

Алгоритм и его свойства

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ C++



Что такое алгоритм?

Алгоритм — это точное описание порядка действий, которые должен выполнить исполнитель для решения задачи за конечное время.

Исполнитель — это устройство или одушевленное существо (человек), способное понять и выполнить команды, составляющие алгоритм.

Формальные исполнители: не понимают (и не могут понять) смысл команд.



Мухаммед ал-Хорезми
(ок. 783—ок. 850 гг.)

Свойства алгоритма

Дискретность — алгоритм состоит из отдельных команд, каждая из которых выполняется за конечное время.

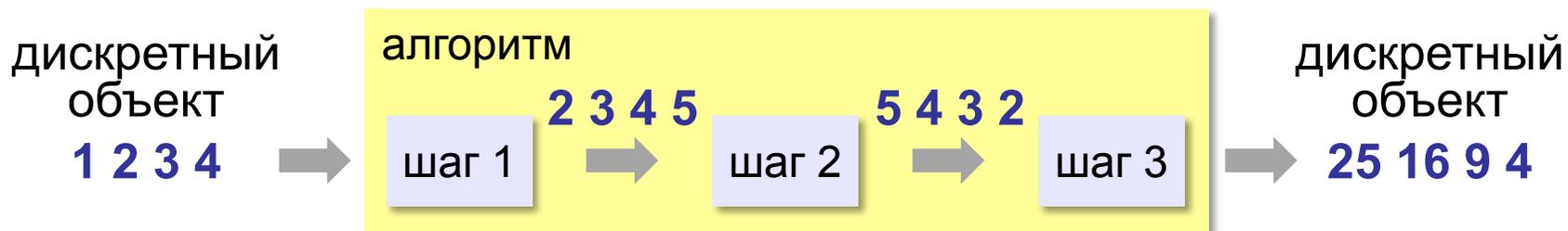
Детерминированность (определённость) — при каждом запуске алгоритма с одними и теми же исходными данными получается один и тот же результат.

Понятность — алгоритм содержит только команды, входящие в систему команд исполнителя.

Конечность (результативность) — для корректного набора данных алгоритм должен завершаться через конечное время.

Корректность — для допустимых исходных данных алгоритм должен приводить к правильному результату.

Как работает алгоритм?



- получает на вход дискретный объект
- в результате строит другой дискретный объект (или выдаёт сообщение об ошибке)
- обрабатывает объект по шагам
- на каждом шаге получается новый дискретный объект

Способы записи алгоритмов

- **естественный язык**

**установить соединение
пока не принята команда «стоп»
принять команду
выполнить команду
завершить сеанс связи**

Способы записи алгоритмов

- псевдокод

установить соединение

начало цикла

 принять команду

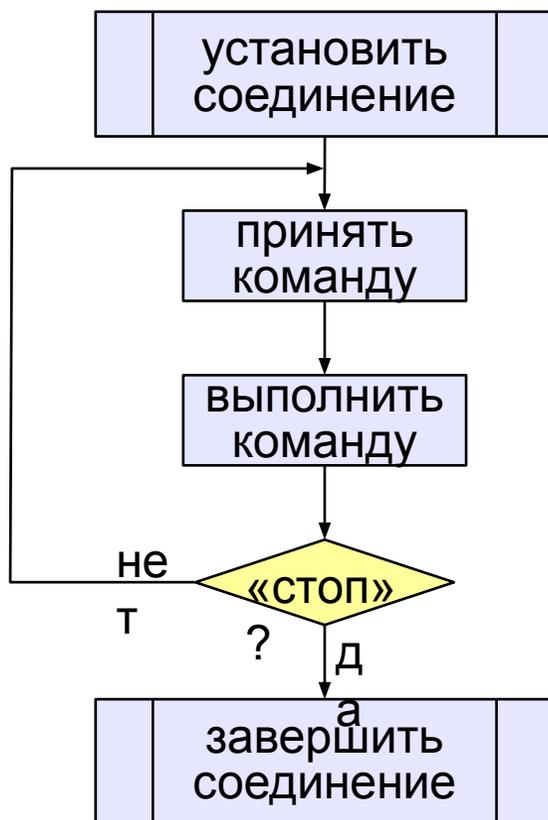
 выполнить команду

конец цикла при команда = 'stop'

завершить сеанс связи

Способы записи алгоритмов

- блок-схема



- программа

```
установитьСоединение  
начало цикла  
cmd = получитьКоманду  
выполнитьКоманду (cmd)  
конец при cmd = 'stop'  
закрытьСоединение
```

Простейшие программы

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ C++

Простейшая программа

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

```
main()  
{  
    printf("2+");  
    printf("2=?\n");  
    printf("ответ: 4");  
  
    return 0;  
}
```

"\n" – новая строка

Подключение библиотечных функций

```
#include <stdio.h>
```

```
main()
```

```
{
```

```
    printf("2+");
```

```
    printf("2=?\n");
```

```
    printf("Ответ: 4");
```

```
    return 0;
```

```
}
```

стандартные функции
ввода и вывода

Задания

«В»: Вывести на экран текст «лесенкой»

Вася

пошел

гулять

«С»: Вывести на экран рисунок из букв

```
Ж
ЖЖЖ
ЖЖЖЖЖ
ЖЖЖЖЖЖЖ
НН НН
ZZZZZ
```

Сложение чисел

Задача. Ввести с клавиатуры два числа и найти их сумму.

Протокол:

Введите два целых числа

компьютер

25 30

пользователь

25+30=55

компьютер считает сам!

?

1. Как ввести числа в память?
2. Где хранить введенные числа?
3. Как вычислить?
4. Как вывести результат?

Сумма: псевдокод

```
main()  
{  
    // ввести два числа  
    // вычислить их сумму  
    // вывести сумму на экран  
}
```

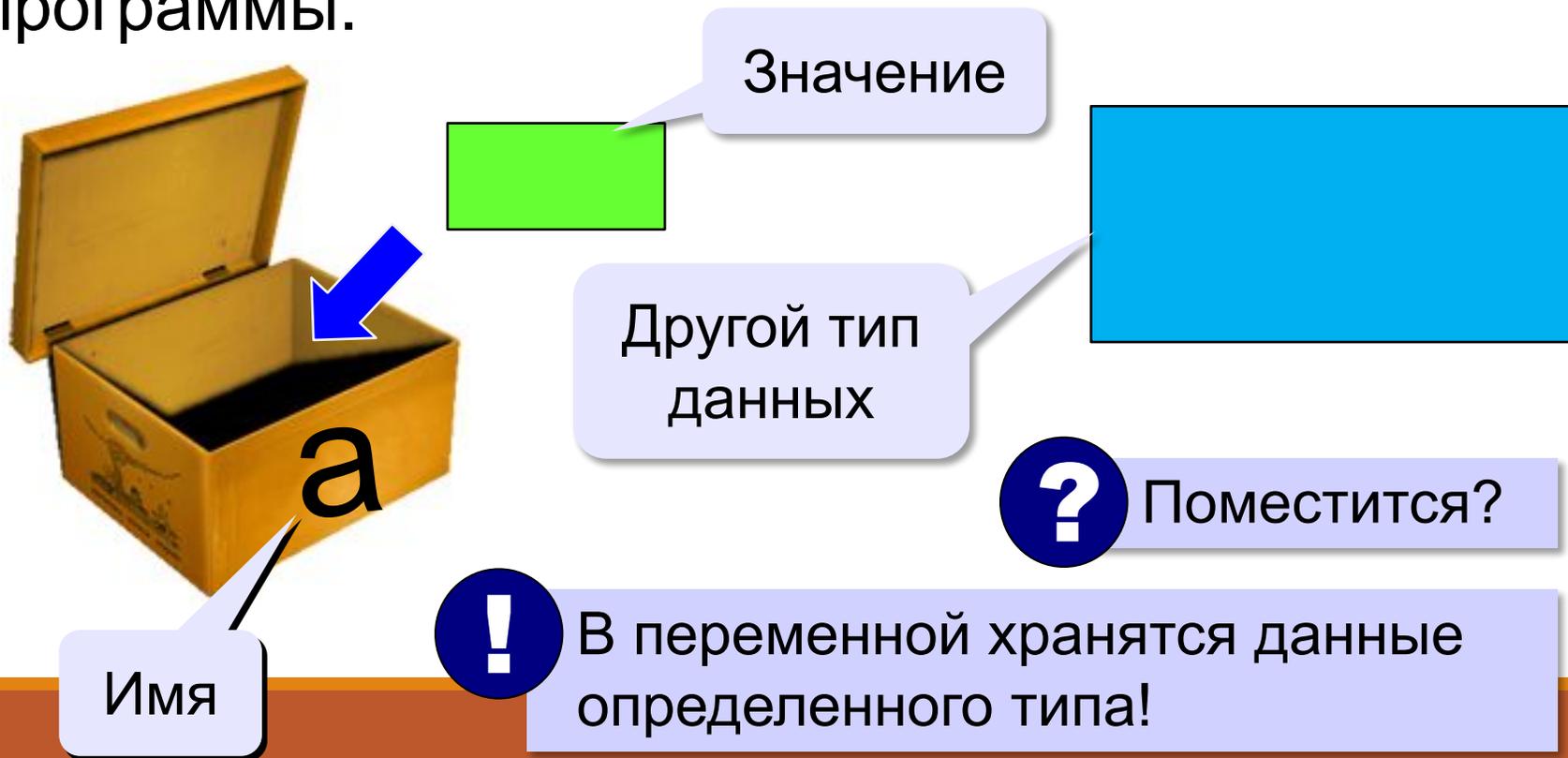
Псевдокод – алгоритм на русском языке с элементами языка программирования.



