Или определение переменной (Объявление+
Инициализация)
```

Вариант 2 (полуслово **2B=16 b) – только положительные (беззнаковый)**



Max по модулю число

$$2^{16} - 1 = 2*32\ 768$$

```
unsigned short int UA; // Объявление переменной
UA=2;   // Инициализация переменной
unsigned short  int UA=2; //Или определение переменной
          //(Объявление+ Инициализация)
```

Вариант 3 (слово **4B=32 b)**

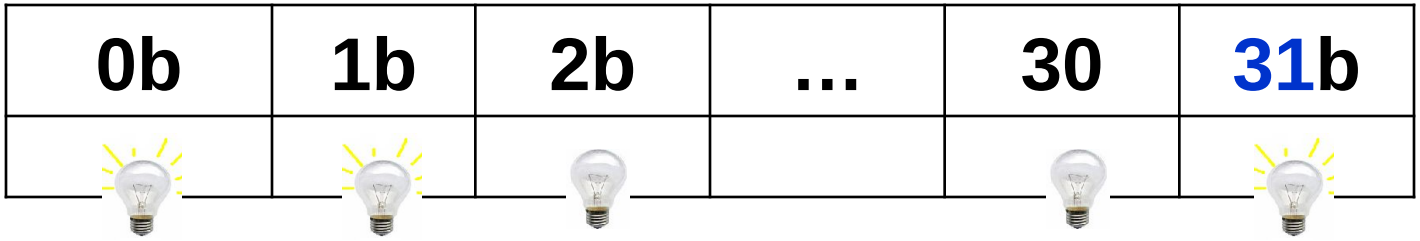
0b	1b	2b	…	30	31b
sign					

Max по модулю число

$$2^{31} - 1$$

```
long int LA; // Объявление переменной с именем
LA=2;        // Инициализация переменной
long int LA=2; //Или определение переменной
        //(Объявление+ Инициализация)
```

Вариант 4 (слово 4**B=32 b) – только положительные (беззнаковый)**



Max по модулю число

$$2^{32} - 1$$

```
unsigned long int ULA; // Объявление переменной
ULA =2;   // Инициализация переменной
unsignet long int ULA =2; //Или определение переменной
          //(Объявление+ Инициализация)
```

Тип	Размер памяти в байтах	Диапазон значений
char	1	от -128 до 127
short int **int** **int**	IBMAT,AT,SX,DX 2 4	от -32768 до 32767 **от -2 147 483 648 до 2 147 483 647**
long int	4	от -2 147 483 648 до 2 147 483 647
unsigned char	1	от 0 до 255
unsigned short int	2	от 0 до 65535
unsigned long int	4	от 0 до 4 294 967 295

4.2. Вещественные числа

2 формы представления

- **с фиксир. десятичной точкой**

  123.456

- **с плавающей десятичной точкой** , т.е. нормализованной форме

  $123.456 = 1.23456 * 10^2 =$

  $= 0.123456 * 10^3$

$$N = mq^p$$

m - мантисса числа
p - порядок числа

с фиксир. десятичной точкой

123.456

0b	1b	. . .	i-1	i	i+1	…	31b
Sign	Целая часть				Дробная часть		

с плавающей десятичной точкой ,

$$N = mq^{p}$$

123.456 =1.23456 *10 ^{2}=

= 0.123456* 10^{3}

m - мантисса числа $1<m<2$ **или** $m<1$

p - порядок числа – указывает местоположение в числе ●, отделяющую целую часть от дробной

0b	1b	…	8b	9b	…	31b
Sign	Порядок числа p			Мантисса m		

Max P = $2^{8-1} - 1 = 256-1$

Max m = $2^{23} - 1 = 2^{10}\ 2^{10}\ 2^{3}-1 =$

1024 * 1024* 8 -1

```
float f; // Объявление переменной
f=2.0;   // Инициализация переменной
float f =2.0; //Или определение переменной
              //(Объявление+ Инициализация)
```

0b	1b	…	11	12	…	63b
Sign	Порядок числа p			Мантисса m		

Max P = $2^{11-1} - 1$

Max m = $2^{42} - 1$

```
double df; // Объявление переменной df=2.0;
           // Инициализация переменной
double df =2.0; // определение переменной
           //(Объявление+ Инициализация)
```

4.3. символьные данные

Код ASCII

(American Standart Code for Information Interchange)

Любой символ – 1 байт

ПРИМЕРЫ

char symbol=‘A’;

char c=65;

Символы можно «сравнивать»

Таблицы кодов ASCII – см.

