Программирование на языке C++

Составила О.Н.Кулик

По материалам Полякова
Константина Юрьевича
д.т.н., учитель информатики
ГБОУ СОШ № 163, г. Санкт-Петербур

Простейшая программа

результат – целое число

это основная программа

```
int main()
{
  // это основная программа
  /* здесь записывают
     операторы  */
}
```

комментарии после // не обрабатываются

это тоже комментарий

? Что делает эта программа?

Вывод на экран

```
int main()
{
    cout << "2+";
    cout << "2=?\n";
    cout << "Ответ: 4";
}
```

character output – **выходной поток** [символов] на консоль

"\n" – новая строка

Протокол:

```
2+
Ответ: 4
```

Подключение библиотечных функций

```
#include <iostream>
using namespace std;
int main()
{
    cout << "2+";
    cout << "2=?\n";
    cout << "Ответ: 4";
    cin.get();
}
```

стандартные потоки ввода и вывода

стандартное пространство имен

ждать нажатия любой клавиши

character input – **входной поток** [символов] с консоли

Если не подключить пространство имён…

```
#include <iostream>
int main()
{
    std::cout << "2+";
    std::cout << "2=?\n";
    std::cout << "Ответ: 4";
    std::cin.get();
}
```

пространство имен **std**

Вывод в поток

```
cout << "2+" << "2=?" << "\n"
     << "Ответ: 4";
```

```
cout << "2+" << "2=?" << endl
     << "Ответ: 4";
```

end of line – конец строки

Имена переменных

МОЖНО использовать

- латинские буквы (A-Z, a-z)

заглавные и строчные буквы **различаются**

- цифры

имя не может начинаться с цифры

- знак подчеркивания _

НЕЛЬЗЯ использовать

- ~~русские буквы~~
- ~~скобки~~
- ~~знаки +, =, !, ? и др.~~

Какие имена правильные?

AXby R&B 4Wheel Вася “PesBarbos”

TU154 [QuQu] _ABBA A+B

Объявление переменных

тип – целые

список имен переменных

```
int a, b, c;
```

выделение места в памяти

Начальные значения:

```
int a, b = 1, c = 55;
```

Типы данных

- `Int` // целое
- `long int` // длинное целое
- `float` // вещественное
- `double` // веществ. двойной точности
- `bool` // логические значения
- `Char` // символ
- `String` // символьная строка
- и др.

Тип переменной

- область допустимых значений
- допустимые операции
- объём памяти
- формат хранения данных
- для предотвращения случайных ошибок

Как записать значение в переменную?

оператор присваивания

```
a = 5;
```

! При записи нового значения старое стирается!

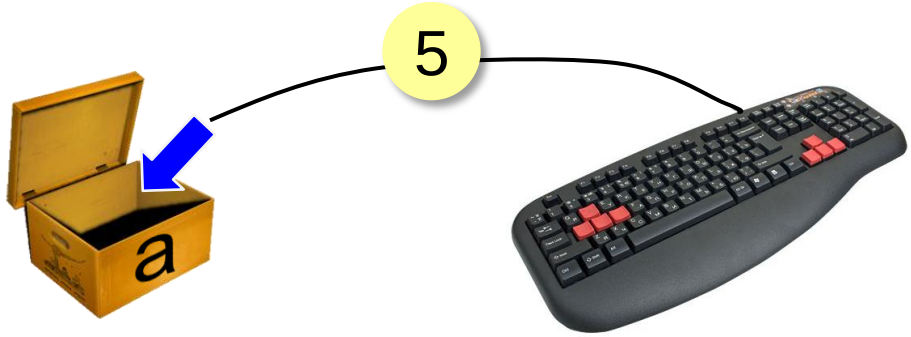
Оператор – это команда языка программирования (инструкция).

Оператор присваивания – это команда для записи нового значения в переменную.

Ввод значения с клавиатуры

ввести значение **a** из входного потока

```
cin >> a;
```

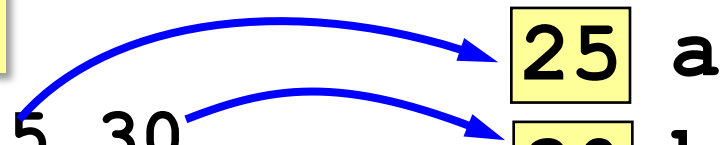


1. Программа ждет, пока пользователь введет значение и нажмет *Enter*.
2. Введенное значение записывается в переменную **a**.