Компьютер не может исполнить псевдокод!

Переменные

Переменная – это величина, имеющая имя, тип и значение.
Значение переменной можно изменять во время работы программы.



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

Типы переменных:

- `int` // целая
- `float` // вещественная
- и другие...

Объявление переменных:

выделение
места в памяти

тип – целые

СПИСОК ИМЕН
переменных

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

Тип	байт	Диапазон принимаемых значений
целочисленный (логический) тип данных		
bool	1	0 / 255
целочисленный (символьный) тип данных		
char	1	0 / 255
целочисленные типы данных		
short int	2	-32 768 / 32 767
unsigned short int	2	0 / 65 535
int	4	-2 147 483 648 / 2 147 483 647
unsigned int	4	0 / 4 294 967 295
long int	4	-2 147 483 648 / 2 147 483 647
unsigned long int	4	0 / 4 294 967 295
типы данных с плавающей точкой		
float	4	-2 147 483 648.0 / 2 147 483 647.0
long float	8	-9 223 372 036 854 775 808 .0 / 9 223 372 036 854 775 807.0
double	8	-9 223 372 036 854 775 808 .0 / 9 223 372 036 854 775 807.0

Как записать значение в переменную?

оператор присваивания

```
a = 5;
```

5

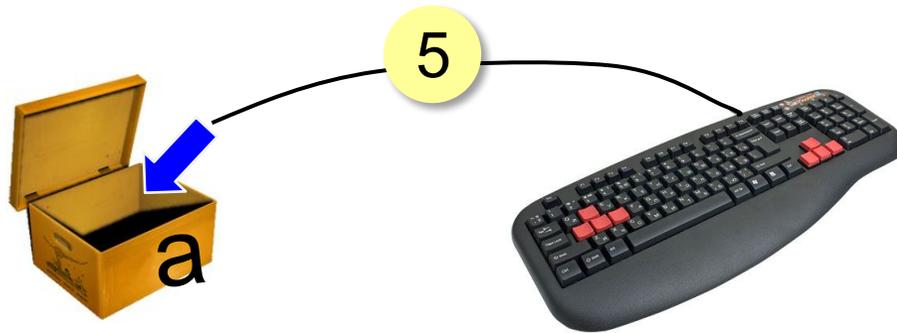
!

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



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

Ввод значения с клавиатуры

```
#include <stdio.h>

main()
{
    int a;
    scanf("%d", &a);

    return 0;
}
```

функция ввода

формат ввода

адрес переменной a

%d – целое
%f – вещественное

ввести целое число и записать в память по адресу переменной a

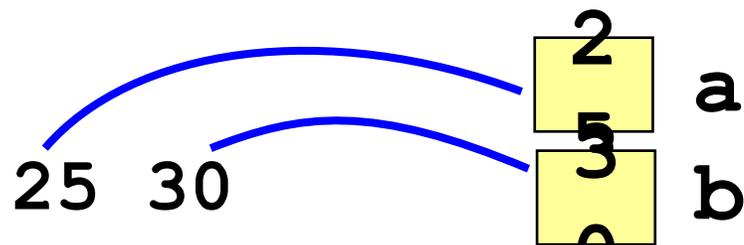
Таблица: Команды форматирования для printf()

Код	Формат
%c	Символ типа char
%d	Десятичное число целого типа со знаком
%i	Десятичное число целого типа со знаком
%e	Научная нотация (e нижнего регистра)
%E	Научная нотация (E верхнего регистра)
%f	Десятичное число с плавающей точкой
%g	Использует код %e или %f — тот из них, который короче (при использовании %g используется e нижнего регистра)
%G	Использует код %E или %f — тот из них, который короче (при использовании %G используется E верхнего регистра)
%o	Восьмеричное целое число без знака
%s	Строка символов
%u	Десятичное число целого типа без знака
%x	Шестнадцатиричное целое число без знака (буквы нижнего регистра)
%X	Шестнадцатиричное целое число без знака (буквы верхнего регистра)
%p	Выводит на экран значение указателя
%n	Ассоциированный аргумент — это указатель на переменную целого типа, в которую помещено количество символов, записанных на данный момент
%%	Выводит символ %

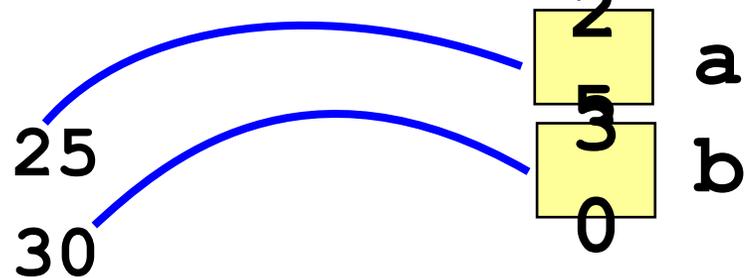
Ввод значений двух переменных

```
int a;  
int b;  
scanf("%d %d", &a, &b);
```

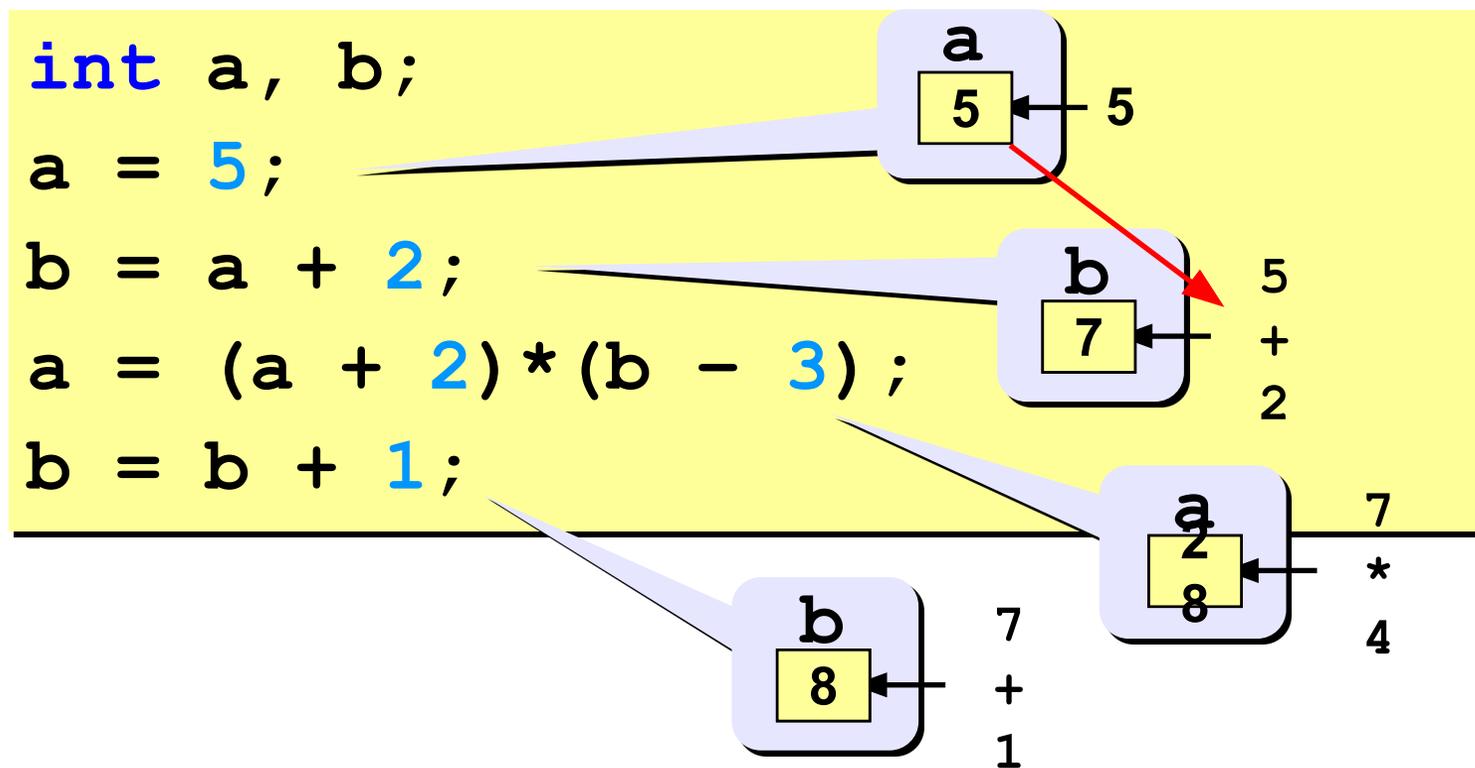
через пробел:



через *Enter*:



Изменение значений переменной



Вывод данных

формат вывода

```
printf("%d", a);  
printf("%d\n", a);
```

"\n" – новая строка

Вывод данных

```
printf("Привет!");  
printf("Ответ: %d", a);  
printf("%d+%d=%d", a, b, c);
```

Сложение чисел: простое решение

```
#include <stdio.h>

main()
{
    int a, b, c;
    scanf("%d%d", &a, &b);

    c = a + b;
    printf("%d", c);

    return 0;
}
```

Снова про оператор вывода

Вычисление выражений:

```
printf( "%d+%d=%d", a, b, a+b );
```

Форматный вывод:

```
a = 123;  
printf( "%5d", a );
```



Вычисления

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ C++

Арифметическое выражения

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

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

```
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);
```



Что запишется в **x**?