Символ	2-ичный	10-чный
0	0011 0000	0
1	0011 0001	1
2	0011 0010	2
…	….	
9	0011 1001	9
…		
A	0100 0001	65
…		…
Z	0101 1010	90
…	….	….
a	0110 0001	97
…		
z	0111 1010	122

Тема 2. Алгоритм

Название **"алгоритм"** произошло от латинской формы имени величайшего среднеазиатского математика **Мухаммеда ибн Муса ал-Хорезми** (Alhorithmi), жившего в 783—850 гг. В своей книге "Об индийском счете" он изложил правила записи натуральных чисел с помощью арабских цифр и правила действий над ними "столбиком", знакомые теперь каждому школьнику. В XII веке эта книга была переведена на латынь и получила широкое распространение в Европе.

Алгоритм — заранее заданное понятное и точное предписание возможному исполнителю совершить определенную последовательность действий для получения решения задачи за конечное число шагов.

Алгоритм

Алгоритм – это четко определенный план действий для исполнителя.

1. Свойства алгоритма

- **дискретность**: состоит из отдельных шагов (команд)
- **понятность**: должен включать только команды, известные исполнителю (входящие в СКИ)
- **определенность**: при одинаковых исходных данных всегда выдает один и тот же результат
- **конечность**: заканчивается за конечное число шагов
- **массовость**: может применяться многократно при различных исходных данных
- **корректность**: дает верное решение при любых допустимых исходных данных

Что такое **АЛГОРИТМ?** Как его записать?

Аль Хорезми

Структурное программирование

Дейкстра

Ветвления

if - then - else

switch

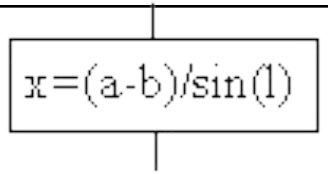
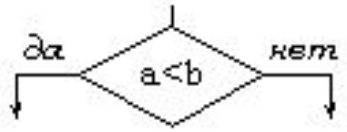
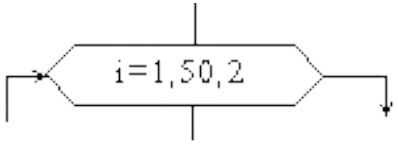
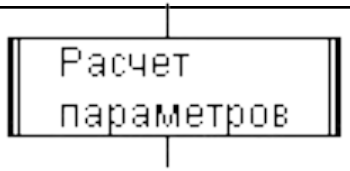
Следование

Циклы

while

do - while

for

Название	Обозначение и пример заполнения	Пояснение
Процесс		Вычис. действие
разветвление		Проверка условий
Модификация		Начало цикла с известным числом повторений
Предопреде-ленный процесс		Вычисление по подпрограмме
Ввод-вывод		Ввод-вывод
Пуск-останов		Начало-конец

Программа

Программа – это

- алгоритм, записанный на каком-либо языке программирования
- набор команд для компьютера

Команда – это описание действий, которые должен выполнить компьютер.