Ввод значений двух переменных

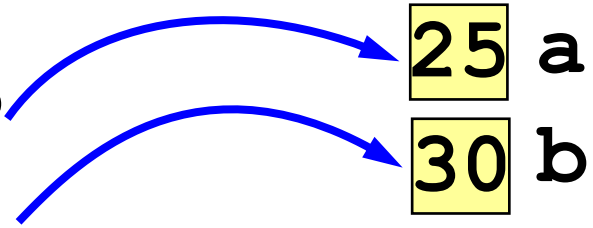
```
cin >> a >> b;
```

через пробел: 25 30



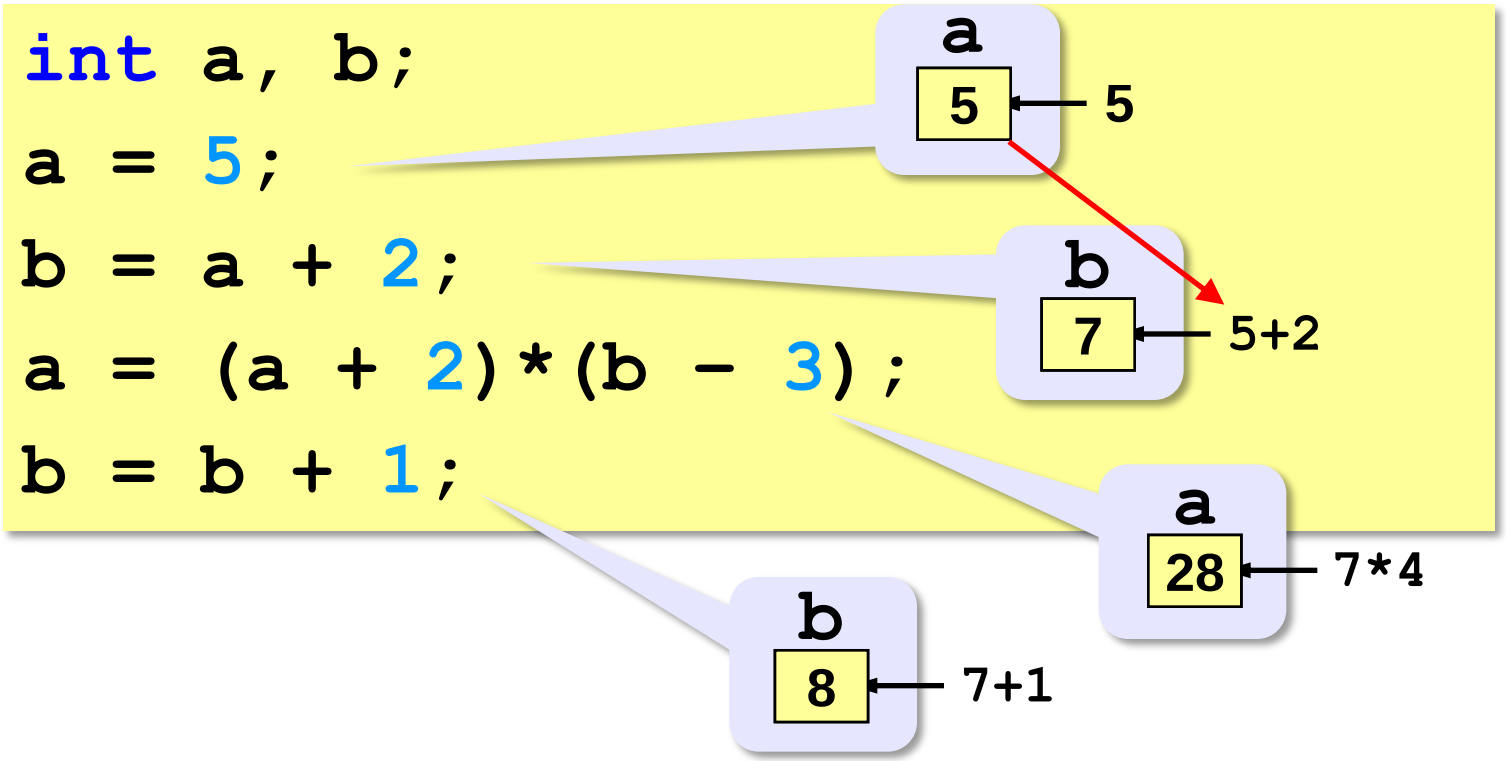
через *Enter*:

25
30



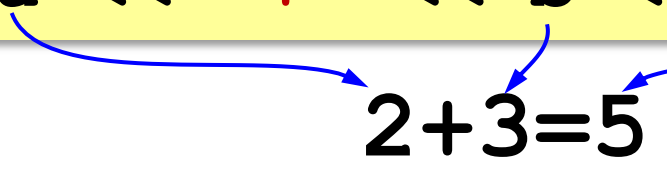
Изменение значений переменной

```
int a, b;
a = 5;
b = a + 2;
a = (a + 2)*(b - 3);
b = b + 1;
```



Вывод данных

```
cout << a;                    //вывод значения
                              //переменной a

cout << a << endl;            //...и переход
                              //на новую строку

cout << "Привет!";            //вывод текста

cout << "Ответ: " << c;

//вывод текста и значения переменной c

cout << a << "+" << b << "=" << c;
```



Сложение чисел: полное решение

```
int a, b, c;
cout << "Введите два целых числа\n";
cin >> a >> b;
c = a + b;
cout << a << "+" << b << "=" << c;
```

подсказка

Протокол:

```
Введите два целых числа
25 30
25+30=55
```

компьютер

пользователь

Снова про оператор вывода

Вычисление выражений:

```
cout << a << "+" << b << "=" << a+b;
```

Форматный вывод:

```
#include <iomanip>
...
a = 123;
cout << setw(5) << a;
```

манипуляторы для управления потоками

␣␣123

5 знаков

set width – установить ширину поля

Арифметическое выражения

```
          3   1 2   4       5    6
a = (c + b*5*3 - 1) / 2 * d;
```

Приоритет (*старшинство*):

1) скобки
2) умножение и деление
3) сложение и вычитание

$$a = \frac{c + b \cdot 5 \cdot 3 - 1}{2} \cdot d$$

Деление

Результат деления целого на целое – **целое** число (остаток отбрасывается):

```
int a = 3, b = 4;
float x;
x = 3 / 4;
x = 3. / 4;
x = 3 / 4.;
x = a / 4;
x = a / 4.;
x = a / b;
x = float(a) / 4;
x = a / float(b);
```

Остаток от деления

% – остаток от деления

```
int a, b, d;
d = 85;
b = d / 10;
a = d % 10;
d = a % b;
d = b % a;
```

Для отрицательных чисел:

```
int a = -7;
b = a / 2;
d = a % 2;
```

В математике не так!

