

C++

# **Основы языка C++**

## **Ч1**

Методист курса: разработчик АО НПО “РусБиТех”.  
Гудзенко Артем Валерьевич

# Создание архива курса

- Весь материал после занятия будет храниться на гите по этой ссылке ... [https://github.com/Danykrane/EU\\_C\\_flow16](https://github.com/Danykrane/EU_C_flow16)
- **При сдаче дз (можно мне в тг) необходимо указывать:**
  - Номер дз,
  - Свое имя!
- **Пример:** "Высылаю дз1 Дима"

# Структура курса по C++

**Занятия:** 64 часа – 32 занятия (4 модуля)

**Домашние работы:** 32 задания (6 + 8 + 8 + 10) на 4 модуля

**1 модуль:** Основы языка C++

**2 модуль:** ООП + шаблоны проектирования

**3 модуль:** Контейнеры STL.

**4 модуль:** Промышленная разработка.

# Совместная Практика

Три/пять совместных заданий (можно выполнять и одному):

- На 3 занятия (квадратные матрицы)
- На 6 занятия (указатели и ссылки)
- На 10 занятия (Классы ООП)
- На 22 занятия (разработка проекта с определенным шаблоном проектирования)
- На 30 занятия (разработка приложения QT5)

**Из чего состоит программа  
на языке C++?**

A hand-drawn blue oval frame with a double-line border, centered on the page. The text is written in the center of this frame.

**Програма на C++**

# Структура программы

```
#include <iostream> // библиотека ввода вывода

using namespace std;

int main() {

    cout << "My name is - Artyom\nGlad to see you!\n";

    // "\n" – символ переноса строки

    cout << "My name is - Artyom"<<endl;
    cout <<"Glad to see you!";
}
```

# Структура программы

```
#include <iostream> // библиотека ввода вывода
```

- Подключение библиотек

```
using namespace std;
```

- Пространство имен

```
int main() {  
    // основной код тут  
}
```

- Основная функция





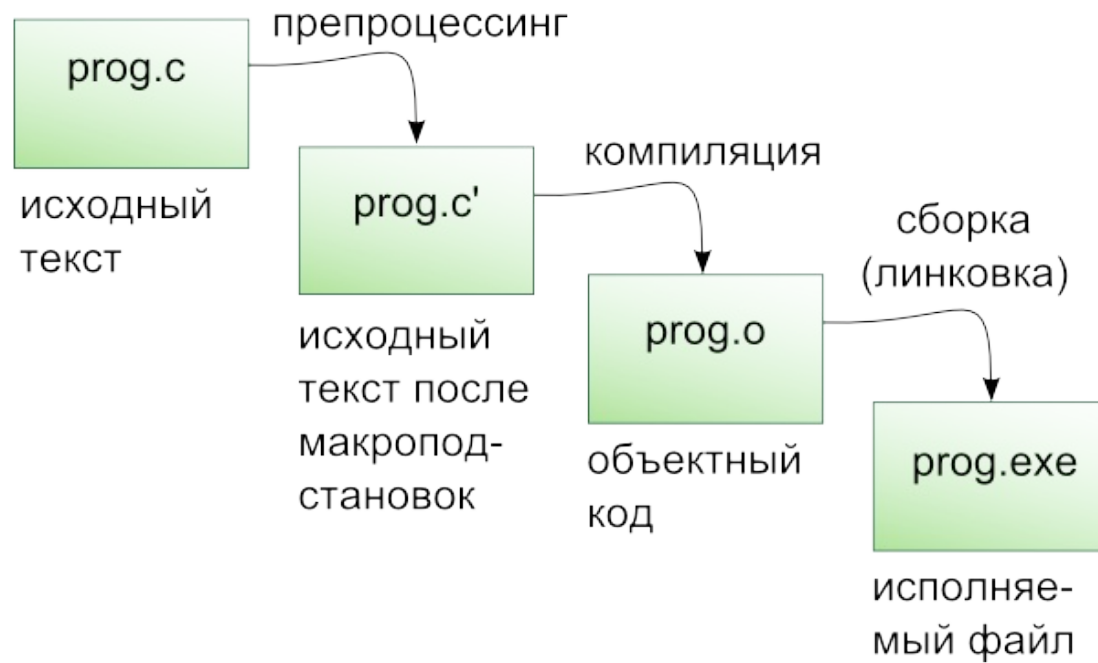
**Как проходит компиляция?**

A hand-drawn blue oval frame with a slightly irregular, sketchy border, centered on the page. It contains the word 'Компиляция' in bold black text.