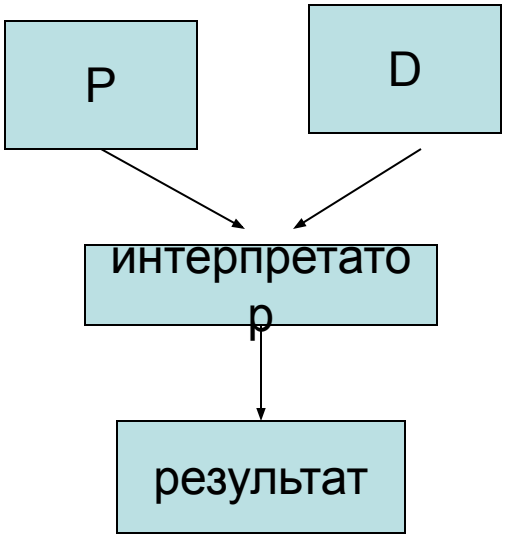
- откуда взять исходные данные?
- что нужно с ними сделать?
- куда поместить результат?

Язык программирования - формальный язык, воспринимаемый ЭВМ и предназначенный для общения человека с компьютером (входной язык для РС).

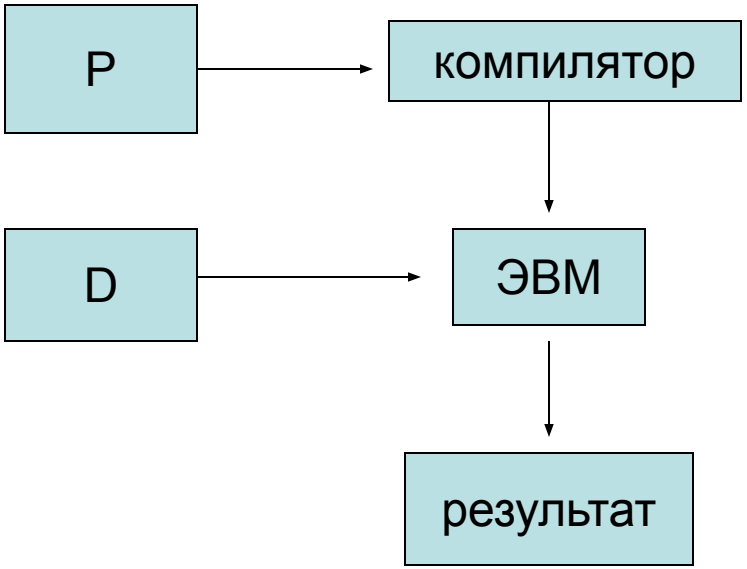
Программа – алгоритм, записанный на языке программирования.

ОПЕРАТОР – формальная запись предписания для выполнения действия или последовательности действий, заданных алгоритмом.

1.ИНТЕРПРЕТАЦИЯ



2. КОМПИЛЯЦИЯ



2.Базовые алгоритмические структуры

