Основы языка C++

СТРУКТУРА КУРСА

- 1 – Алгоритмы и структуры данных. C++
- 2 – Объектно-ориентированное программирование. C++

HELLO, WORLD

```
#include <iostream>
using namespace std;

int main ( )
{
    cout <<"Hello, world! “ <<endl;

    return 0;
}
```

Подключение стандартной библиотеки ввода-вывода

Стандартное пространство имен

С фунции **main** начинается выполнение программы

Печать строки

Возвращаем целое число (обычно код ошибки)

HELLO, WORLD

```
#include <iostream>
using std::cout;
using std::endl;

int main ( )
{
    cout <<"Hello, world! “ <<endl;

    return 0;
}
```

Подключение стандартной библиотеки ввода-вывода

Стандартное пространство имен

С фунции **main** начинается выполнение программы

Печать строки

Возвращаем целое число (обычно код ошибки)

HELLO, WORLD

```
#include <iostream>


int main ( )
{
    std::cout <<"Hello, world! “ <<std::endl;

    return 0;
}
```

Подключение стандартной библиотеки ввода-вывода

С фунции **main** начинается выполнение программы

Стандартное пространство имен

Печать строки

Возвращаем целое число (обычно код ошибки)

КОММЕНТАРИИ

```
#include <iostream>
// This is a single-line comment.
using namespace std;

int main ( ) // Any line can be commented.
{
    cout <<"Hello, world! “
                        <<endl;
    /* This is a multi-line
               comment. */
    return /* You can put it literally anywhere. */ 0;
}
```

Однострочный комментарий

Многострочный комментарий

ФОРМАТИРОВАНИЕ

```
#include <iostream>
int main()
{
    using namespace std;
    cout << "Come up and C++ me some time.";
    cout << endl;
    cout << "You won't regret it!" << endl;
    return 0;
}
```

```
int ma in()
re
turn 0;
cout << "Behold the Beans
of Beauty!";
```

```
#include <iostream>
       int
main
() {   using
   namespace
            std; cout
                 <<
"Come up and C++ me some time."
;  cout <<
endl; cout <<
"You won't regret it!" <<
endl;return 0; }
```

ПЕРЕМЕННЫЕ

- Переменная - именованная область памяти, в которой хранится значение
- Имена переменных (идентификаторы)
  - ✓ a-z A-Z 0-9 _ (не может начинаться с цифры)
  - X Пробельные символы
- Code style
  - ✓ massiveOfInteger
  - ✓ i_outer
  - ✓ number_of_cows_on_the_field
  - X jhg95
  - X _KJY98jhg
  - X Vasya_007

```
int a_lonely_integer;
a_lonely_integer = 5;
```

```
int a_lonely_integer = 5;
```

ТИПЫ ДАННЫХ

- целочисленные
  - bool (true/false)
  - char (1 байт)
  - short
  - int
  - long
- дробные
  - float
  - double

```
int a_lonely_integer;
a_lonely_integer = 5;
```

```
int a_lonely_integer = 5;
```

```
#include <climits>
```

```
cout << "int is " << sizeof (int) << " bytes.\n";
cout << "short is " << sizeof n_short << " bytes.\n";
```

ТИПЫ ДАННЫХ

- целочисленные
  - char (1 байт)
  - short
  - int
  - long
- модификаторы
  - signed
  - unsigned
  - const

```
unsigned int a_lonely_unsigned_int = 10;
unsigned the_same_lonely_unsigned_int = 0xA;
unsigned char a_lonely_uchar = 255;
unsigned long long int gargantuan = ULLONG_MAX;
const int AN_IRREPLACEABLE_INT = 100;
const int AN_UNDECIDED_INT;
AN_UNDECIDED_INT = 100; // Too late!
```

```
#include <climits>
```

АРИФМЕТИЧЕСКИЕ ОПЕРАТОРЫ

- = присвоение
- == сравнение
- + сложение
- - вычитание
- * умножение
- / деление

- % деление по модулю (взятие остатка)
  9%5 даст 4

```
int a = 3, b = 4;
cout <<“The sum of 3 and 4 is ”<< a + b <<endl;
int c = a * b, d = (a + b) / b + a;
a = b = c = d = 0;
```

```
fahr = fahr + step;
fahr += step;

fahr = fahr + 1;
fahr += 1;
fahr ++;
++ fahr;
```