остаток ≥ 0

```
-7 = (-4)*2 + 1
```

Сокращенная запись операций

```
int a, b;

...
a ++;      // a = a + 1;
a --;      // a = a - 1;
a += b;    // a = a + b;
a -= b;    // a = a - b;
a *= b;    // a = a * b;
a /= b;    // a = a / b;
a %= b;    // a = a % b;
```

Стандартные функции

```
#include <cmath>
```

подключить математическую библиотеку

abs(x) – модуль целого числа
fabs(x) – модуль вещественного числа
sqrt(x) – квадратный корень
sin(x) – синус угла, заданного **в радианах**
cos(x) – косинус угла, заданного **в радианах**
exp(x) – экспонента e^x
ln(x) – натуральный логарифм
pow(x,y) – x^y: возведение числа x в степень y
floor(x) – округление «вниз»
ceil(x) – округление «вверх»

```
float x;
x = floor(1.6);// 1
x = ceil(1.6); // 2
```

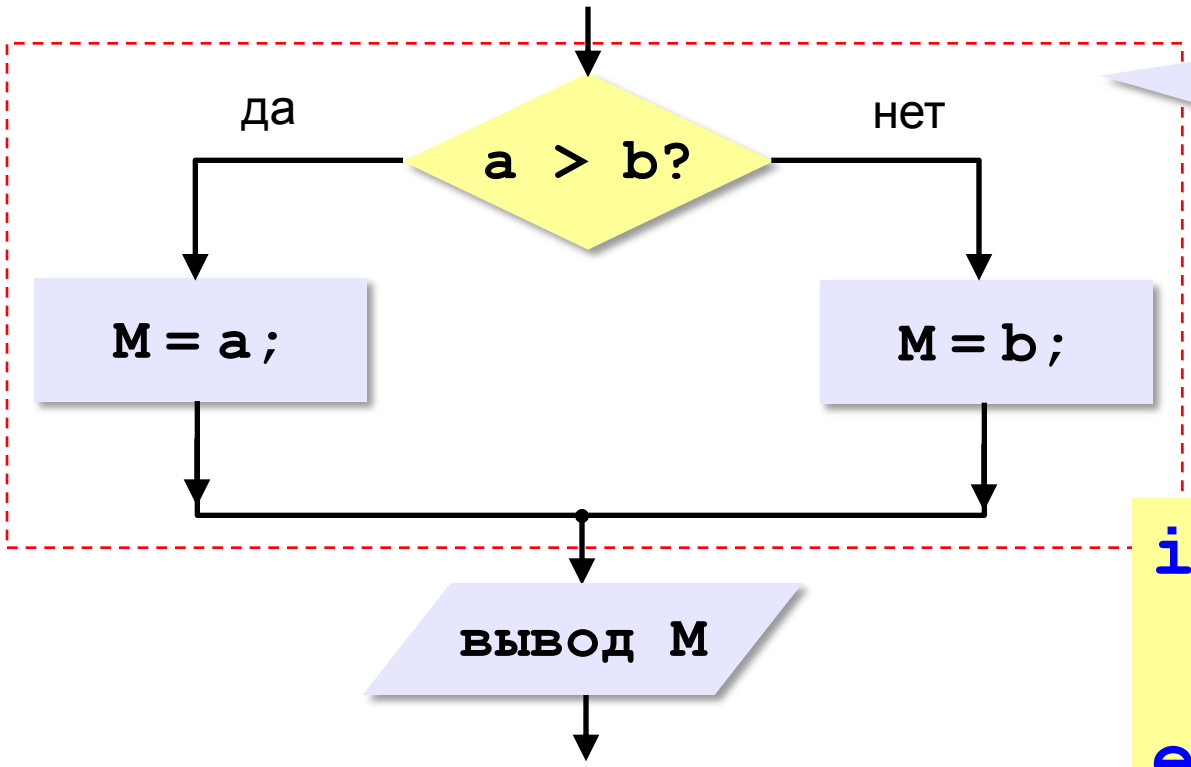
```
x = floor(-1.6);//-2
x = ceil(-1.6); //-1
```