Остаток от деления

`%` – остаток от деления

```
int a, b, d;
```

```
d = 85;
```

```
b = d / 10;
```

```
a = d % 10;
```

```
d = a % b;
```

```
d = b % a;
```

Остаток от деления

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;
```

Вещественные числа

Форматы вывода:

```
float x = 123.456;  
printf("%f\n", x);  
printf("%10.2f\n", x);
```

6 цифр в дробной части

123.456001

____123.46

всего знаков

в дробной части

Стандартные функции

```
#include <math.h>
```

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»

Случайные числа на компьютере

```
#include <stdlib.h>
```

Генератор на отрезке [0,RAND_MAX]:

```
int X, Y;  
X = rand(); // псевдослучайное число  
Y = rand()  // это уже другое число!
```

англ. *random* – случайный

Случайные числа на компьютере

Целые числа на отрезке $[a,b]$:

```
int X, Y;  
X = a + rand() % (b - a + 1);  
Y = a + rand() % (b - a + 1);
```

$[0, b-a]$

Задачи

«А»: Ввести с клавиатуры три целых числа, найти их сумму, произведение и среднее арифметическое.

Пример:

Введите три целых числа :

5 7 8

$$5+7+8=20$$

$$5*7*8=280$$

$$(5+7+8) / 3=6.667$$

Задачи

«В»: Ввести с клавиатуры координаты двух точек (А и В) на плоскости (вещественные числа). Вычислить длину отрезка АВ.

Пример:

Введите координаты точки А:

5.5 3.5

Введите координаты точки В:

1.5 2

Длина отрезка АВ = 4.272

Задачи

«С»: Получить случайное трехзначное число и вывести через запятую его отдельные цифры.

Пример:

Получено число 123.

Его цифры 1, 2, 3.

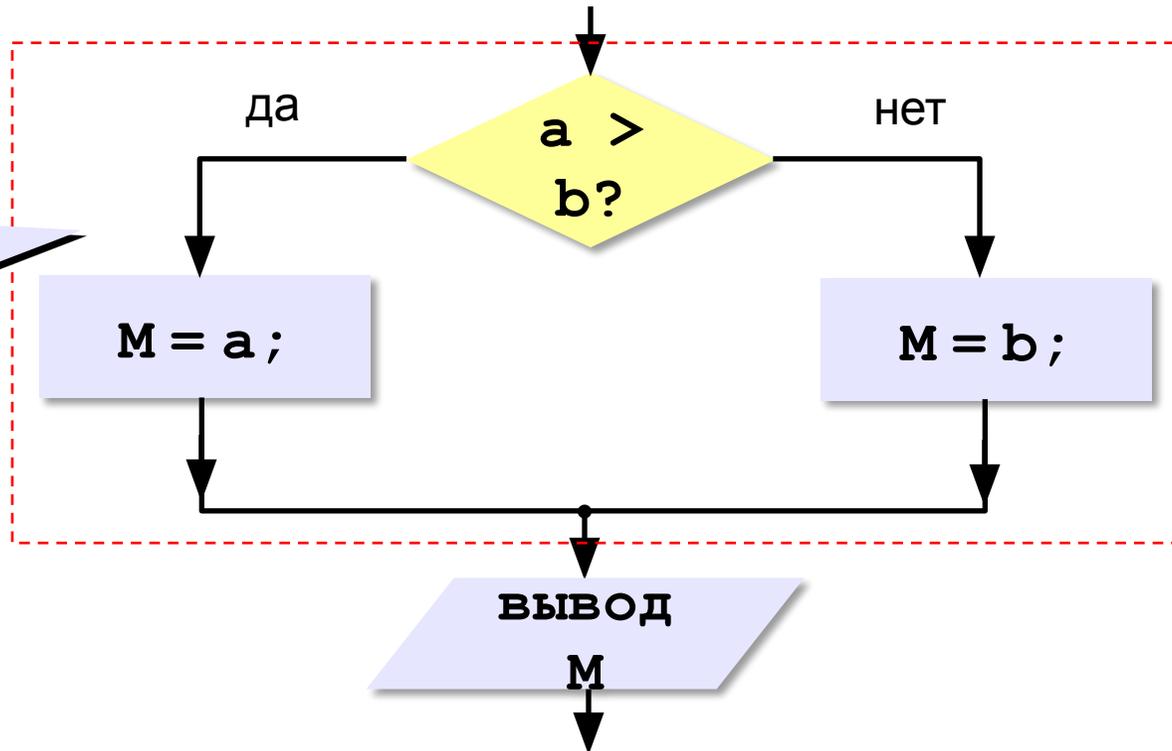
Ветвления

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ C++

Условный оператор

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

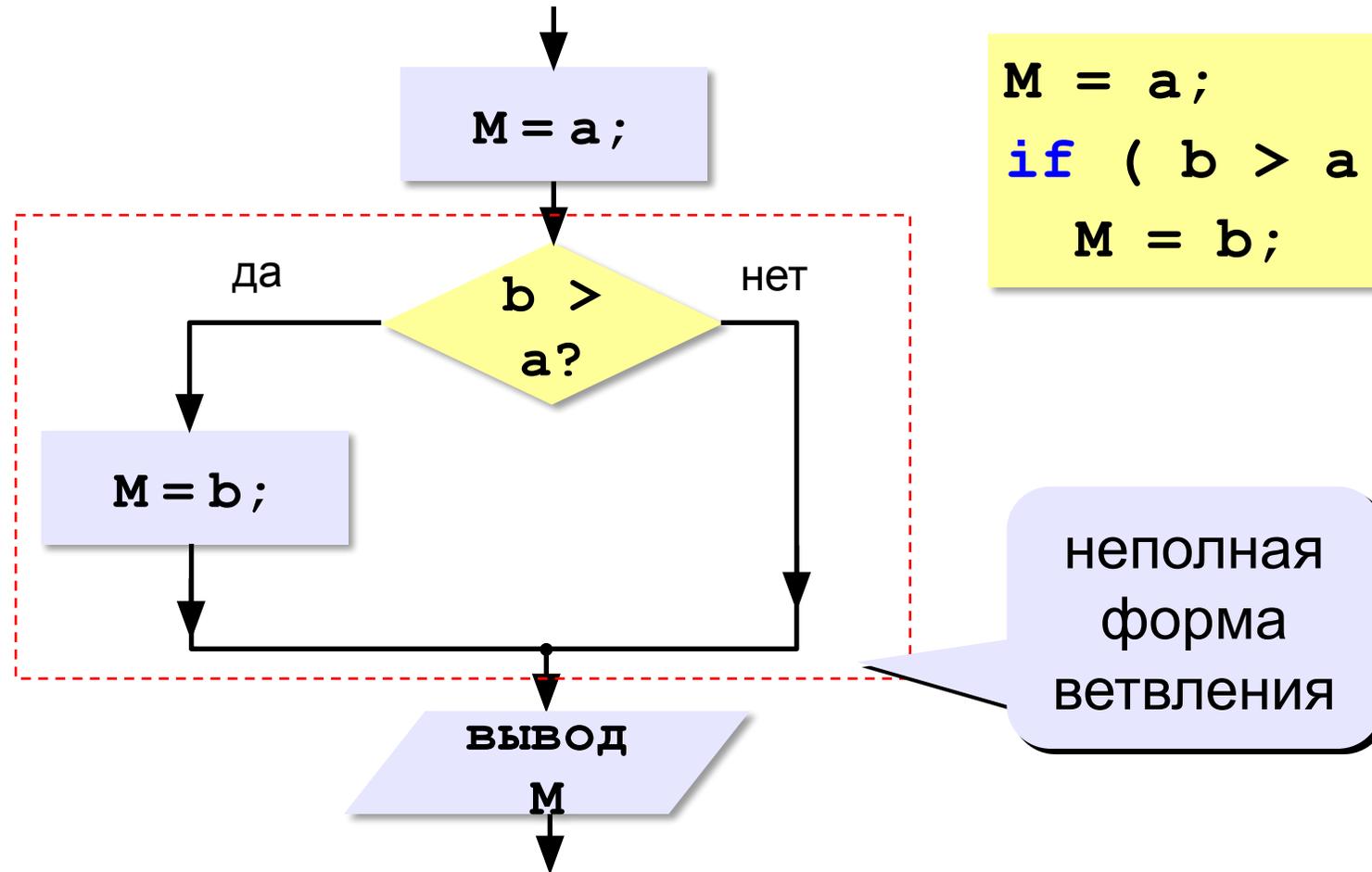
полная форма ветвления



? Если $a = b$?

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

Условный оператор: неполная форма



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

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 Вовы. Кто из них старше?

? Сколько вариантов?