Логическая структура любого алгоритма может быть представлена комбинацией трех базовых (основных) структур (элементов):

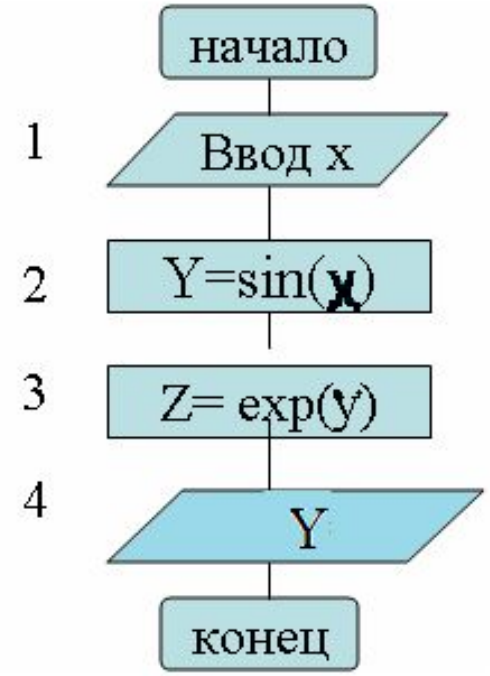
- **следование,**
- **ветвление,**
- **цикл.**

Характерной особенностью базовых структур является наличие в них **одного входа и одного выхода.**

2.1.Базовая структура "следование"

Образуется последовательностью действий, следующих одно за другим:

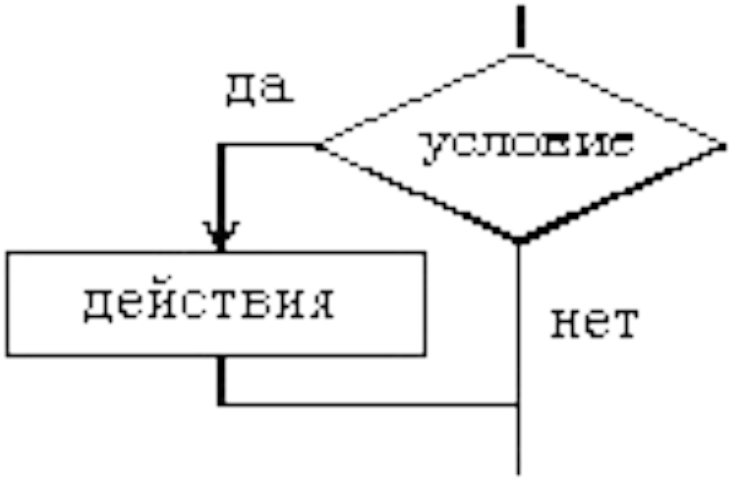
$Z=exp(y)$, $y=\sin x$

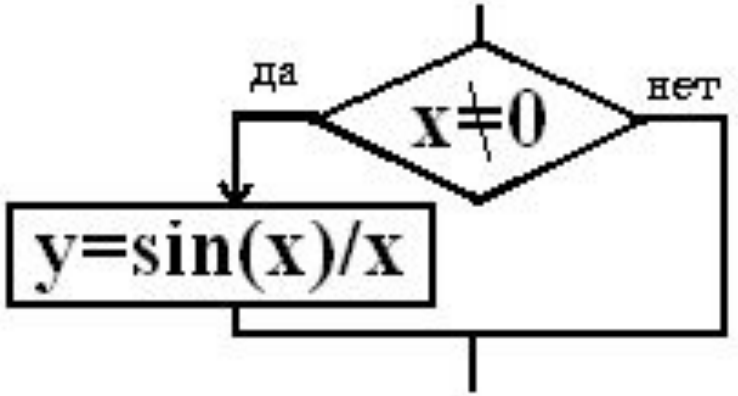


2.2. Базовая структура "ветвление".