**Компиляция**

# Компиляция программы

Этапы после того, как  
нажмем на *run code*





**Ввод и вывод**

# ВВОД И ВЫВОД

## Вывод

```
#include <iostream> // библиотека ввода вывода
```

- Библиотека для операций ввода и вывода в консоль

I – input (ввод)  
o – output (вывод)  
Stream (поток)

**iostream**

# Ввод и вывод

## Вывод

```
#include <iostream> // библиотека ввода вывода

using namespace std;

int main() {

    cout << "My name is - Artyom"<<endl;
    cout <<"Glad to see you!";

    cout << "My name is - Artyom\nGlad to see you!\n";

    // "\n" - символ переноса строки

}
```

с подключенным  
пространством имен std

```
#include <iostream> // библиотека ввода вывода

int main() {

    std::cout << "My name is - Artyom"<<std::endl;
    std::cout <<"Glad to see you!";

    std::cout << "My name is - Artyom\nGlad to see you!\n";

    // "\n" - символ переноса строки
}
```

без подключения  
пространства имен std

# Ввод и вывод

## Ввод

```
#include <iostream> // библиотека ввода вывода

using namespace std;

int main() {

    int age; //создани переменной для хранения возраста
    cin >> age; // ввод в переменную age значения с клавиатуры

}
```

# Ввод и вывод

## Ввод + Вывод

```
#include <iostream> // библиотека ввода вывода

using namespace std;

int main() {

    int age; //создани переменной для хранения возраста
    cout<<"Сколько вам лет?"<<endl;
    cin >> age; // ввод в переменную age значения с клавиатуры

    cout << "Ваш возраст: "<<age<<endl;
}
```



# Ввод и вывод

## Ввод + Вывод

```
Сколько вам лет?
```

```
22
```

```
Ваш возраст: 22
```

```
artemgudzenko@MacBook-Air-Artem 1_less %
```

# Задача

**Задача:** выведите 10 раз строку \*\*\*\*\*. После вывода строки перенос на новую.



**Что такое переменная?**



**Приведите пример из жизни**



**Переменная**

# Переменная

**Переменная** - это именованная область памяти, к которой мы имеем доступ из программы;

A hand-drawn blue oval frame with a double-line border, centered on the page. The text is written in the center of this frame.

# **Типы данных**

# Типы данных

```
graph TD; A[Типы данных] --> B[• Целые числа]; A --> C[• Символы]; A --> D[• Строки]; A --> E[• Неопределенный тип]; A --> F[• Вещественные числа];
```

The diagram illustrates the classification of data types. A central title 'Типы данных' (Data Types) is connected by green arrows to five categories: 'Целые числа' (Integer numbers), 'Символы' (Symbols), 'Строки' (Strings), 'Неопределенный тип' (Undefined type), and 'Вещественные числа' (Real numbers). The entire content is framed by a blue border.