```
if ( a == b )
    printf("Одного возраста");
else
    if ( a > b )
        printf("Андрей старше");
    else
        printf("«Вова старше");
```

? Зачем нужен?

вложенный
условный оператор

Задачи

«А»: Ввести три целых числа, найти максимальное из них.

Пример:

Введите три целых числа:

1 5 4

Максимальное число 5

Задачи

«В»: Ввести пять целых чисел, найти максимальное из них.

Пример:

Введите пять целых чисел:

1 5 4 3 2

Максимальное число 5

Задачи

«С»: Ввести последовательно возраст Антона, Бориса и Виктора. Определить, кто из них старше.

Пример:

Возраст Антона: 15

Возраст Бориса: 17

Возраст Виктора: 16

Ответ: Борис старше всех.

Пример:

Возраст Антона: 17

Возраст Бориса: 17

Возраст Виктора: 16

Ответ: Антон и Борис старше Виктора.

Сложные условия

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

сложное условие

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

&& «И»

|| «ИЛИ»

! «НЕ»

Приоритет :

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

Задачи

«А»: Напишите программу, которая получает три числа и выводит количество одинаковых чисел в этой цепочке.

Пример:

Введите три числа:

5 5 5

Все числа одинаковые.

Пример:

Введите три числа:

5 7 5

Два числа одинаковые.

Пример:

Введите три числа:

5 7 8

Нет одинаковых чисел.

Задачи

«В»: Напишите программу, которая получает номер месяца и выводит соответствующее ему время года или сообщение об ошибке.

Пример:

Введите номер месяца :

5

Весна .

Пример:

Введите номер месяца :

15

Неверный номер месяца .

Задачи

«С»: Напишите программу, которая получает возраст человека (целое число, не превышающее 120) и выводит этот возраст со словом «год», «года» или «лет». Например, «21 год», «22 года», «25 лет».

Пример:

Введите возраст: **18**
Вам 18 лет.

Пример:

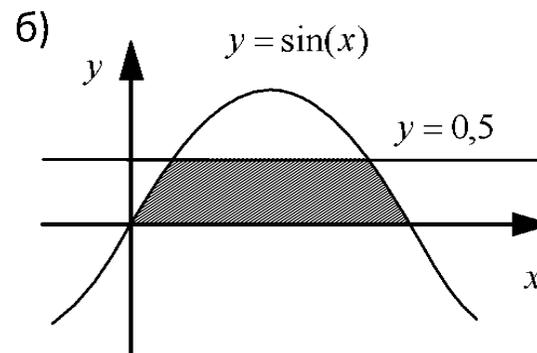
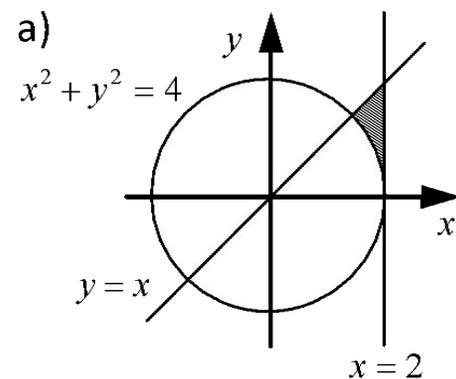
Введите возраст: **21**
Вам 21 год.

Пример:

Введите возраст: **22**
Вам 22 года.

Задачи

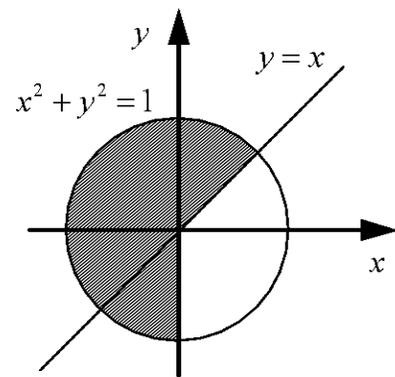
«А»: Напишите условие, которое определяет заштрихованную область.



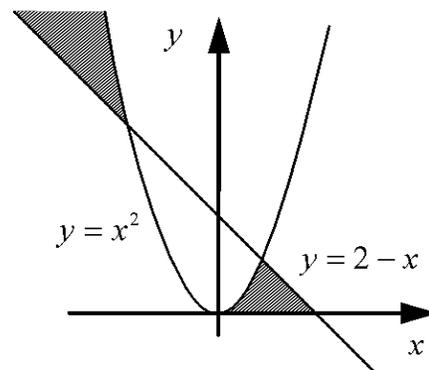
Задачи

«В»: Напишите условие, которое определяет заштрихованную область.

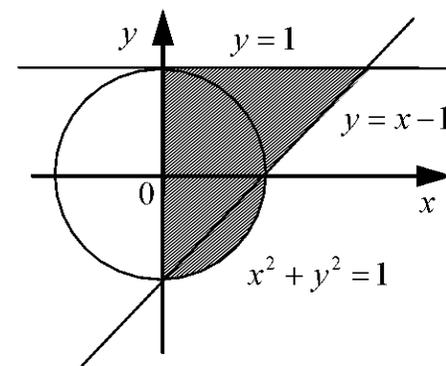
а)



б)

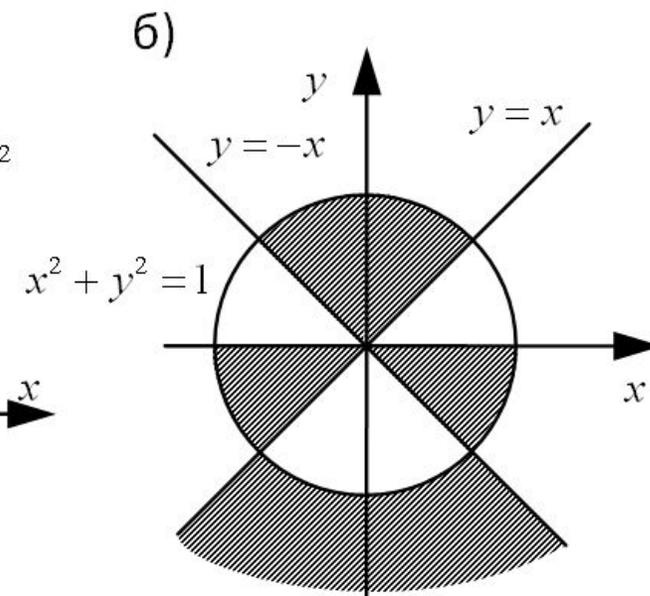
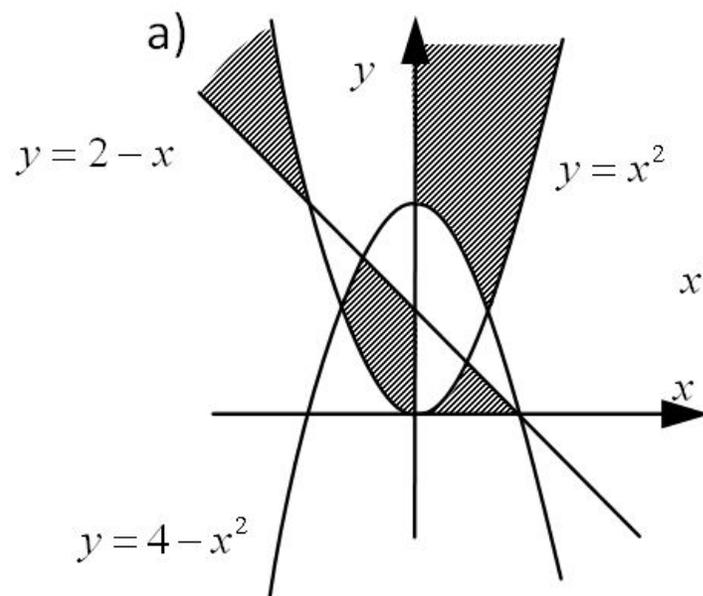


в)



Задачи

«С»: Напишите условие, которое определяет заштрихованную область.



Множественный выбор