ПРИОРИТЕТ ОПЕРАТОРОВ

Приоритет	Операция	Описание	Ассоциативность
1	::	Область видимости	Слева направо
2	() [] . –>	Вызов функции Обращение к массиву по индексу Выбор элемента по ссылке Выбор элемента по указателю	
3	++a --a + – ! ~ (*type*) * & sizeof new, new[] delete, delete[]	Прединкремент и преддекремент Унарный плюс и минус Логическое НЕ и побитовое НЕ Приведение к типу type Indirection (разыменование) Адрес Размер Динамическое выделение памяти Динамическое освобождение памяти	Справа налево
4	.* ->*	Указатель на член	Слева направо
5	* / %	Умножение, деление и остаток	
6	+ –	Сложение и вычитание	
7	<< >>	Побитовый сдвиг влево и вправо	
8	< <= > >=	Операции сравнения < и ≤ Операции сравнения > и ≥	
9	== !=	Операции сравнения = и ≠	
10	&	Побитовое И	
11	^	Побитовый XOR (исключающее ИЛИ)	
12	|	Побитовое ИЛИ (inclusive or)	
13	&&	Логическое И	
14	||	Логическое ИЛИ	
15	?: = += –= *= /= %= <<= >>= &= ^= |=	Тернарное условие Прямое присваивание (предоставляемое по умолчанию для C++ классов) Присвоение с суммированием и разностью Присвоение с умножением, делением и остатком от деления Присвоение с побитовым сдвигом слево и вправо Присвоение с побитовыми логическими операциями (И, XOR, ИЛИ)	Справа налево
16	throw	Операция выброса исключения	
17	a++ a--	Постинкремент и постдекремент	Слева направо

ПРИОРИТЕТ ОПЕРАТОРОВ

Приоритет	Операция	Описание	Ассоциативность
1	::	Область видимости	Слева направо
2	() []	Вызов функции Обращение к массиву по индексу	
3	++a --a	Прединкремент и преддекремент	Справа налево
	+ − ! ~ (*type*)	Унарный плюс и минус Логическое НЕ и побитовое НЕ Приведение к типу type	
	sizeof	Размер	
5	* / %	Умножение, деление и остаток	Слева направо
6	+ −	Сложение и вычитание	
8	< <= > >=	Операции сравнения < и ≤ Операции сравнения > и ≥	
9	== !=	Операции сравнения = и ≠	
13	&&	Логическое И	
14	||	Логическое ИЛИ	
15	?:	Тернарное условие	Справа налево
	=	Прямое присваивание	
	+= −=	Присвоение с суммированием и разностью	
	*= /= %=	Присвоение с умножением, делением и остатком от деления	
17	a++ a--	Постинкремент и постдекремент	Слева направо

ПРИОРИТЕТ ОПЕРАТОРОВ

ПРЕФИКСНЫЙ И ПОСТФИКСНЫЙ ИНКРЕМЕНТ

```
int x = 5;
int y = ++x;                    // изменить x, затем присвоить его y
                                // y равно 6, x равно 6
int z = 5;
int y = z++;                    // присвоить y, затем изменить z
                                // y равно 5, z равно 6
```

ЦИКЛ FOR

```
for (/*инициализация*/ ;
        /*условие продолжения цикла*/ ;
                /*итерация*/)
   {
     /*тело цикла*/
   }

for (i = 0; i < N; i++)  //среднестатистический цикл for
{
   a += b + c;
   b *= i + c;
}
```

ЦИКЛ FOR

ПРИМЕР

```
#include <iostream>
using namespace std;
int main ( )
{
    int up = 300, low = 0, step = 20;
    for (int fahr = low; fahr <= up ; fahr += step)
    {
        cout << fahr <<“ to ” << 5 * (fahr – 32) / 9);
    }
    return 0;
}
```

ЦИКЛ FOR

```
for (;;)   //          можно ничего не писать, но
                       //          ; ставить обязательно
{
   i++;
   a += i;
}

for (; i < 10; i++)
   a *= i;  //если оператор один, { } можно не ставить

for (; i < 10; i++, a *= i);

for (; i < 10; i++);
   a *= i;
```

```
for (x = 20; x > 5; x--)                    // продолжать, пока x больше чем 5
for (x = 1; y != x; ++x)                    // продолжать, пока y не равно x
for (cin >> x; x == 0; cin >> x))           // продолжать, пока x равно 0
```

ЦИКЛ WHILE

```
#include <iostream>
using namespace std;
int main ( )
{
    int fahr = low, up = 300, low = 0, step = 20;
    while (fahr <= up) {
       int cels = 5 * (fahr – 32) / 9;
       cout << fahr <<“ to ” << cels;
       fahr = fahr + step;
    }
   return 0;
}
```

ЦИКЛ DO-WHILE

```
#include <iostream>
int main()
{
    using namespace std;
    int n;
    cout << "Enter numbers in the range 1-10 to find ";
    cout << "my favorite number\n"; // запрос на ввод любимого числа из диапазона 1-10
    do
    {
        cin >> n;                          // выполнить тело
    } while (n != 7);                      // затем проверить
    cout << "Yes, 7 is my favorite.\n" ;  // любимое число - 7
    return 0;
}
```

ОПЕРАТОР IF-ELSE

- if (выражение)
  инструкция_1 ;
  else
  инструкция_2 ;

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

- проверяется выражение
- если оно истинно (не равно 0), то выполняется инструкция_1
- иначе выполняется инструкция_2
- часть **else** можно опустить

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

ВЛОЖЕННЫЕ IF-ELSE

```
if (character == CYLON)
{
   execute(character);
}
else
{
   if (character == SLEEPER_AGENT)
         execute_secretly(character);
   else
         do_not_execute(character);
}
```