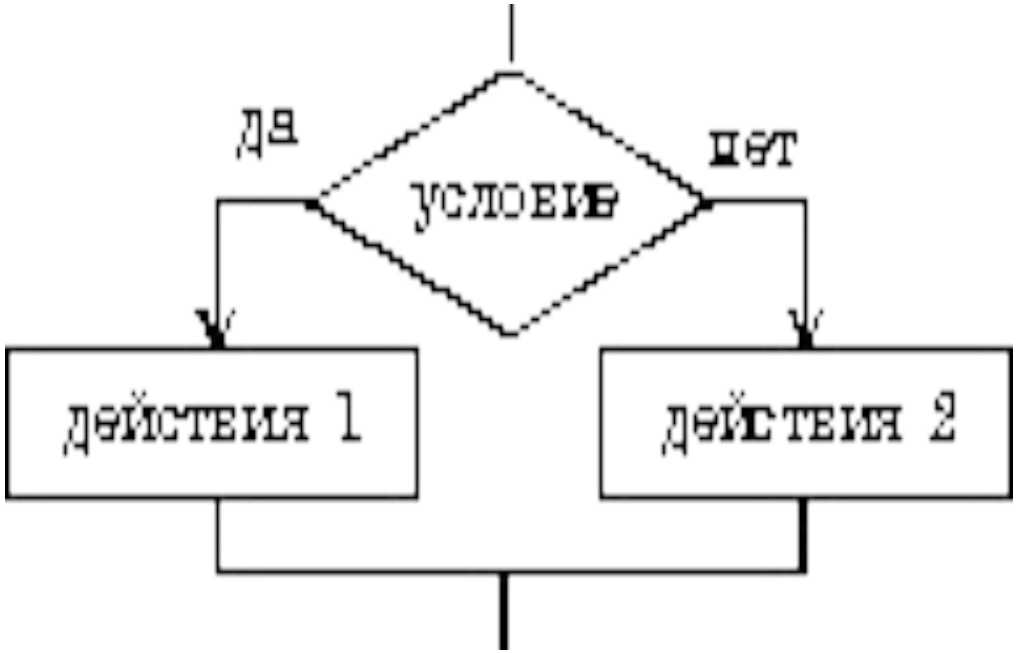
Обеспечивает в зависимости от результата проверки условия (**да** или **нет**) выбор одного из альтернативных путей работы алгоритма. Каждый из путей ведет к **общему выходу**.

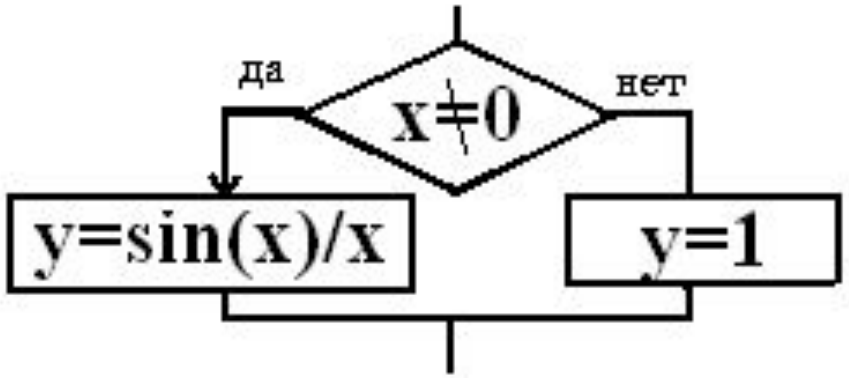
1. если—то



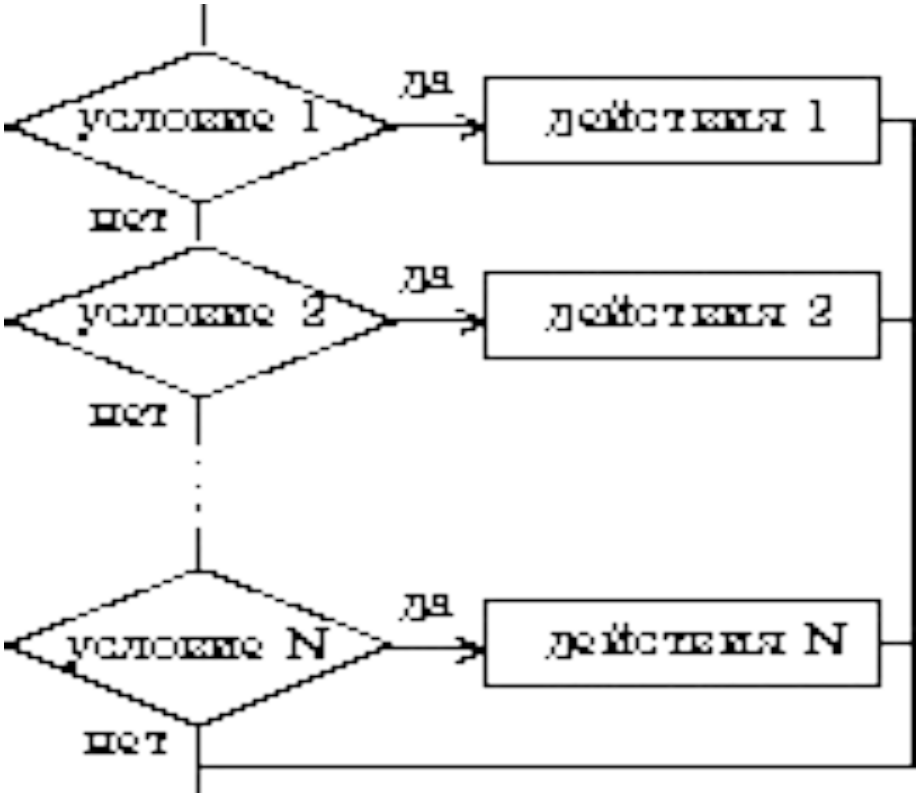