```
if (m == 1) printf("январь");  
...  
if (m == 12) printf("декабрь");
```

```
switch ( m ) {  
    case 1: printf("январь");  
            break;  
    ...  
    case 12: printf("декабрь");  
            break;  
    default: printf("ошибка");  
}
```

Множественный выбор

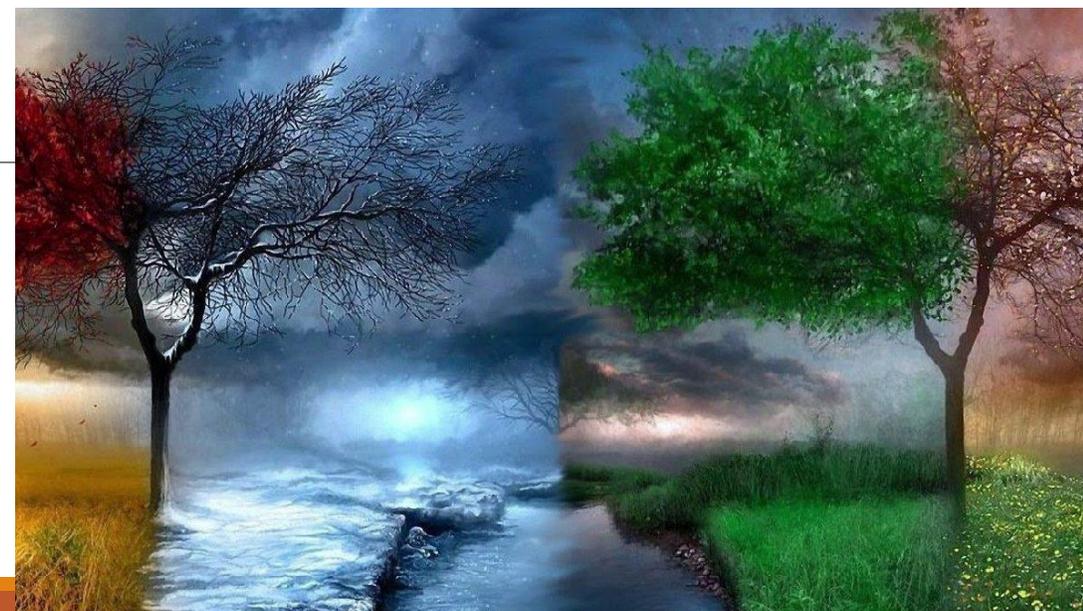
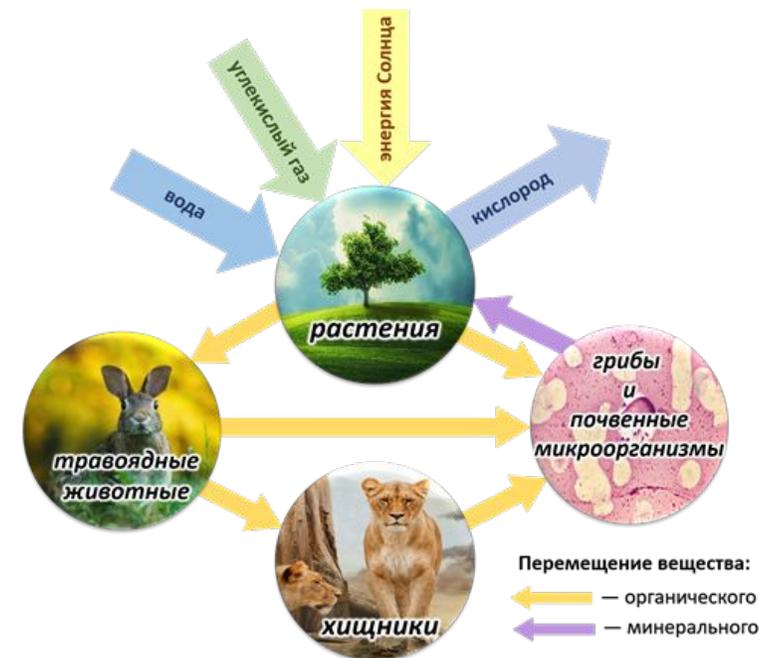
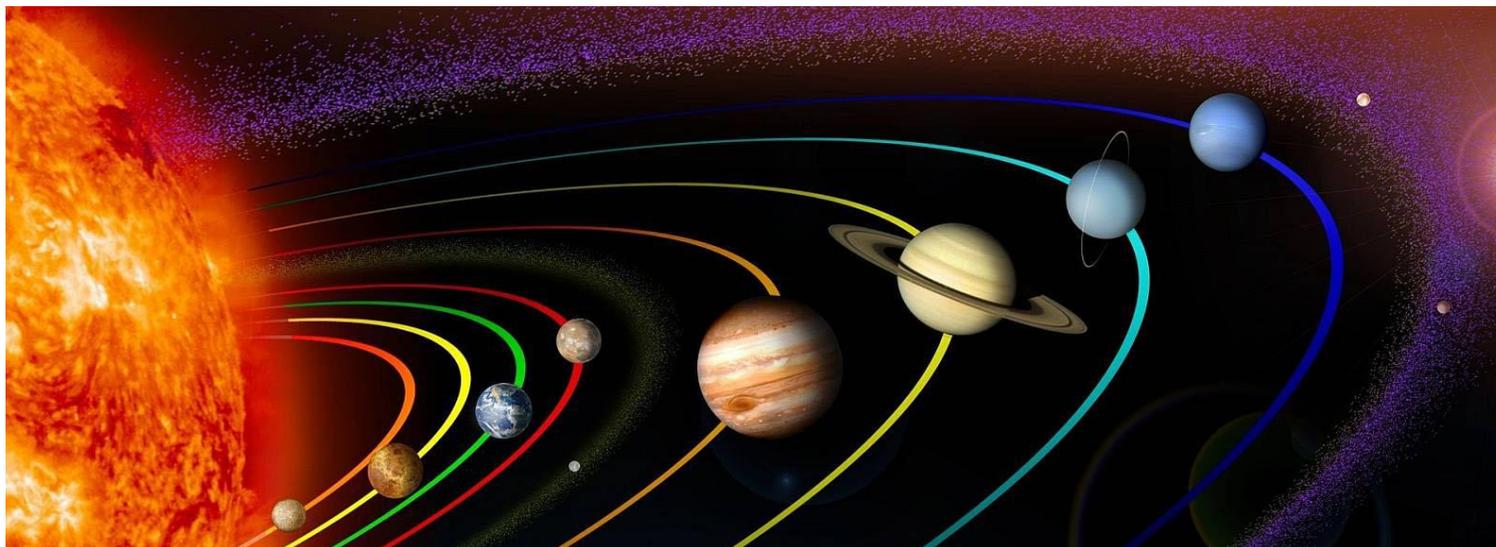
Если не ставить **break**:

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

При $m = 2$: февральмартошибка

Циклические алгоритмы

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ C++



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)

Задача. Вывести на экран 10 раз слово «Привет».



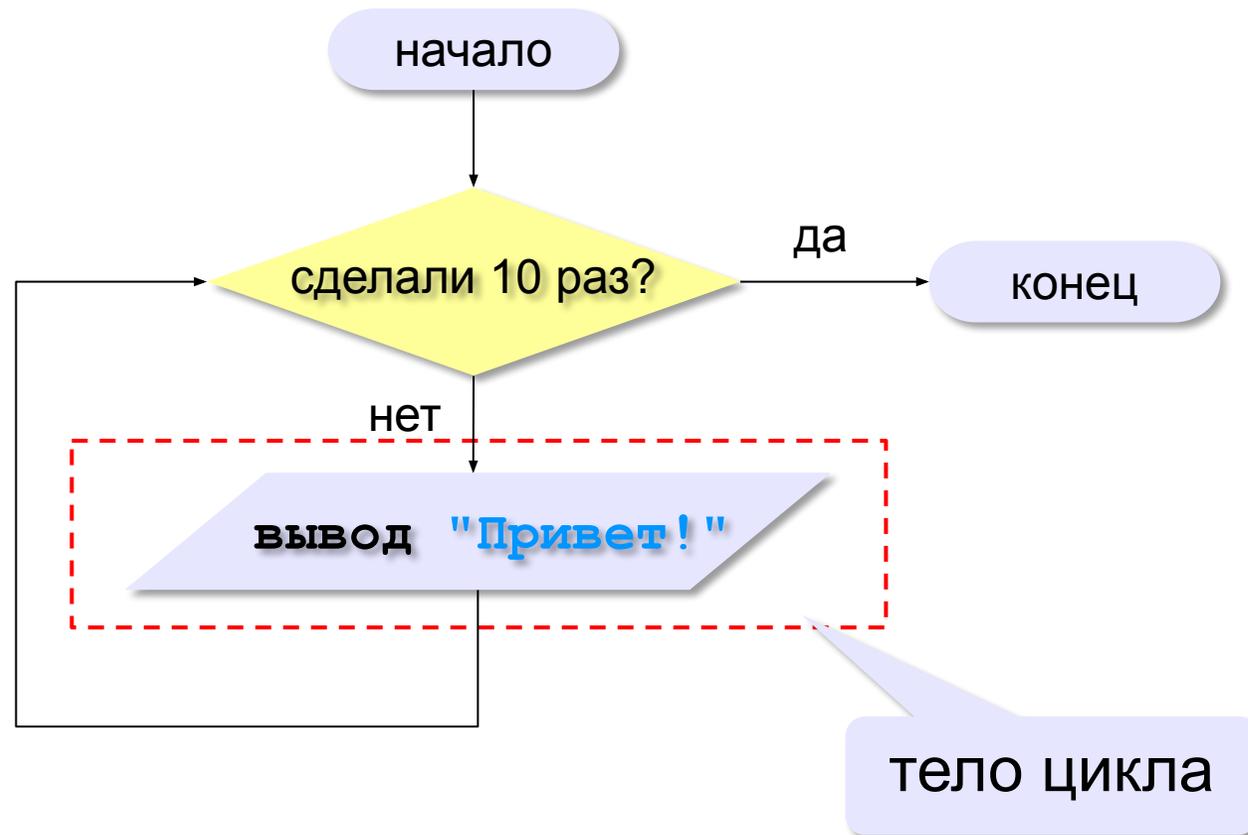
Можно ли решить известными методами?

Повторения в программе

```
printf ("Привет\n") ;  
printf ("Привет\n") ;  
...  
printf ("Привет\n") ;
```



Блок-схема цикла



Как организовать цикл?