Программирование на языке C++

Ветвления

Условный оператор

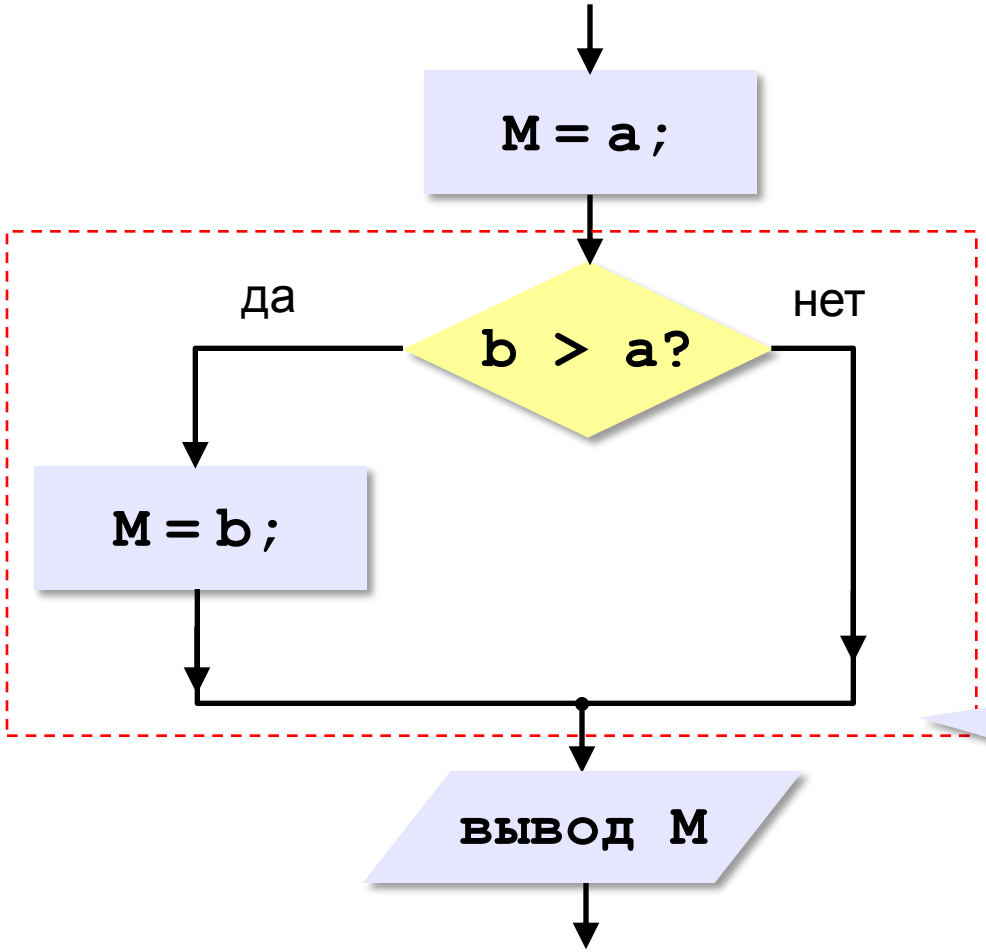
Задача: **изменить порядок действий** в зависимости от выполнения некоторого условия.



полная форма ветвления

```
if ( a > b )
   M = a;
else
   M = b;
```

Условный оператор: неполная форма




```
M = a;
if ( b > a )
   M = b;
```

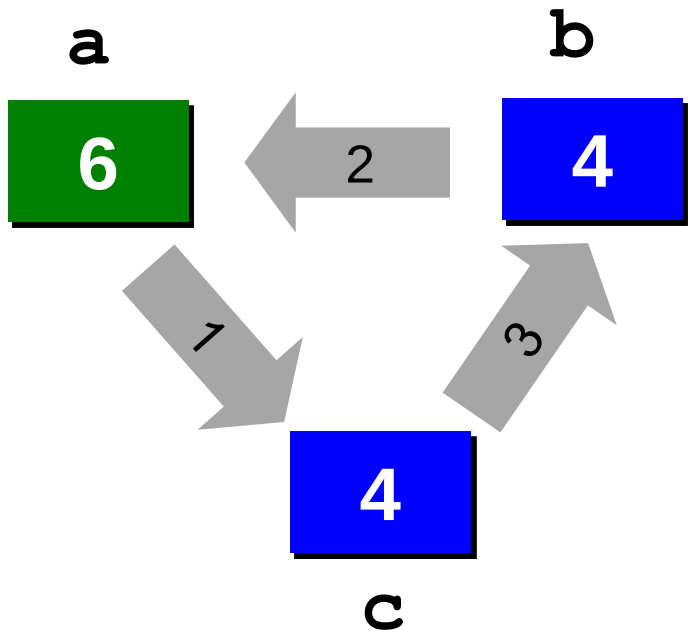
неполная форма ветвления

Условный оператор

```
if ( a < b )
  {
  c = a;
  a = b;
  b = c;
  }
```

блок (составной оператор)

? Что делает?



Знаки отношений

> <	больше, меньше
>=	больше или равно
<=	меньше или равно
==	равно
!=	не равно

Вложенные условные операторы

Задача: в переменных **a** и **b** записаны возрасты Андрея и Бориса. Кто из них старше?

```
if ( a > b )
   cout << "Андрей старше";
else
   if ( a == b )
      cout << "Одного возраста";
   else
      cout << "Борис старше";
```

вложенный условный оператор

Сложные условия

Задача: набор сотрудников в возрасте **25-40 лет** (включительно).

сложное условие

```
if ( v >= 25 && v <= 40 )
   cout << "подходит";
else
   cout << "не подходит";
```

`&&`	«И»
`||`	«ИЛИ»
`!`	«НЕ»

Приоритет :

1) отношения (`<`, `>`, `<=`, `>=`, `==`, `!=`)
2) `!` («НЕ»)
3) `&&` («И»)
4) `||` («ИЛИ»)

Множественный выбор

```
if (m == 1) cout << "январь";
if (m == 2) cout << "февраль";
...
if (m == 12) cout << "декабрь";
```

```
switch ( m ) {
   case 1: cout << "январь";
           break;
   case 2: cout << "февраль";
           break;
   ...
   case 12: cout << "декабрь";
            break;
   default: cout << "ошибка";
 }
```

Множественный выбор

Если не ставить **break**:

```
switch ( m ) {
   case 1:    cout << "январь";
   case 2:    cout << "февраль";
   case 3:    cout << "март";
   default:   cout << "ошибка";
 }
```