• Символы

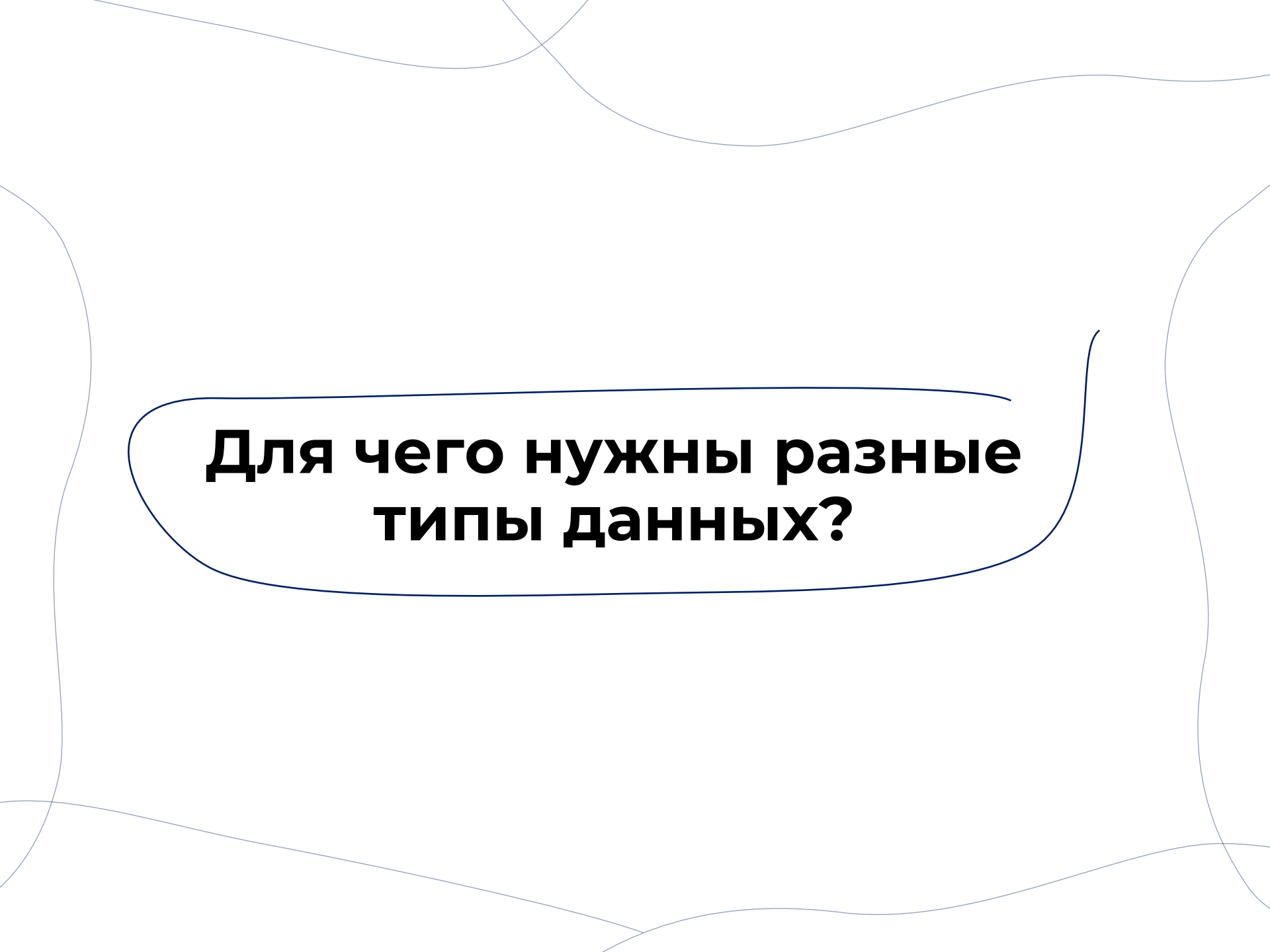
• Целые числа

• Строки

• Неопределенный тип

• Вещественные числа





**Для чего нужны разные  
типы данных?**

# Типы данных

## 1. Целые числа

Имя типа	Подтипы	Размер , байт	Интервал значений
<b>char</b> или <b>_int[8]</b>	<b>[signed] char</b> <b>unsigned char</b>	1	-128..127 0..255
<b>short</b> или <b>_int[16]</b>	<b>[signed] short</b> <b>unsigned short</b>	2	-32768..32767 0..65535
<b>int</b> или <b>long</b> или <b>_int[32]</b>	<b>[signed] int</b> <b>unsigned int</b> <b>[signed] long</b> <b>unsigned long</b>	4	$-2^{31}.. 2^{31}-1$ $0.. 2^{32}-1$
<b>long long</b> или <b>_int[64]</b>	<b>[signed] long long</b> <b>unsigned long long</b>	8	$-2^{63}.. 2^{63}-1$ $0.. 2^{64}-1$
<b>bool</b>		1	false (0), true(1)

# Типы данных

## 2. Вещественные типы

Тип	Размер, байт	Значащих цифр	Минимальное положительное число	Максимальное положительное число
<b>float</b>	4	6	1.175494351e-38	3.402823466e38
<b>double (long double)</b>	8	15	2.2250738585072014 e-308	1.797693134862318 e308

## 3. Неопределенный («пустой») тип void

Для:

- нетипизированных указателей;
- функций.