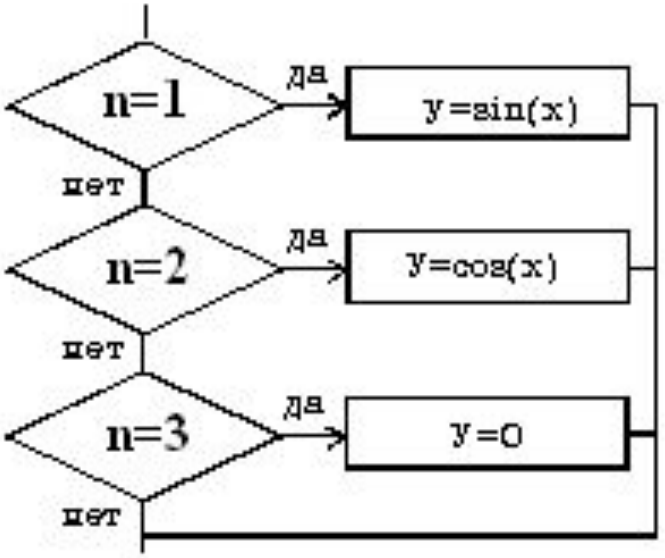
2. если—то—иначе





3. выбор


условие 1
да
действия 1
нет
условие 2
да
действия 2
нет
условие N
да
действия N
нет


n=1
да
y=sin(x)
нет
n=2
да
y=cos(x)
нет
n=3
да
y=0
нет

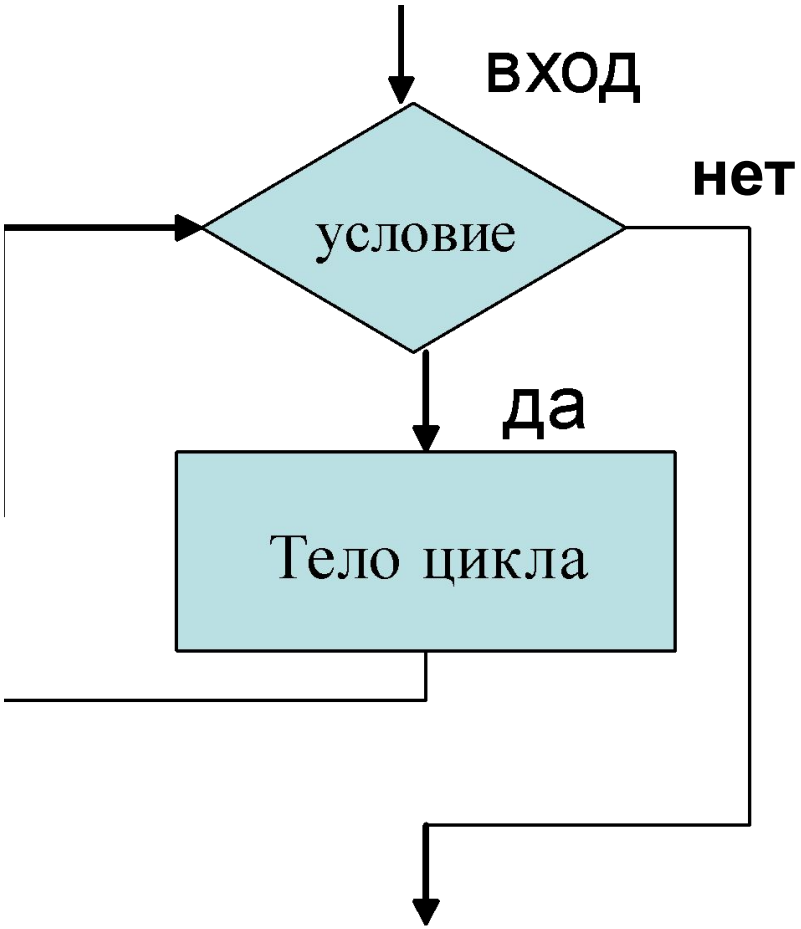
2.3. Базовая структура "цикл".