```
счётчик = 0
пока счётчик < 10
    printf("Привет\n");
    увеличить счётчик на 1
```

Добавить кусок кода и совместить с блоксхемой, чтоб было последовательно

Цикл с условием

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

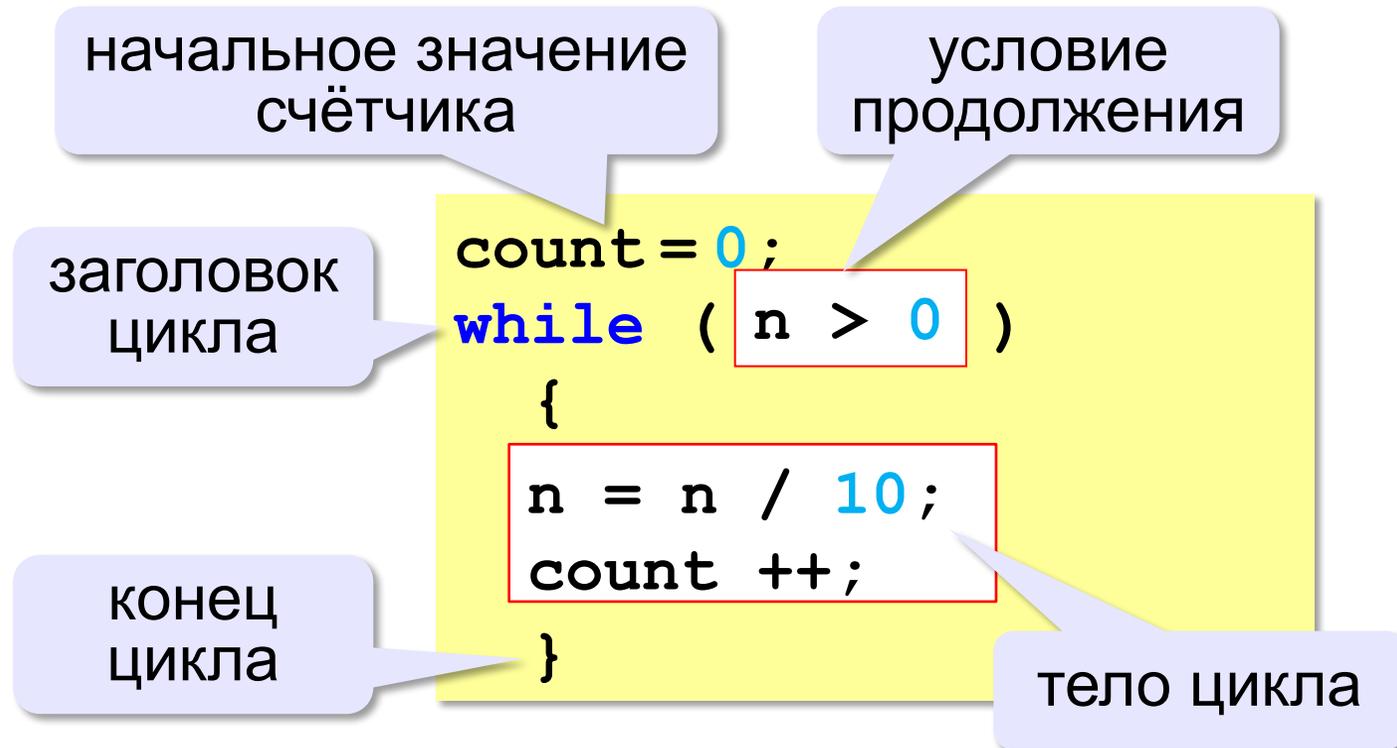
```
счётчик = 0
пока n > 0
    отсечь последнюю цифру n
    увеличить счётчик на 1
```

n	счётчик
1234	0

? Как отсечь последнюю цифру?

? Как увеличить счётчик на 1?

Цикл с условием



! Цикл с предусловием – проверка на входе в цикл!

Цикл с условием

При известном количестве шагов:

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

Защипливание:

```
k = 0;
while ( k < 10 )
{
    printf ( "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 ) a --;
```

зацикливание

Цикл с постусловием

заголовок
цикла

```
do
```

```
{
```

```
printf("Введите n > 0: ");
```

```
scanf( "%d", &n );
```

```
}
```

```
while ( n <= 0 );
```

тело цикла

условие
продолжения

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

Задачи

«А»: Напишите программу, которая получает два целых числа A и B ($0 < A < B$) и выводит квадраты всех натуральных чисел в интервале от A до B .

Пример:

Введите два целых числа:

10 12

10*10=100

11*11=121

12*12=144

Задачи

«В»: Напишите программу, которая получает два целых числа и находит их произведение, не используя операцию умножения. Учтите, что числа могут быть отрицательными.

Пример:

Введите два числа:

10 -15

$10 * (-15) = -150$

Задачи

«С»: Ввести натуральное число N и вычислить сумму всех чисел Фибоначчи, меньших N . Предусмотрите защиту от ввода отрицательного числа N .

Пример:

Введите число N :

10000

Сумма 17710

Задачи

«А»: Ввести натуральное число и найти сумму его цифр.

Пример:

Введите натуральное число:

12345

Сумма цифр 15.

Задачи

«В»: Ввести натуральное число и определить, верно ли, что в его записи есть две одинаковые цифры, стоящие рядом.

Пример:

Введите натуральное число :

12342

Нет .

Пример:

Введите натуральное число :

12245

Да .

Задачи

«С»: Ввести натуральное число и определить, верно ли, что в его записи есть две одинаковые цифры (не обязательно стоящие рядом).

Пример:

Введите натуральное число:

12342

Да.

Пример:

Введите натуральное число:

12345

Нет.

Цикл с переменной

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



Можно ли сделать с циклом «пока»?

```
i = 1;  
n = 2;  
while ( i <= 10 )  
{  
    printf ("%d\n", n);  
    n *= 2;  
    i ++;  
}
```

```
n = 2;  
for ( int i=0; i<10; ++i )  
{  
    printf ("%d\n", n);  
    n *= 2;  
}
```

ЦИКЛ С
переменной

Цикл с переменной: другой шаг

```
for ( i = 10; i >= 1; i-- )  
    printf( "%d\n", i*i );
```



Что получится?

1
9
25
49
81

```
for ( i = 1; i <= 10; i += 2 )  
    printf( "%d\n", i*i );
```

100
81
64
49
36
25
16
9
4
1

```
a = 1;
```

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

a = 4

```
a = 1;
```

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

a = 1

```
a = 1;
```

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

заЦИКЛИВАНИЕ

```
a = 1;
```

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

a = 4

Задачи

«А»: Найдите все пятизначные числа, которые при делении на 133 дают в остатке 125, а при делении на 134 дают в остатке 111.

Задачи

«В»: Натуральное число называется **числом Армстронга**, если сумма цифр числа, возведенных в N-ную степень (где N – количество цифр в числе) равна самому числу. Например, $153 = 1^3 + 5^3 + 3^3$.
Найдите все трёхзначные Армстронга.

Задачи

«С»: Натуральное число называется автоморфным, если оно равно последним цифрам своего квадрата. Например, $25^2 = 625$. Напишите программу, которая получает натуральное число N и выводит на экран все автоморфные числа, не превосходящие N.

Пример:

Введите N:

1000

1*1=1

5*5=25

6*6=36

25*25=625

76*76=5776

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!

 Что значит «простое число»?

Вложенные циклы

```
for ( n = 2; n <= 1000; n++ )  
{  
    count = 0;  
    for ( k = 2; k < n; k++ )  
        if ( n % k == 0 )  
            count++;  
    if ( count == 0 )  
        printf("%d\n", n);  
}
```

ВЛОЖЕННЫЙ ЦИКЛ

Вложенные циклы

```
for ( i = 1; i <= 4; i++ )  
  {  
    for ( k = 1; k <= i; k++ )  
      {  
        ...  
      }  
  }
```

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



Переменная внутреннего цикла изменяется быстрее!

Задачи

«А»: Напишите программу, которая получает натуральные числа A и B ($A < B$) и выводит все простые числа в интервале от A до B .

Пример:

Введите границы диапазона:

10 20

11 13 17 19

Задачи

«В»: В магазине продается мастика в ящиках по 15 кг, 17 кг, 21 кг. Как купить ровно 185 кг мастики, не вскрывая ящики? Сколькими способами можно это сделать?

Задачи

«С»: Ввести натуральное число N и вывести все натуральные числа, не превосходящие N и делящиеся на каждую из своих цифр.

Пример:

Введите N :

15

1 2 3 4 5 6 7 8 9 11 12 15

Процедуры

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ C++

Зачем нужны процедуры?

МНОГО раз!