При **m = 2**: **февральмартошибка**

Множественный выбор

```
char c;
c = getch();
switch(c)
  {
  case 'а':
       cout << "антилопа\n";
       cout << "Анапа\n";
       break;
  ...
  case 'я':
       cout << "ягуар\n";
       cout << "Якутск\n";
       break;
  default: cout << "Ошибка!";
  }
```

ждать нажатия клавиши, получить её код

несколько операторов в блоке

Программирование на языке C++

Циклические алгоритмы

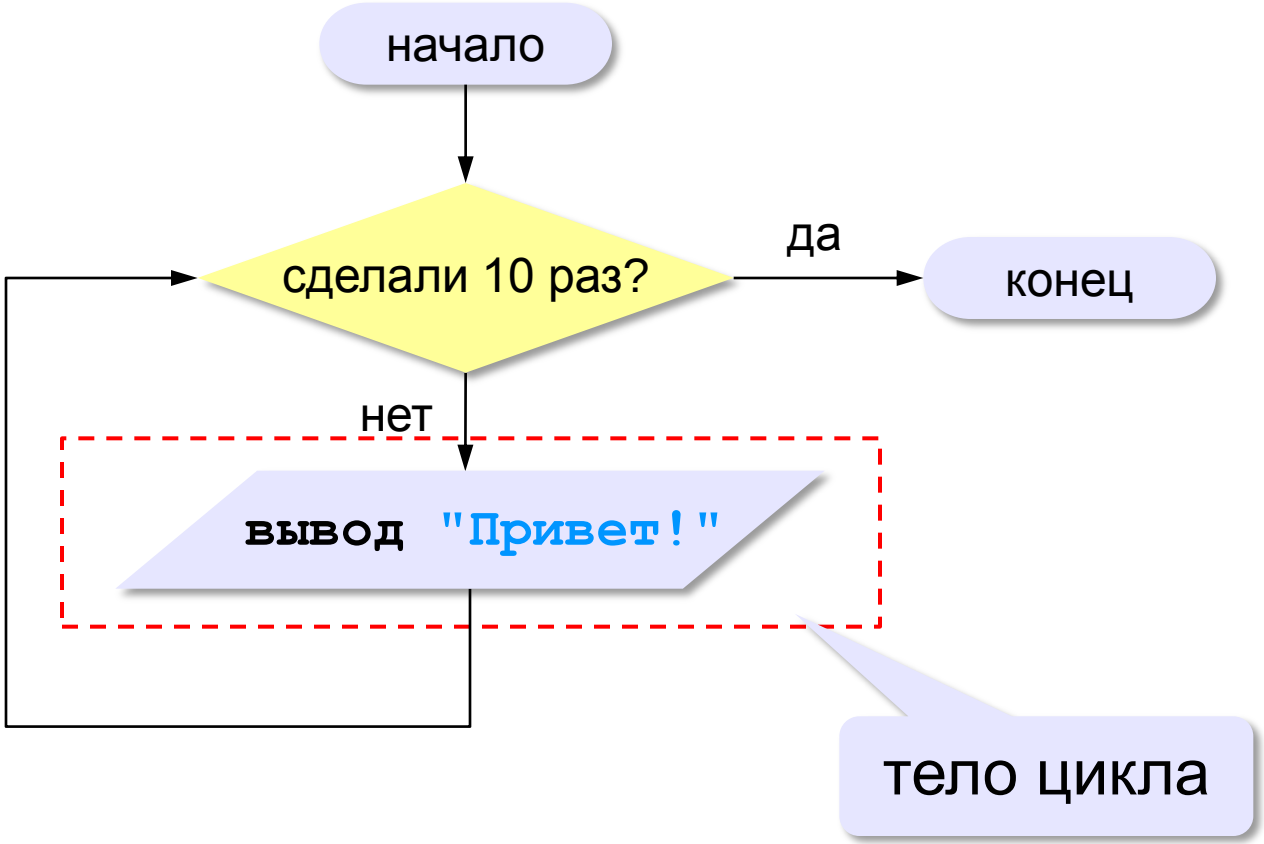
Что такое цикл?

Цикл – это многократное выполнение одинаковых действий.

Два вида циклов:

- цикл с **известным** числом шагов (сделать 10 раз)
- цикл с **неизвестным** числом шагов (делать, пока не надоест)

Блок-схема цикла



Как организовать цикл?

```
счётчик = 0
пока счётчик < 10
   cout << "Привет\n";
   увеличить счётчик на 1
```

```
счётчик = 10
пока счётчик > 0
   cout << "Привет\n";
   уменьшить счётчик на 1
```

результат операции автоматически сравнивается с нулём!

Цикл с условием

Задача. Определить **количество цифр** в десятичной записи целого положительного числа, записанного в переменную **n**.

```
счётчик = 0
пока n > 0
   отсечь последнюю цифру n
   увеличить счётчик на 1
```

n	счётчик
1234	0

? Как отсечь последнюю цифру?

