

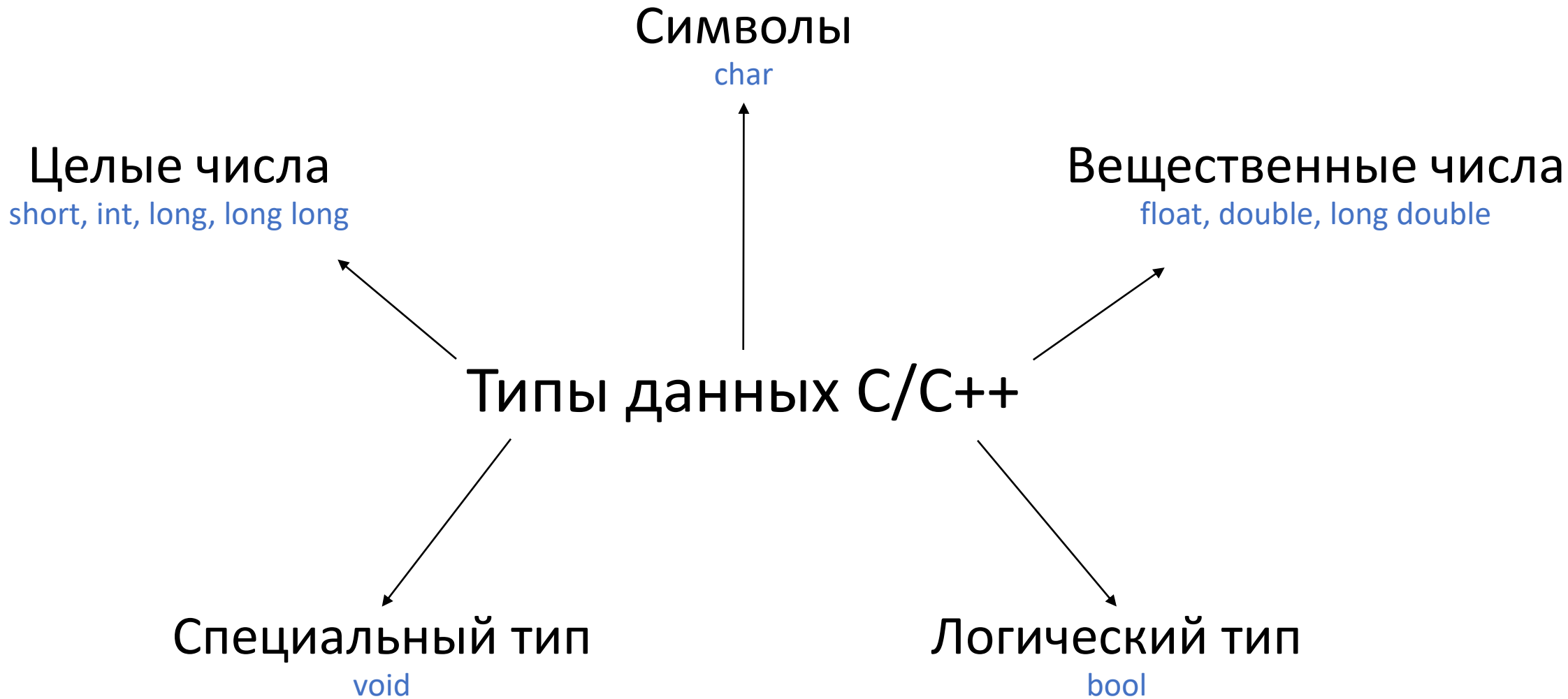
Типы данных и преобразование типов.

В рамках модуля «Язык программирования C/C++»

Мухаметов Данил Илгизович

seemsclever@mail.ru

2025



Целочисленные типы данных

Целочисленные типы предназначены для хранения целых чисел без дробной части. В C и C++ используются несколько разновидностей:

short – короткое целое число;

int – целое число стандартного размера;

long – длинное целое число;

long long – целое число расширенного диапазона.

Тип определяет размер в байтах и диапазон значений.

Тип данных	Диапазон значений	Занимаемая память
Short	от -32768 до 32767	2 байта
Int	от -2 147 483 648 до 2 147 483 647	4 байта
Long	от -2 147 483 648 до 2 147 483 647 или от -9 квинтиллионов до +9 квинтиллионов	4 или 8 байт
Long long	от -9 квинтиллионов до +9 квинтиллионов или более	8 байт или более

Символьный тип данных (char)

Символьный тип `char` используется для хранения отдельных символов.