```
printf ( "Ошибка" );
```

```
void Error()  
{  
    printf ("Ошибка");  
}
```

```
main()  
{  
    int n;  
    scanf ( "%d", &n );  
    if ( n < 0 ) Error();  
    ...  
}
```

ВЫЗОВ
процедуры

Что такое процедура?

Процедура – вспомогательный алгоритм, который выполняет некоторые действия.

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

Процедура с параметрами

Задача. Вывести на экран запись целого числа (0..255) в 8-битном двоичном коде.

много раз!

Алгоритм:

$$178 \Rightarrow 10110010_2$$

? Как вывести первую цифру?

? Как вывести вторую цифру?

n = $\overset{7}{1} \overset{6}{0} \overset{5}{1} \overset{4}{1} \overset{3}{0} \overset{2}{0} \overset{1}{1} \overset{0}{0}_2$ разряды

n / 128

n % 128

n1 / 64

Процедура с параметрами

```
void printBin ( int n )  
{  
  int k;  
  k = 128;  
  while ( k > 0 )  
  {  
    printf ( "%d" , n / k );  
    n = n % k;  
    k = k / 2;  
  }  
}
```

локальные
переменные

Параметры – данные,
изменяющие работу
процедуры.

```
main()  
{  
  printBin ( 99 );  
}
```

значение параметра
(аргумент)

Несколько параметров

```
void printSred ( int a, int b )  
{  
    printf ( "%f", (a+b)/2. );  
}
```

Задачи

«А»: Напишите процедуру, которая принимает параметр – натуральное число N – и выводит на экран линию из N символов '-'.

Пример:

Введите N :

10

Задачи

«В»: Напишите процедуру, которая выводит на экран в столбик все цифры переданного ей числа, начиная с первой.

Пример:

Введите натуральное число:

1234

1

2

3

4

Задачи

«С»: Напишите процедуру, которая выводит на экран запись переданного ей числа в римской системе счисления.

Пример:

Введите натуральное число:

2013

ММХІІІ

Изменяемые параметры

Задача. Написать процедуру, которая меняет местами значения двух переменных.

```
void Swap ( int a, int b )  
{  
    int c;  
    c = a; a = b; b = c;  
}
```

передача по
значению



Процедура работает с копиями переданных значений параметров!



Почему не работает?

```
main()  
{  
    int x = 2, y = 3;  
    Swap ( x, y );  
    printf ( "%d %d", x, y );  
}
```

2 3

Изменяем

передаются адреса
переменных

передача по
адресу

```
void Swap ( int * adrA, int * adrB )  
{  
    int c;  
    c = *adrA; *adrA = *adrB; *adrB = c;  
}
```

значение переменной
по адресу

Вызов:

```
int a, b;  
Swap ( &a, &b );    // правильно  
Swap ( 2, 3 );     // неправильно  
Swap ( &a, b+3 );  // неправильно
```

Изменяемые параметры (C++)

переменные могут изменяться

```
void Swap ( int & a, int & b )  
{  
    int c;  
    c = a; a = b; b = c;  
}
```

передача по
ССЫЛКЕ

```
int a, b;  
Swap (a, b);    // правильно  
Swap (2, 3);   // неправильно  
Swap (a, b+3); // неправильно
```

Задачи

«А»: Напишите процедуру, которая переставляет три переданные ей числа в порядке возрастания.

Пример:

Введите три натуральных числа:

10 15 5

5 10 15

Задачи

«В»: Напишите процедуру, которая сокращает дробь вида M/N . Числитель и знаменатель дроби передаются как изменяемые параметры.

Пример:

Введите числитель и знаменатель дроби:

25 15

После сокращения: 5/3

Задачи

«С»: Напишите процедуру, которая вычисляет наибольший общий делитель и наименьшее общее кратное двух натуральных чисел и возвращает их через изменяемые параметры.

Пример:

Введите два натуральных числа :

10 15

НОД (10 , 15) =5

НОК (10 , 15) =30

Функции

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ C++

Что такое функция?

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

Задача. Написать функцию, которая вычисляет сумму цифр числа.

Алгоритм:

```
сумма = 0
пока n != 0
    сумма = сумма + n % 10
    n = n / 10
```

Сумма цифр числа

```
int sumDigits ( int n )  
{  
    int sum = 0;  
    while ( n != 0 )  
    {  
        sum += n % 10;  
        n /= 10;  
    }  
    return sum;  
}
```

тип результата

передача
результата

Использование функций

```
x = 2*sumDigits (n+5) ;  
z = sumDigits (k) + sumDigits (m) ;  
if ( sumDigits (n) % 2 == 0 )  
{  
    printf ( "Сумма цифр чётная\n" ) ;  
    printf ( "Она равна %d", sumDigits (n) ) ;  
}
```



Функция, возвращающая целое число, может использоваться везде, где и целая величина!

Задачи

«А»: Напишите функцию, которая находит наибольший общий делитель двух натуральных чисел.

Пример:

Введите два натуральных числа:

7006652 112307574

НОД(7006652, 112307574) = 1234.

Задачи

«В»: Напишите функцию, которая определяет сумму цифр переданного ей числа.

Пример:

Введите натуральное число:

123

Сумма цифр числа 123 равна 6.

Задачи

«С»: Напишите функцию, которая «переворачивает» число, то есть возвращает число, в котором цифры стоят в обратном порядке.

Пример:

Введите натуральное число:

1234

После переворота: 4321.

Логические функции

Задача. Найти все простые числа в диапазоне от 2 до 100.

```
main ()
{
    int i;
    for ( i = 2; i <= 100; i++)
        if ( isPrime (i) )
            printf ( "%d\n" , i );
}
```

функция,
возвращающая
логическое
значение (да/нет)

Функция: простое число или нет?

```
bool isPrime ( int n )
{
    int count=0, k=2;
    while ( k*k <=n && count==0 )
    {
        if ( n % k == 0 )
            count ++;
        k ++;
    }
    return (count == 0);
}
```

if (count == 0)
 return true;
else return false;

Логические функции: использование



Функция, возвращающая логическое значение, может использоваться везде, где и логическая величина!