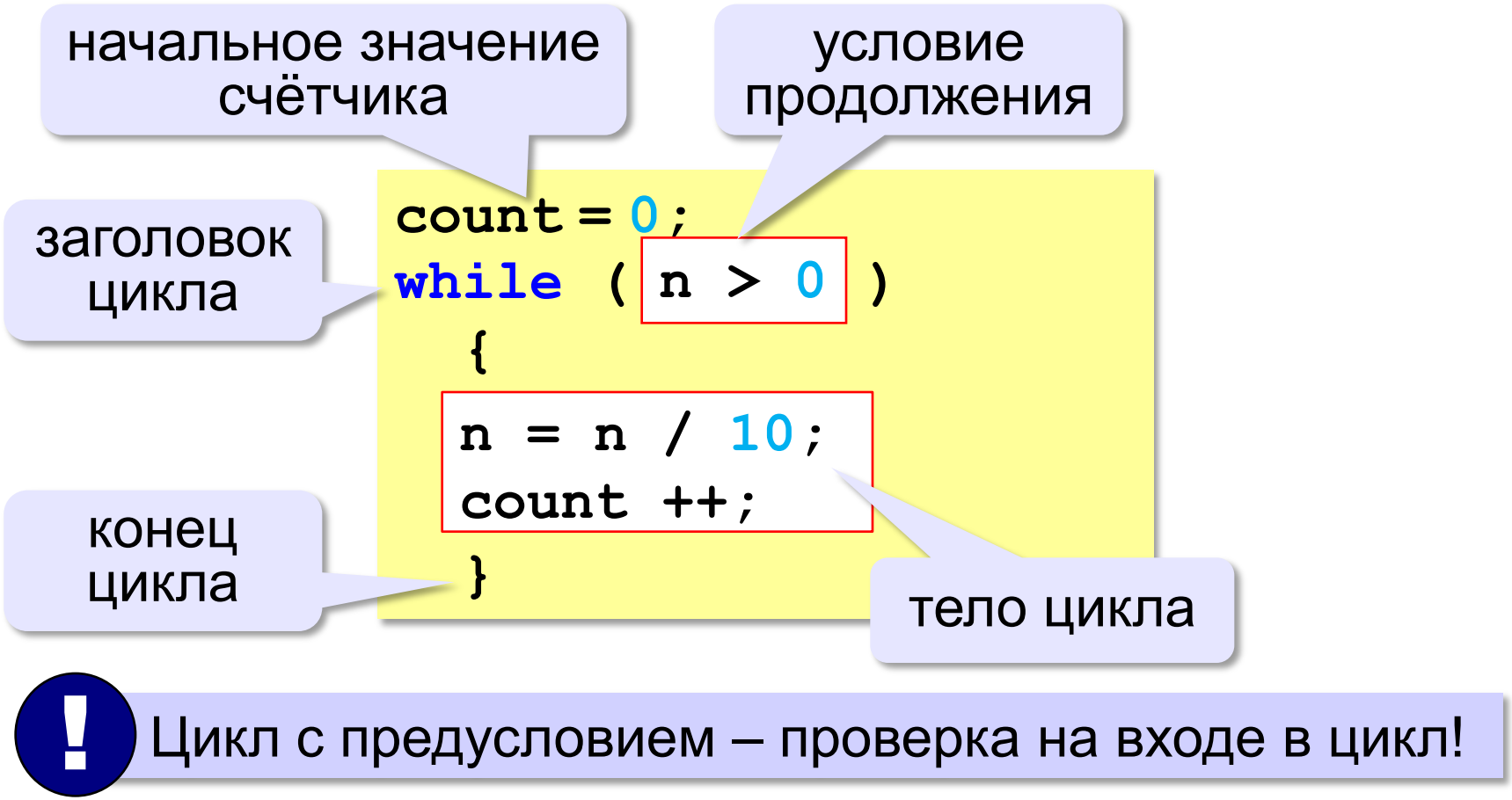
```
n = n / 10;
```

? Как увеличить счётчик на 1?

```
счётчик = счётчик + 1;
```

```
счётчик ++;
```

Цикл с условием



! Цикл с предусловием – проверка на входе в цикл!

Цикл с условием

При известном количестве шагов:

```
k = 0;
while ( k < 10 )
   {
   cout <<  "привет\n";
   k ++;
   }
```

Зацикливание:

```
k = 0;
while ( k < 10 )
   {
   cout <<  "привет\n";
   }
```

Сколько раз выполняется цикл?

```
a = 4; b = 6;
while ( a < b ) a = a + 1;
```

2 раза
a = 6

```
a = 4; b = 6;
while ( a < b ) a = a + b;
```

1 раз
a = 10

```
a = 4; b = 6;
while ( a > b ) a ++;
```

0 раз
a = 4

```
a = 4; b = 6;
while ( a < b ) b = a - b;
```

1 раз
b = -2

```
a = 4; b = 6;
while ( a < b ) a --;
```

зацикливание

Цикл с постусловием

заголовок цикла

тело цикла

```
do
   {
   cout << "Введите n > 0: ";
   cin >> n;
   }
while ( n <= 0 );
```

условие продолжения

- при входе в цикл условие ***не*** **проверяется**
- цикл всегда выполняется **хотя бы один раз**

Программирование на языке C++

Циклы по переменной

Цикл по переменной

Задача. Вывести все степени двойки от 2^1 до 2^{10}.

? Можно ли сделать с циклом «пока»?

```
k = 1;
n = 2;
while ( k <= 10 )
   {
   cout << n << endl;
   n *= 2;
   k ++;
   }
```

```
n = 2;
for( k=1; k<=10; k++ )
   {
   cout << n << endl;
   n *= 2;
   }
```

цикл с переменной

Цикл по переменной: другой шаг

это внутренняя переменная цикла

```
for ( int k = 10; k >= 1; k-- )
  cout << k*k << endl;
```

100
81
64
49
36
25
16
9
4
1

Что получится?

```
for ( int k = 1; k <= 10; k += 2 )
  cout << k*k << endl;
```

1
9
25
49
81

Сколько раз выполняется цикл?

```
int a=1;
for( int i=1; i<=3; i++ ) a=a+1;
```

a = 4

```
int a=1;
for( int i=3; i<=1; i++ ) a=a+1;
```

a = 1

```
int a=1;
for( int i=1; i<=3; i-- ) a=a+1;
```

зацикливание

```
int a=1;
for( int i=3; i>=1; i-- ) a=a+1;
```

a = 4

Вложенные циклы

Задача. Вывести все простые числа в диапазоне от 2 до 1000.

```
сделать для n от 2 до 1000
  если число n простое то
    вывод n
```

нет делителей [2.. **n-1**]: проверка в цикле!