Обеспечивает **многократное выполнение некоторой совокупности действий,** которая называется **телом цикла**.

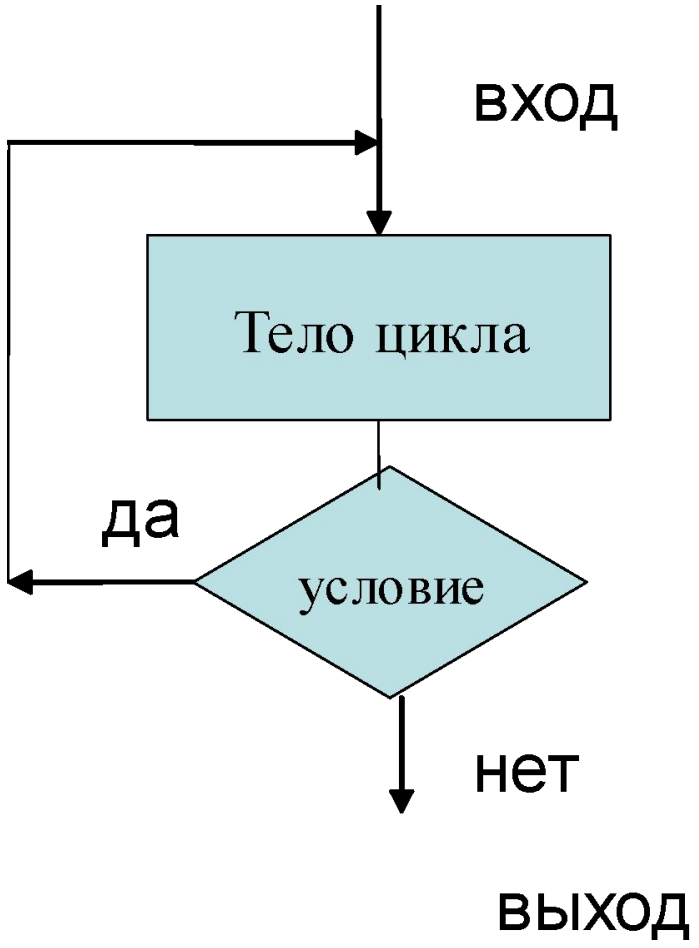
Управляющие операторы цикла:

- проверка условия выхода из цикла
- задание начального значения управляющего параметра
- изменение значения управ. параметра

1. Цикл типа пока
(с предусловием)



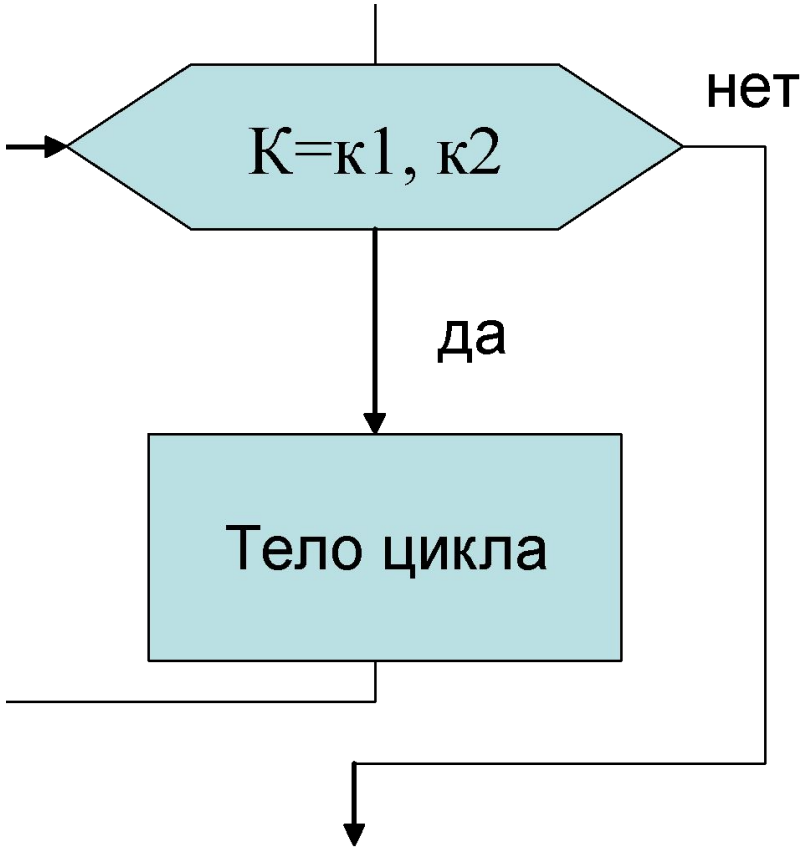
2. Цикл типа до
(с постусловием)

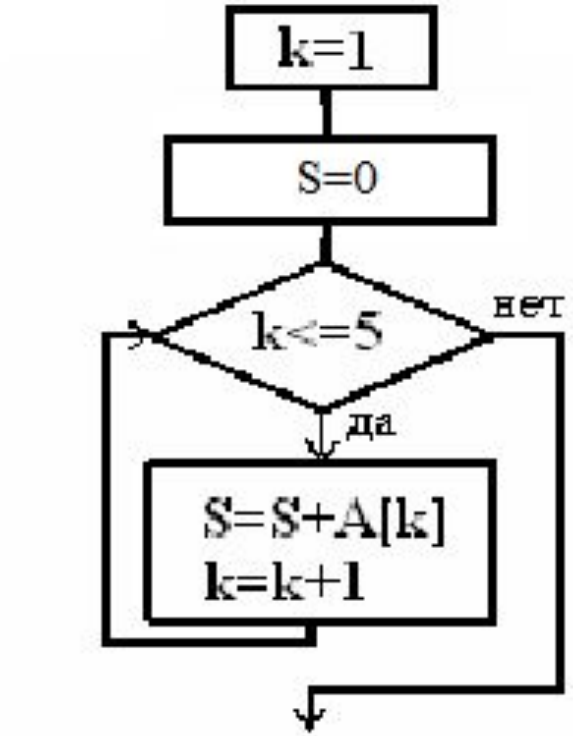


3. Модификация

Цикл с известным числом повторений

Цикл с индексом





$$S = \sum_{k=1}^{5} A_k = A_1 + A_2 + \ldots + A_5$$

Задача

вычитание дробей (a, b, c, d — натуральные числа)

$$\frac{a}{b} - \frac{c}{d} = \frac{ad - cb}{bd}, b * d \neq 0$$

Эффективный алгоритм предполагает использовать в качестве вспомогательного алгоритм нахождения наибольшего общего делителя двух натуральных чисел.

Это позволяет представить результат в виде обыкновенной несократимой дроби.