# **Объявление переменных**

# Объявление переменных и констант

Формат:

**Изменчивость** **Тип** **Имя** = **Значение** ;

**Изменчивость** – описатель возможности изменения значений:

- **const** – константа – неизменяемое значение,
- **без указания** – обычная переменная

**Тип** – тип данных: **int**, **char**, **float**, **double** и т.д.;

**Имя** – само имя переменной или константы;

**Значение** – начальное значение переменной или значение константы.

# Объявление переменных и констант

## Примеры:

```
#include <iostream> // библиотека ввода вывода

using namespace std;

int main() {

    int a, b; // объявили две целые переменные
    float pi=3.14,k(5.45); // объявили + сразу инициализировали значения
    const unsigned char letter='a'; // константа – код буквы «a»
    int a=7; // объявили + инициализировали переменную 15
    const int a(1);

}
```

# Различай

**Объявление переменной** - выделение для нее памяти;

**Инициализация** – передача переменной какого-то значения (присвоили переменной определенное значение).

# Задача

**Задача:** пользователь вводит с клавиатуры свой вес, рост, величина бицепса.

Выведите результат в следующем формате:

Данные для врача получены!

Вес: 86.8 кг;

Рост: 185 см;

Размер бицепса: 35 см.





**Какие математические  
операции вы знаете?**



# **Математические операции**

# Операции

- **Арифметические:**

+ – сложение;

- – вычитание;

\* – умножение;

/ – деление.

Результат – вещественное, если хотя бы одно из чисел вещественное.

Результат – целое, если делимое и делитель целые.

% - остаток от деления целых чисел.

- **Логические:** ! (не), && (и), || (или).

- **Логические поразрядные:**

- (не), & (и), | (или), ^ (исключающее или).

- **Отношения:**

<, >, <=, >=, == (равно), != (неравно).

# Задача

**Задача:** пользователь вводит с клавиатуры свой рост, однако в стране лилипутии он становится гигантом, поэтому лилипуты сделали новую шкалу измерения: 1 см человеческий = 3.86 лилипутский.

Выведите рост пользователя по лилипутским меркам.

# Задача

**Задача:** пользователь вводит с клавиатуры свой год рождения, посчитайте сколько ему лет сейчас и выведите результат на экран. (занесите результат в новую переменную и выведите ее)

# Особенности операций

## Целочисленное деление на $10^n$

```
#include <iostream>

using namespace std;

int main() {

    int number = 123456;
    cout << number /10 <<endl; //12345
    cout << number /100 <<endl; //1234
    cout << number /1000 <<endl; //123
    cout << number /10000 <<endl; //12
    cout << number /100000 <<endl; //1
    cout << number /1000000 <<endl; //0

}
```

**Отбрасываем** от числа столько цифр, сколько степень у 10 (количество нулей числа)

# Особенности операций

## Остаток от деления на $10^n$

```
#include <iostream>

using namespace std;

int main() {

    int number = 123456;
    cout << number % 10 << endl; //6
    cout << number % 100 << endl; //56
    cout << number % 1000 << endl; //456
    cout << number % 10000 << endl; //3456
    cout << number % 100000 << endl; //23456
    cout << number % 1000000 << endl; //123456

}
```

**Берем** от числа столько цифр, сколько степень у 10  
(количество нулей числа)

# Задача

На вход подаются числа: 45 123, 1 223, 756 678, 567

**Задача:** вывести у каждого числа первую цифру и вторую справ (с конца) цифру. Каждая цифра на новой строке



# Задача

На вход подаются числа: 45 123, 567

**Задача:** вывести у каждого числа все цифры. Каждая цифра на новой строке

# Особенности операций

## Остаток от деления на 2

```
#include <iostream>

using namespace std;

int main() {

    int number1 = 6;
    int number2 = 7;
    cout << number1 % 2 << endl; //0
    cout << number2 % 2 << endl; //1

}
```

Если **при остатке от деления на 2** получается **1** – число **не четное**;

Если **при остатке от деления на 2** получается **0** – число **четное**.

# Задача

На вход подаются числа: 45 123, 567

**Задача:** доказать, что оба числа нечетные

A hand-drawn blue oval frame with a slightly irregular, sketchy border, centered on the page. The word "Повторение" is written inside this frame in a bold, black, sans-serif font.