Каждая переменная этого типа занимает 1 байт памяти и хранит числовой код символа.

Значения интерпретируются согласно кодировке (например, ASCII)

Тип данных	Диапазон значений	Занимаемая память
Char	от -128 до 127	1 байт

```
char c = 'A';  
int code = c; // 65
```

Типы с плавающей точкой (вещественные)

Типы с плавающей точкой предназначены для хранения дробных чисел. В языке C и C++ используются три основных типа:

`float` – число одинарной точности;

`double` – число двойной точности;

`long double` – число повышенной точности.

Различие между ними – в точности и количестве занимаемой памяти.

Тип данных	Точность	Занимаемая память
Float	7 знаков после запятой	4 байта
Double	15 знаков после запятой	8 байт
Long double	18-33 знака после запятой	10-16 байт

Логический тип данных (bool)

Логический тип `bool` предназначен для хранения значений истина или ложь. Он принимает только два возможных значения:

`true` – истина (обычно представляется числом 1);

`false` – ложь (обычно представляется числом 0).

Тип часто используется в условиях, циклах и логических выражениях.

Тип данных	Диапазон значений	Занимаемая память
Bool	true(1) или false(0)	1 байта

Специальный тип (void)

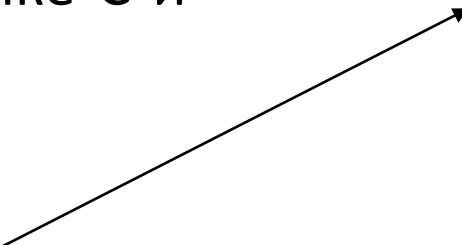
Тип void используется в языке C и C++ в особых случаях:

функция не возвращает значения;

указатель на неопределённый тип данных;

как параметр функции.


Модификатор void указывает на отсутствие значения или отсутствие конкретного типа.



```
void functionName(){  
    // Код функции  
}
```



```
void* p;
```



```
void functionName(void){  
    // Код функции  
}
```

Модификаторы signed и unsigned

В языке C и C++ для целочисленных типов можно указывать модификаторы:

`signed` — число может быть как положительным, так и отрицательным;

`unsigned` — число только неотрицательное, диапазон значений смещён в сторону больших положительных чисел.

Модификаторы применяются к `char`, `short`, `int`, `long`.

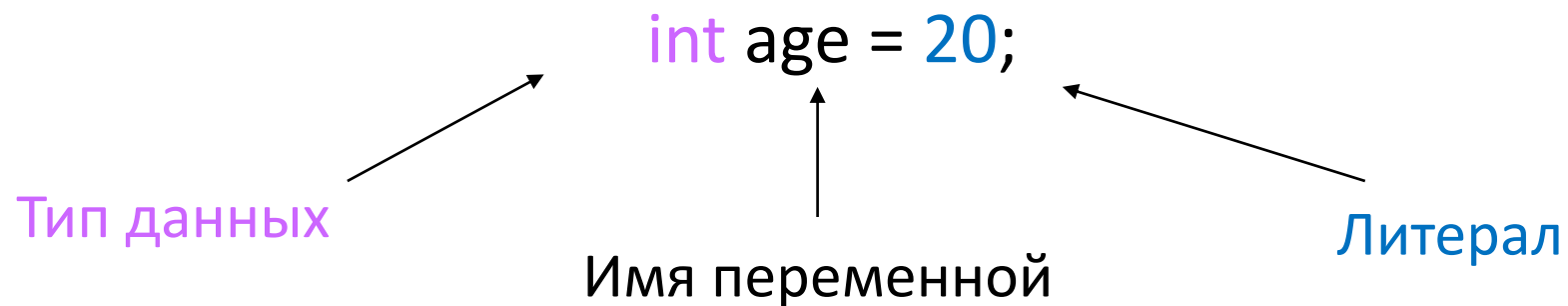
Тип данных	Диапазон значений	Занимаемая память
Unsigned char	от 0 до 255	1 байт
Unsigned short	от 0 до 65535	2 байта
Unsigned int	от 0 до 4294967295	4 байта
Unsigned long	от 0 до 4294967295 или более	4 байта или более