Вложенные циклы

```
for ( int n = 2; n <= 1000; n ++ )
  {
  count = 0;
  for ( int k = 2; k < n; k ++ )
      if ( n % k == 0 )
        count ++;
  if ( count == 0 )
    cout << n << endl;
  }
```

вложенный цикл

Вложенные циклы

```
for ( int i = 1; i <= 4; i++ )
   {
   for ( int k = 1; k <= i; k++ )
      {
      cout << i << " " << k
           << endl;
      }
   }
```

```
1 1
2 1
2 2
3 1
3 2
3 3
4 1
4 2
4 3
4 4
```

? Как меняются переменные?

! Переменная внутреннего цикла изменяется быстрее!

Программирование на языке C++

ПОНЯТИЯ разные

Что такое процедура?

Процедура – вспомогательный алгоритм, который выполняет некоторые действия.

- в момент вызова процедура должна уже быть известна (например, расположена до основной программы)
- в программе может быть **много процедур**
- чтобы процедура заработала, нужно **вызвать** её по имени из основной программы или из другой процедуры

Что такое функция?

Функция – это вспомогательный алгоритм, который возвращает *значение-результат* (число, символ или объект другого типа).

Примеры:

```
float s = sin(x);
char c = getch();
int v = rand() % 10;
```

Что такое рекурсия?

Натуральные числа:

- 1 – натуральное число
- если n – натуральное число, то $n+1$ – натуральное число

индуктивное определение

Рекурсия — это способ определения множества объектов через само это множество на основе заданных простых базовых случаев.

Числа Фибоначчи:

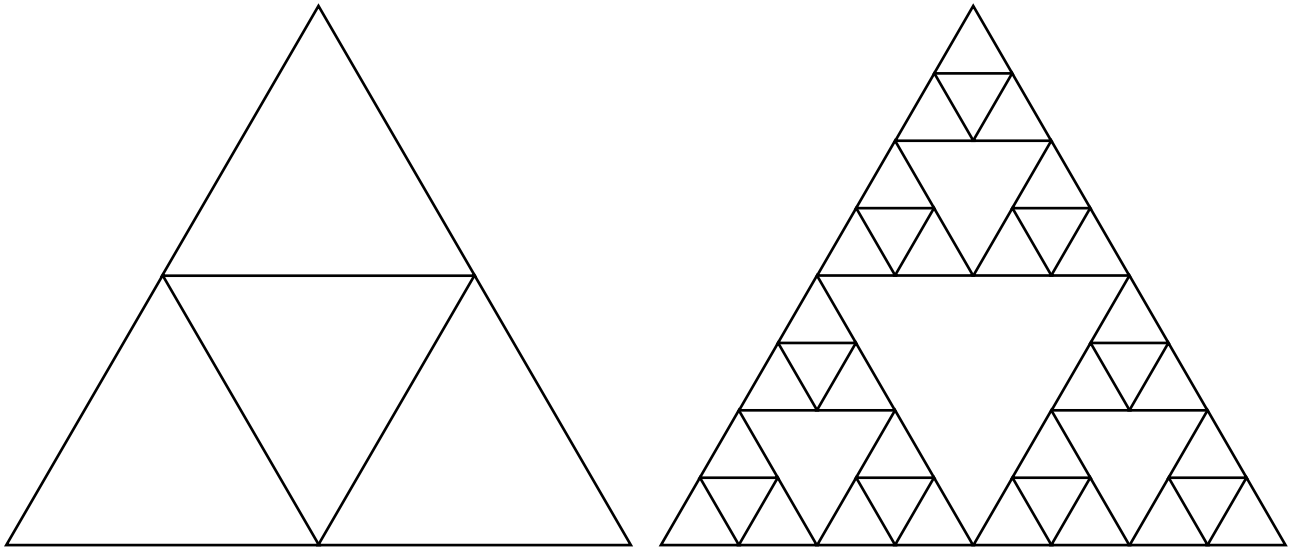
- $F_1 = F_2 = 1$
- $F_n = F_{n-1} + F_{n-2}$ при $n > 2$

1, 1, 2, 3, 5, 8, 13, 21, 34, …

Фракталы

Фракталы – геометрические фигуры, обладающие самоподобием.