**Повторение**



**Из чего состоит программа?**

**С помощью какой библиотеки  
реализуется ввод и вывод?**



**Что такое переменная?**

# **Математические операции**

## **часть 2**



# Особенности операций

## Сокращенное присваивание

```
int q = 5;  
var = var + q;  
var+=q; // сокращенное сложение  
  
var = var - q;  
var-=q; // сокращенное вычитание  
var = var / q;  
var/=q; // сокращенное деление  
  
var = var * q;  
var*=q; // сокращенное умножение
```

Исключаем дублирование кода

# Особенности операций

## Сокращенное присваивание

```
int sum, a, b, c, d;  
sum = 0;  
a = 5;  
sum = sum + a; // 5  
b = 4;  
sum = sum + b; // 4 + 5 = 9  
c = 8;  
  
sum += c; // 9 + 8 = 17  
d = 8;  
sum += d; // 17 + 8 = 25
```

Исключаем дублирование кода

**Кто знает библиотеку, которая расширяет  
возможности использования математики?**

# Библиотека cmath

```
#include <iostream>
// библиотека ввода вывода
#include <cmath>
// использование математических операций
#include <iomanip>
// для фиксированного вывода
```

**#include <cmath>** // математические операции

+

**#include <iomanip>** // для ограничения кол-ва знаков при выводе

# Библиотека cmath

```
#include <iostream>
// библиотека ввода вывода
#include <cmath>
// использование математических операций
#include <iomanip>
// для фиксированного вывода
```

**#include <cmath>** // математические операции

+

**#include <iomanip>** // для ограничения кол-ва знаков при выводе

Для более сложных математических  
действий

# Библиотека cmath

## Округления

<b>round</b>	Округляет число по правилам арифметики, то есть $\text{round}(1.5) == 2$ , $\text{round}(-1.5) == -2$
<b>floor</b>	Округляет число вниз ("пол"), при этом $\text{floor}(1.5) == 1$ , $\text{floor}(-1.5) == -2$
<b>ceil</b>	Округляет число вверх ("потолок"), при этом $\text{ceil}(1.5) == 2$ , $\text{ceil}(-1.5) == -1$
<b>fabs</b>	Модуль (абсолютная величина)

## Корни и степени

<b>sqrt</b>	Квадратный корень. Использование: $\text{sqrt}(x)$
<b>cbrt</b>	Кубический корень. Использование: $\text{cbrt}(x)$
<b>pow</b>	Возведение в степень, возвращает $a^b$ . Использование: $\text{pow}(a,b)$

## Тригонометрия

<b>sin</b>	Использование: $\text{sin}(x)$
<b>cos</b>	Использование: $\text{cos}(x)$
<b>tan</b>	Использование: $\text{tan}(x)$

# cmath + iomanip

## Код программы

```
double a = 34.56;  
a = fabs(a); //модуль числа  
  
double rez = sqrt(a); //квадратный  
double rez2 = cbrt(a); //кубический корень  
double rez3 = pow(a,8); //возведение в степень  
double rez4 = pow(a,(1.0/8)); // КОРНИ  
  
long double y = 1/3.0; //преобразование к вещ типу операнд  
cout <<fixed<<setprecision(6)<<y<<endl;  
//вывод с ограничением на кол-во цифр после запятой
```


**Вывод с точностью до 6 знаков после запятой**

# Задача

Пользователь вводит в консоль число (x);

**Задача:** Посчитайте значение выражения  $(x^2 + \sin(x) - \sqrt{x}) / x^{-2}$  и выведите значение **с точностью до 5 знаков после запятой**.





**Что такое логическое  
выражение?**

# Логические выражения

**Логическое выражение** – высказывание, о котором можно сказать **ИСТИННО** оно или **ЛОЖНО**.

**Пример:**

- Автобус тяжелее человека – истина;
- Йогурт состоит из молока – истина;
- У человека есть жабры – ложь;

**Приведите свой пример**

# **Операции отношения**

# Операции отношения

```
a > b  
a < b;  
a >= b;  
a <= b;  
a == b;  
a != b;
```

Где a, b – целые числа, которые вводит пользователь

**Результат:**

истина (1) или ложь (0);