Типы данных

Тип данных	Занимаемая память	Диапазон значений (signed)	Диапазон значений (unsigned)
Char	1 байт	от -128 до 127	от 0 до 255
Short	2 байта	от -32 768 до 32 767	от 0 до 65 535
Int	4 байта	от -2 147 483 648 до 2 147 483 647	от 0 до 4 294 967 295
Long	4 или 8 байт	от -2 147 483 648 до 2 147 483 647 или от -9 квинтиллионов до +9 квинтиллионов	от 0 до 4 294 967 295 или от 0 до 18 квинтиллионов
Long long	8 байт или более	от -9 квинтиллионов до +9 квинтиллионов или более	от 0 до 18 квинтиллионов или более
Float	4 байта	$\sim \pm 3.4\text{E-}38 \dots \pm 3.4\text{E+}38$	-
Double	8 байт	$\sim \pm 1.7\text{E-}308 \dots \pm 1.7\text{E+}308$	-
Bool	1 байт	true или false	-
Void	-	Не имеет значений	-

Что такое переменная?

Переменная – это именованная область памяти, предназначенная для хранения данных определённого типа.



Что такое литерал?

Литерал – это фиксированное значение, которое явно записывается в коде программы. Литералы бывают:

числовые (например: 10, 3.14);

символьные (например: 'a');

строковые (например: "Hello");

логические (true, false).

Литералы используются для присваивания значения переменным.

```
int phoneCode = 52;
```

```
// литерал - 52
```

```
char c = 'b';
```

```
// литерал - 'b'
```

```
string greeting = 'Hello, world';
```

```
// литерал - 'Hello, world'
```

```
bool isAllowed = false;
```

```
// литерал - false
```

Инициализация переменной

Инициализация – это присвоение начального значения переменной в момент её объявления. Основные формы инициализации в C/C++:

Присваивание

```
int phoneCode = 52;
```

Конструкторная форма

```
int phoneCode(52);
```

Списковая
инициализация

```
int phoneCode{52};
```

Если переменная объявлена, но не инициализирована, её значение может быть рандомным.

Динамическая инициализация

Динамическая инициализация – это присвоение переменной значения, вычисленного во время выполнения программы.

```
int x = 6;
```

```
int y = x + 10;  // значение зависит от x
```

```
double z = sqrt(y);  // использование функции
```

Такой подход позволяет задавать начальные значения на основе выражений или вычислений.

Создание констант

Константа – это переменная, значение которой нельзя изменить после инициализации. Для объявления используется ключевое слово `const`.

```
const int MAX_USERS = 100;
```

```
const double PI = 3.14159;
```

```
MAX_USERS = 200; // ошибка!
```

Неявное преобразование типов

Неявное преобразование выполняется автоматически, когда значения разных типов участвуют в выражении. Основные правила:

Меньший тип преобразуется к большему (int -> double);

При арифметике разные типы приводятся к общему:

```
int a = 5;
```

```
double b = a + 0.5; // a будет преобразован в double
```

bool преобразуется к числовому типу и наоборот:

```
bool isAllowed = true;
```

```
int a = isAllowed; // a будет присвоено значение 1
```

Потери данных при неявном приведении типов

Неявное преобразование типов может приводить к потере данных.

Например:

Преобразование double -> int
отбрасывает дробную часть;

Преобразование большого int -> short
может привести к переполнению;

Преобразование отрицательного
числа в unsigned даст большое
положительное значение.

```
double d = 3.9;
```

```
int x = d;    // x = 3, дробная часть  
потеряна
```

```
int a = 70000;
```

```
short s = a;    // переполнение,  
значение непредсказуемо
```

```
int b = -1;
```

```
unsigned int u = b; // u = большое  
положительное число
```


Переполнение целочисленных типов

Переполнение – это ситуация, когда значение выходит за пределы допустимого диапазона типа.

Для unsigned типов значение “оборачивается” по модулю диапазона

```
unsigned char x = 255;  
x = x + 1; // результат: 0
```

Для signed типов результат непредсказуем (зависит от реализации)

```
signed char y = 127;  
y = y + 1; // результат зависит от  
компилятора
```

Явное преобразование типов (cast)

Явное преобразование типов выполняется программистом, чтобы преобразовать значение из одного типа в другой.

Основные формы:

C-стиль: (type) value

C++-стиль: static_cast<type>(value)

```
double d = 3.14;
```

```
int x = (int)d;    // C-стиль
```

```
int y = static_cast<int>(d);    // C++-стиль
```

Спасибо за внимание!