ВЛОЖЕННЫЕ IF-ELSE

```
if (character == CYLON)
    execute(character);
else
    if (character == SLEEPER_AGENT)
            execute_secretly(character);
    else
            do_not_execute(character);
```

ВЛОЖЕННЫЕ IF-ELSE

```
if (character == CYLON)
    execute(character);
else if (character == SLEEPER_AGENT)
    execute_secretly(character);
else
    do_not_execute(character);
```

ВЛОЖЕННЫЕ IF-ELSE

```
if (!(character == CYLON))
    if (character == SLEEPER_AGENT)
            execute_secretly(character);
    else
            do_not_execute(character);
else
    execute(character);
```

ВЛОЖЕННЫЕ IF-ELSE

```
if (character != CYLON)
    if (character == SLEEPER_AGENT)
            execute_secretly(character);
else
    execute(character);
```

СОКРАЩЕННЫЙ IF-ELSE

```
max = x > y ? x : y;
abs = x >= 0 ? x : -x;

a = x > y || x * y <= z && ! x ? func(x) :
  func(y)%7;

cout << ( (i < 2) ? ! i ? x : y : x );
```

ОПЕРАТОРЫ СРАВНЕНИЯ

- == равно
- != не равно
- <
- >
- <=
- >=

- пробелы внутри оператора не ставятся
- = это оператор "присвоить"!
  Не путайте с оператором сравнения!

ПРИОРИТЕТ ОПЕРАТОРОВ

ПРИСВОЕНИЕ И СРАВНЕНИЕ

Приоритет	Операция	Описание	Ассоциативность
1	::	Область видимости	Слева направо
2	() []	Вызов функции Обращение к массиву по индексу	
3	++a --a + − ! ~ (*type*) sizeof	Прединкремент и преддекремент Унарный плюс и минус Логическое НЕ и побитовое НЕ Приведение к типу type Размер	Справа налево
5	* / %	Умножение, деление и остаток	Слева направо
6	+ −	Сложение и вычитание	
8	< <= > >=	Операции сравнения < и ≤ Операции сравнения > и ≥	
9	== !=	Операции сравнения = и ≠	
13	&&	Логическое И	
14	||	Логическое ИЛИ	
15	?:	Тернарное условие	Справа налево
	=	Прямое присваивание	
	+= −=	Присвоение с суммированием и разностью	
	*= /= %=	Присвоение с умножением, делением и остатком от деления	
17	a++ a--	Постинкремент и постдекремент	Слева направо

ПРИСВОЕНИЕ И СРАВНЕНИЕ

```
bool check = false;
if (check = true)
    std::destroy_all_humans();
else
    live_long_and_prosper();
```

ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

- && логическое И
- || логическое ИЛИ
- ! отрицание

&&	false	true
false	false	false
true	false	**true**

||	false	true
false	false	**true**
true	**true**	**true**

```
if ( c == 'a' || c == 'A' )
    na ++;
```

```
for (; c == 'a' || c == 'A'; )
    na ++;
```

ПРИОРИТЕТ ОПЕРАТОРОВ

Приоритет	Операция	Описание	Ассоциативность
1	::	Область видимости	Слева направо
2	() []	Вызов функции Обращение к массиву по индексу	
3	++a --a + − ! ~ (*type*) sizeof	Прединкремент и преддекремент Унарный плюс и минус Логическое НЕ и побитовое НЕ Приведение к типу type Размер	Справа налево
5	* / %	Умножение, деление и остаток	Слева направо
6	+ −	Сложение и вычитание	
8	< <= > >=	Операции сравнения < и ≤ Операции сравнения > и ≥	
9	== !=	Операции сравнения = и ≠	
13	&&	Логическое И	
14	||	Логическое ИЛИ	
15	?: = += −= *= /= %=	Тернарное условие Прямое присваивание Присвоение с суммированием и разностью Присвоение с умножением, делением и остатком от деления	Справа налево
17	a++ a--	Постинкремент и постдекремент	Слева направо

ОПЕРАТОР SWITCH

```
#include <iostream>
using namespace std;

int main () {
    char c;
    cin >>c; // оператор ввода
    switch (c) {  // можно рассмотреть различные варианты значения переменной
        case 48:
            cout <<"zero\n";
            break; // если не ставить break, исполнение продолжится
        case '1':
            cout <<“one“ <<endl;
        case 50:
            cout <<“a lot“ <<endl;
            break;
        default:
            cout <<“whatever“ <<endl;
    }
    return 0;
}
```

К СЛОВУ О BREAK

```
for (;;)
{
  i++;
  if (i > 10000) break;
}

for (i = 0; i < N; i++)
{
  if (i % 2) continue;
  make_something_of(i);
}
```

~~goto~~

ОТЛАДКА ПРОГРАММЫ

- Ошибки
  - синтаксические *syntax errors*
    - забыли ; в конце выражения
  - выполнения *runtime errors*
    - деление на 0
  - семантические *semantic errors*
    - (работает, но делает не то, что нужно)
  - плавающие ошибки *heisenbug etc.*
- Отладочная печать
- Дебагер

https://habr.com/ru/post/104952/