# Операции отношения

```
a > b  
a < b;  
a >= b;  
a <= b;  
a == b;  
a != b;
```

Где a, b – целые числа, которые вводит пользователь

**Важно:**

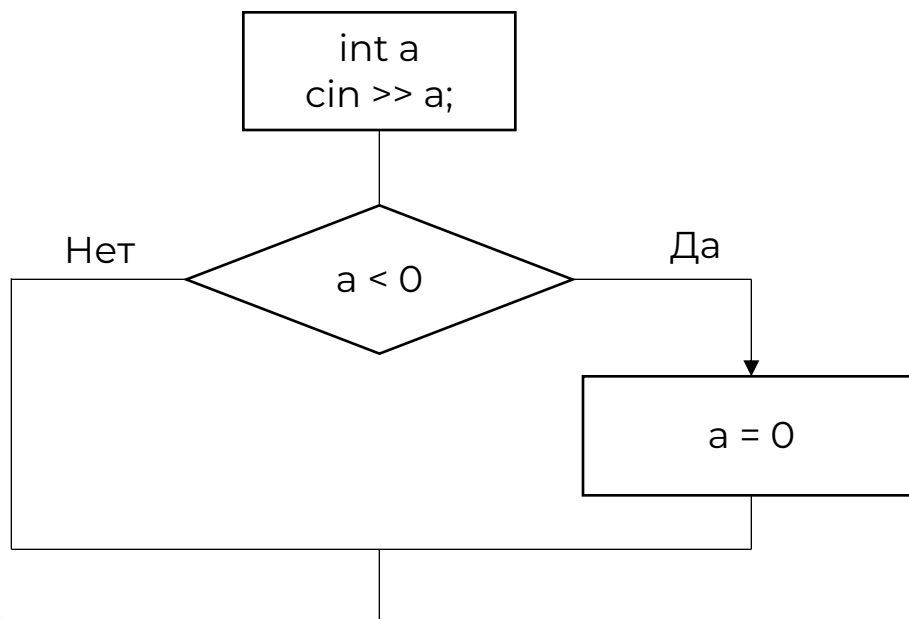
сравнение двух чисел это == (два знака равенства);

# **Условный оператор if**

# Условный оператор if

**if** – оператор, позволяющий реализовывать конструкции:

**ЕСЛИ** (условие сравнения), **ТО** (операция)





# Условный оператор if

Формат:

```
if (условие){  
    действие  
}
```

# Условный оператор if

Пример:

```
if (42 > 40){  
    cout <<"Больше"<<endl;  
}
```

# Условный оператор if

Пример:

```
int a;  
cin >> a;  
  
if (a == 42){  
    cout <<"Введено 42"<<endl;  
}
```

# Условный оператор if

Пример:

```
int a;  
cin >> a;  
  
if (a > 42){  
    cout <<"Введено число больше 42"<<endl;  
}  
  
if (a < 42){  
    cout <<"Введено число меньше 42"<<endl;  
}
```