Треугольник Серпинского:



Ханойские башни – процедура

Рекурсивная процедура (функция) — это процедура (функция), которая вызывает сама себя напрямую или через другие процедуры и функции.

```
void Hanoi ( int n, int k, int m )
{
  int p;
  if ( n == 0 ) return;
  p = 6 - k - m;
  Hanoi ( n - 1, k, p );
  cout << k << " -> " << m << endl;
  Hanoi ( n - 1, p, m );
}
```

условие выхода из рекурсии

```
int main()
{
  Hanoi(4, 1, 3);
}
```

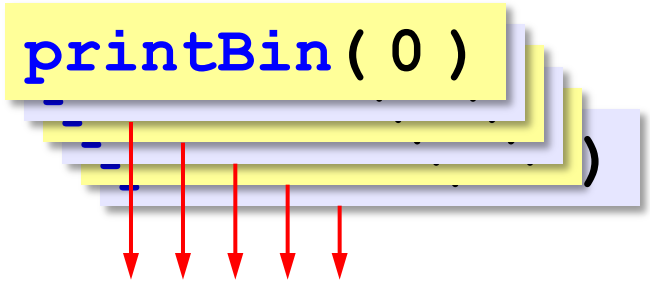
Вывод двоичного кода числа

```
void printBin( int n )
{
  if (n == 0) return;
  printBin(n / 2);
  cout << n % 2;
}
```

условие выхода из рекурсии

напечатать все цифры, кроме последней

вывести последнюю цифру



Алгоритм Евклида

Алгоритм Евклида. Чтобы найти НОД двух натуральных чисел, нужно вычитать из большего числа меньшее до тех пор, пока меньшее не станет равно нулю. Тогда второе число и есть НОД исходных чисел.

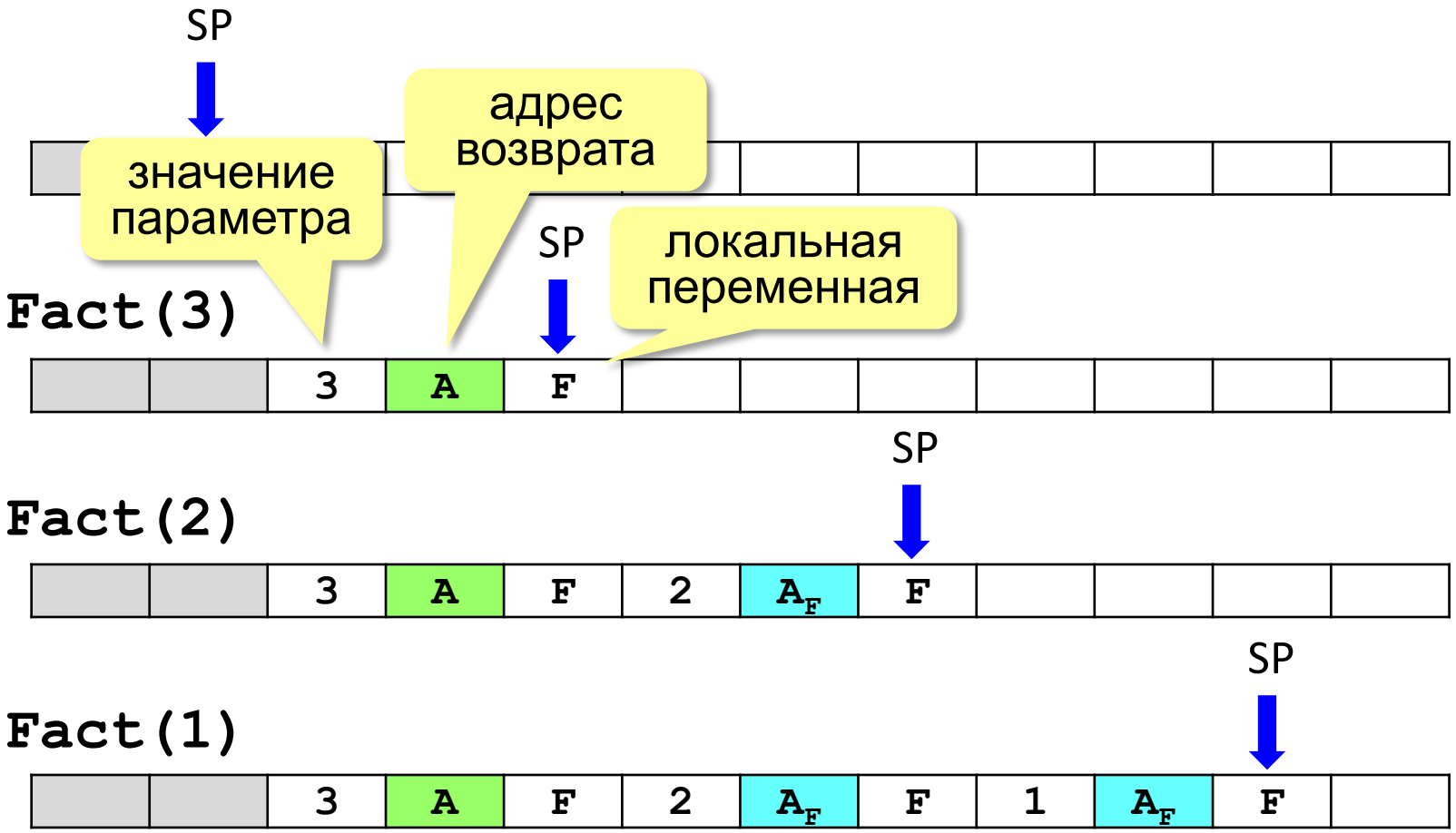
```
int NOD ( int a, int b )
{
  if ( a == 0 || b == 0 )
     return a+b;
  if ( a>b )
        return NOD( a-b, b );
  else return NOD( a, b-a );
}
```

условие окончания рекурсии

рекурсивные вызовы

Стек

Стек – область памяти, в которой хранятся локальные переменные и адреса возврата.



Источники иллюстраций

1. old-moneta.ru
2. www.random.org
3. www.allruletka.ru
4. www.lotterypros.com
5. logos.cs.uic.edu
6. ru.wikipedia.org
7. иллюстрации художников издательства «Бином»
8. авторские материалы