```
scanf ( "%d" , &n ) ;  
while ( isPrime (n) )  
{  
    printf ("простое число\n") ;  
    scanf ( "%d" , &n ) ;  
}
```

Задачи

«А»: Напишите логическую функцию, которая определяет, является ли переданное ей число совершенным, то есть, равно ли оно сумме своих делителей, меньших его самого.

Пример:

Введите натуральное число:

28

Число 28 совершенное.

Пример:

Введите натуральное число:

29

Число 29 не совершенное.

Задачи

«В»: Напишите логическую функцию, которая определяет, являются ли два переданные ей числа взаимно простыми, то есть, не имеющими общих делителей, кроме 1.

Пример:

Введите два натуральных числа:

28 15

Числа 28 и 15 взаимно простые.

Пример:

Введите два натуральных числа:

28 16

Числа 28 и 16 не взаимно простые.

Задачи

«С»: Простое число называется гиперпростым, если любое число, получающееся из него откидыванием нескольких цифр, тоже является простым. Например, число 733 – гиперпростое, так как и оно само, и числа 73 и 7 – простые. Напишите логическую функцию, которая определяет, верно ли, что переданное ей число – гиперпростое. Используйте уже готовую функцию `isPrime`, которая приведена в учебнике.

Пример:

Введите натуральное число:

733

Число 733 гиперпростое.

Пример:

Введите натуральное число:

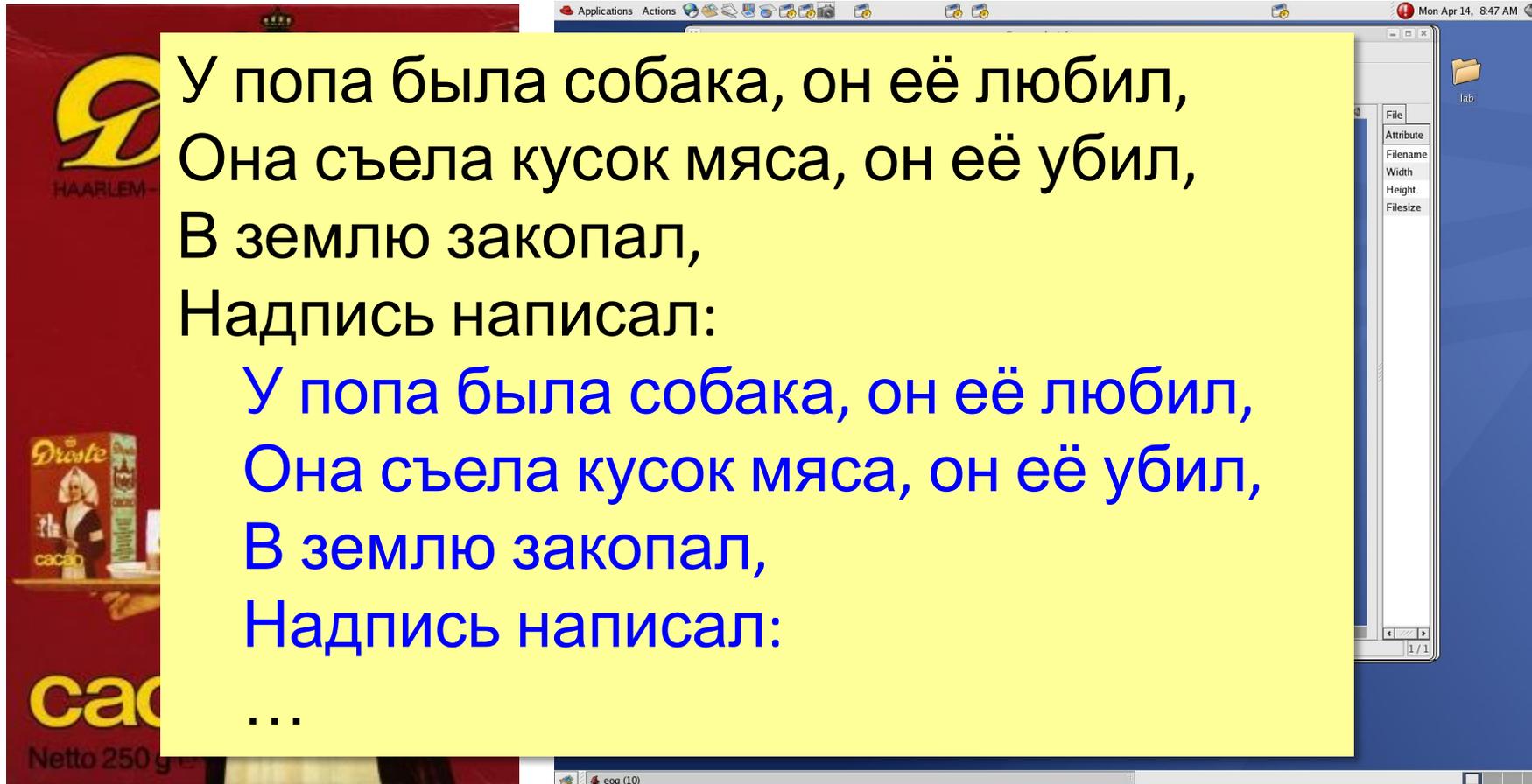
19

Число 19 не гиперпростое.

Рекурсия

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ C++

Что такое рекурсия?



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.

Что такое рекурсия?

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, ...

Что такое рекурсия?

Числа Фибоначчи:

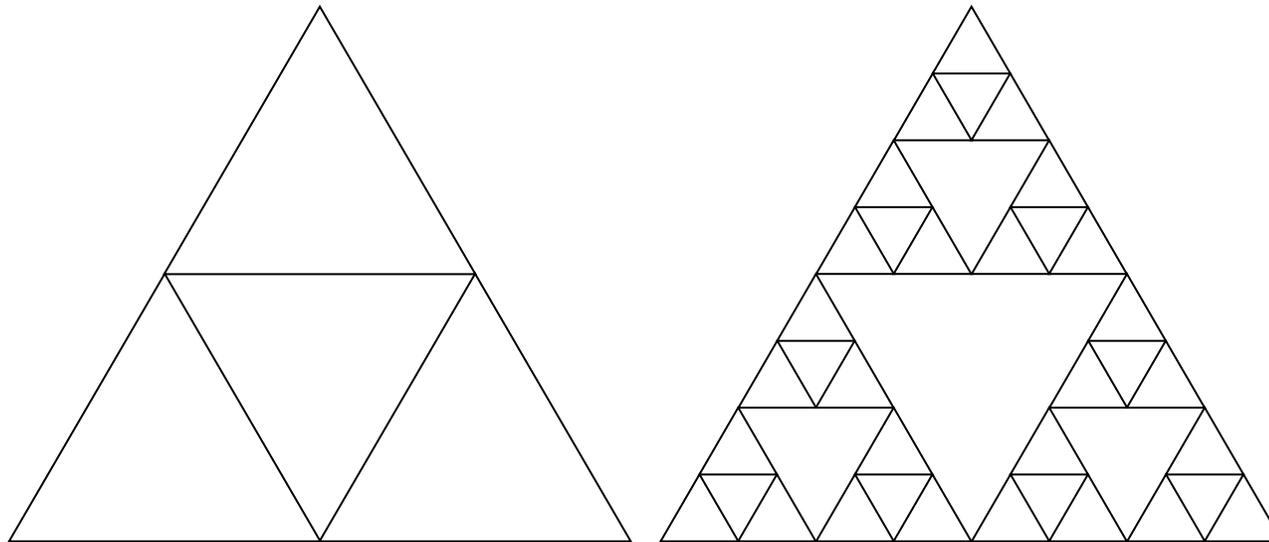
- $F_1 = F_2 = 1$
- $F_n = F_{n-1} + F_{n-2}$ при $n > 2$

1, 1, 2, 3, 5, 8, 13, 21, 34, ...

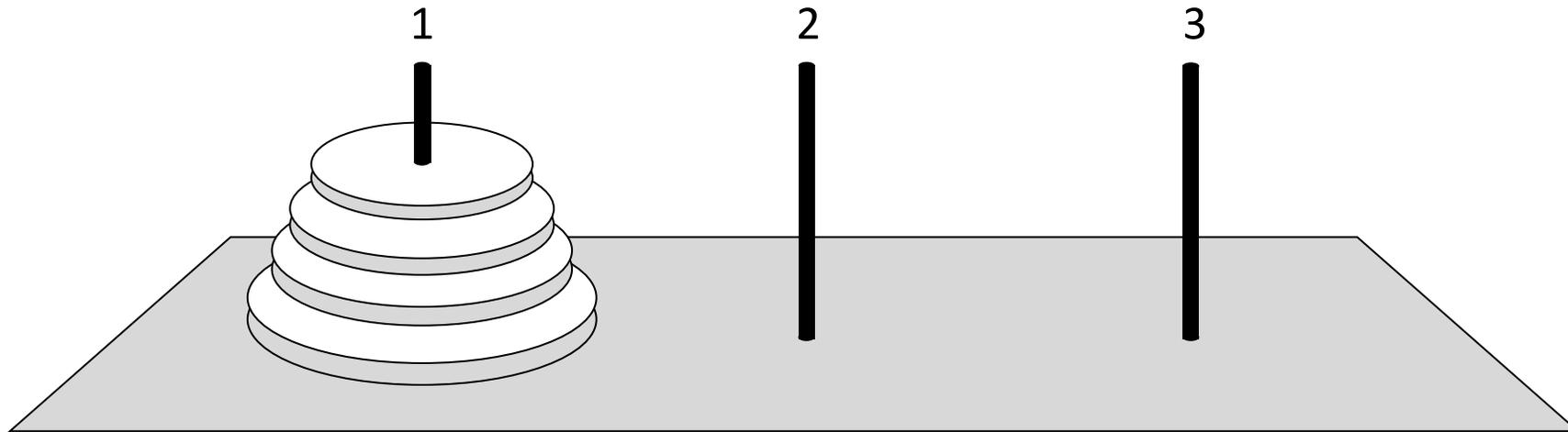
Фракталы

Фракталы – геометрические фигуры, обладающие самоподобием.

Треугольник Серпинского:



Ханойские башни



- за один раз переносится один диск
- класть только меньший диск на больший
- третий стержень вспомогательный

Вывод двоичного кода числа

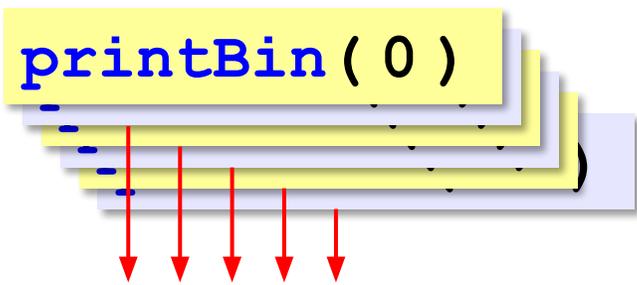
```
void printBin( int n )  
{  
    if ( n == 0 ) return;  
    printBin( n / 2 );  
    printf ( "%d", n % 2 );  
}
```

условие выхода из
рекурсии

напечатать все
цифры, кроме
последней

ВЫВЕСТИ
последнюю цифру

`printBin(0)`



Вычисление суммы цифр числа

```
int sumDig ( int n )  
{  
    int sum;  
    sum = n % 10;  
    if ( n >= 10 )  
        sum += sumDig ( n / 10 );  
    return sum;  
}
```

последняя цифра

рекурсивный вызов



Где условие окончания рекурсии?

sumDig (1234)

4 + sumDig (123)

4 + 3 + sumDig (12)

4 + 3 + 2 + sumDig (1)

4 + 3 + 2 + 1

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

Задачи

«А»: Напишите рекурсивную функцию, которая вычисляет НОД двух натуральных чисел, используя модифицированный алгоритм Евклида.

Пример:

Введите два натуральных числа:

7006652 112307574

НОД (7006652, 112307574) = 1234 .

Задачи

«В»: Напишите рекурсивную функцию, которая раскладывает число на простые сомножители.

Пример:

Введите натуральное число:

378

$378 = 2 * 3 * 3 * 3 * 7$

Задачи

«С»: Дано натуральное число N . Требуется получить и вывести на экран количество всех возможных *различных* способов представления этого числа в виде суммы натуральных чисел (то есть, $1 + 2$ и $2 + 1$ – это один и тот же способ разложения числа 3). Решите задачу с помощью рекурсивной функции.

Пример:

Введите натуральное число:

4

Количество разложений: 4.