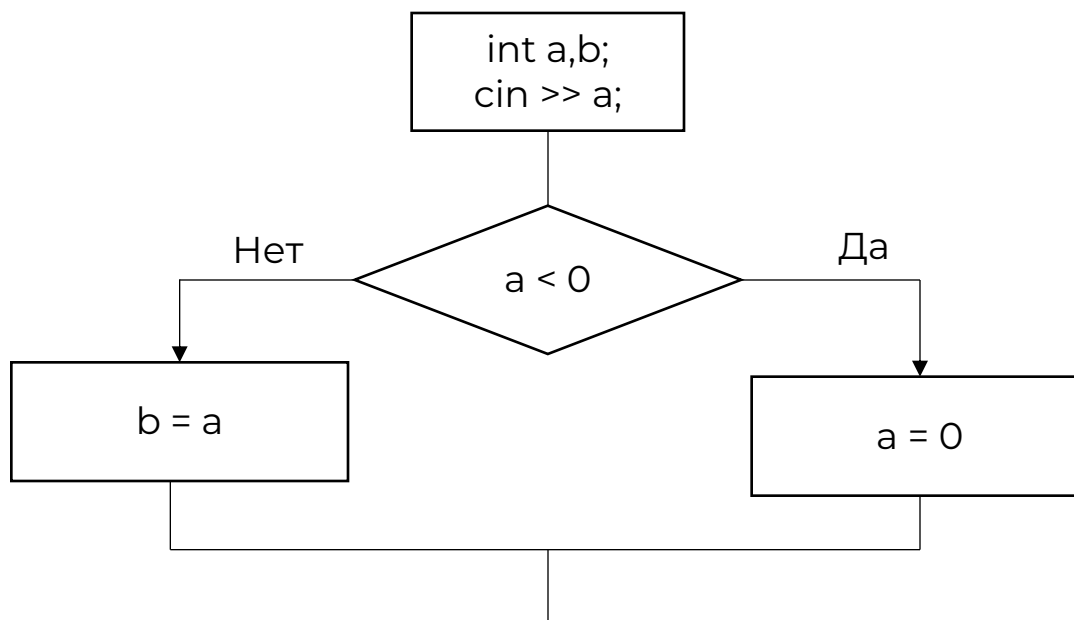
# Задача

Пользователь вводит в консоль число;

**Задача:** проверить, что введенное число кратно 5;

# Условный оператор if + else

**else** – дополняет if ситуацией **иначе**:



# Условный оператор if + else

Пример:

```
if (a > 42){  
    cout <<"Введено число больше 42"<<endl;  
}  
else {  
    cout <<"Введено число меньше 42"<<endl;  
}
```

**Исключаем дублирование кода**

# Задача

Пользователь вводит в консоль число;

**Задача:** проверить, что введенное число четное (вывести “Да”), если нет, то вывести (“Нет”);



# **Логические операции**

# Логические операции

Мы не общаемся только простыми предложениями, но обычно объединяем их в составные.

Так же и в программировании.

Простые высказывания объединяются в сложные путем логических операторов

# Логические операции

**Виды:**

**and or not**

**and** – оператор конъюнкции ( **&&** );

**or** – оператор дизъюнкции ( **||** );

**not** – оператор инверсии (отрицание) ( **!** ).

**Применение:** диапазоны значений

# Логические операции

Конъюнкция (\*)  
Логическое **И**

0	0	<b>0</b>
0	1	<b>0</b>
1	0	<b>0</b>
1	1	<b>1</b>

Дизъюнкция (+)  
Логическое **ИЛИ**

0	0	<b>0</b>
0	1	<b>1</b>
1	0	<b>1</b>
1	1	<b>1</b>

Инверсия  
Логическое **НЕ**

0	<b>1</b>
1	<b>0</b>

# Интервалы и полуинтервалы

```
int val;  
cin >> val;  
// [2, 8] диапазон  
if (val >= 2 && val <= 8){  
    cout << "Число принадлежит промежутку";  
}  
else {  
    cout << "Не принадлежит";  
}
```

# Задача

Пользователь вводит в консоль число;

**Задача:** проверить, что введенное число входит в данные промежутки, интервалы и полуинтервалы  $(-4; 9) \cup \{18\} \cup (65; 90]$ ;

# **Булевы переменные**

# Булевы переменные

- Принимают 2 значения – **True (1)** или **False (0)**;

**Применение:** хранение значения логических выражений.



# Булевы переменные

Вспомним данный пример:

```
int val;  
cin >> val;  
// [2, 8] диапазон  
if (val >= 2 && val <= 8){  
    cout << "Число принадлежит промежутку";  
}  
else {  
    cout << "Не принадлежит";  
}
```

**Где можно использовать  
булеву переменную?**

# Булевы переменные

Реализация через булеву переменную:

```
int val;  
cin >> val;  
bool flag = (val >= 2 && val <= 8);  
if (flag) cout << "Число принадлежит промежутку";  
//то же самое что и if (flag != 0)  
else cout << "Не принадлежит";  
}
```

# Задача

Пользователь вводит в консоль число;

**Задача:** проверить, что введенное число входит в данные промежутки, интервалы и полуинтервалы  $(-4; 9) \cup \{18\} \cup (65; 90]$ ; (**через булеву переменную**)

# **Тернарный оператор**

# тернарный оператор

Формат:

(условие) ? если истина : если ложь;

```
int a;  
cin >> a;  
int rez = (a % 2 == 0)? a: a*(-1);
```

**Применение:** краткая форма записи if else

# тернарный оператор

Для трех исходов:

```
(a > 0)? cout <<"Больше":(a == 0)? cout <<"Равно 0": cout <<"Меньше";
```

**На выводе:** больше, меньше или равно 0