

# Семинар #1: Введение в язык C++ (для тех, кто знает C). Классные задачи.

## Hello world++

Пишем первую программу на C++

```
#include <cstdio>
int main() {
    printf("Hello World++\n");
}
```

Все библиотеки из языка C можно использовать и в языке C++. Только название библиотеки без `.h` на конце и `c` символом `c` в начале:

```
<stdio.h> -----> <cstdio>
```

Для компиляции используйте компилятор `g++`. Вот так:

```
g++ helloworld.cpp
./a.out
```

Или, если вы хотите установить у исполняемого файла своё имя за место `a.out`:

```
g++ -o hello helloworld.cpp
./hello
```

- **Задача 1:** Скомпилируйте и запустите простейшую программу `00helloworld.cpp`.
- **Задача 2:** Разберитесь в программе `01structnaming.cpp`. Скомпилируйте и запустите.

## Тип bool

В прошлом семестре, для хранения результатов логических операций, мы использовали целочисленные типы. В языке C++ есть встроенный тип `bool`, который может принимать 2 значения (`true` и `false`).

```
#include <cstdio>
int main() {
    bool a = true;
    bool b = false;
    bool c = a || b;

    if (c)
        printf("Yes\n");
    else
        printf("No\n");
}
```

- **Задача 3:** Что напечатает эта программа? Скомпилируйте её и запустите (`02bool.cpp`)
- **Задача 4:** Напишите функцию, которая будет принимать на вход целое число и возвращать `true`, если число оканчивается на 0 и `false` иначе. Вызовите эту функцию из `main`.

## Пространство имён

```
#include <stdio>
// Определяем переменные, структуры, функции внутри пространства имён foo
namespace foo {
    int a = 5;
    int square(int x) {
        return x * x;
    }
}
// Чтобы получить доступ к ним вне пространства имён, нужно добавить к именам foo::
int main() {
    printf("%d\n", foo::a);
}
```

- **Задача 5:** Возведите `foo::a` в квадрат с помощью функции `foo::square`.
- **Задача 6:** Создайте своё пространство имён по имени `bar` и определите в нём переменную `a = 7`. Напечатайте значение этой переменной в `main`.
- **Задача 7:** Сделайте задание в программе `03namespace.cpp`.

## Ссылки

Ссылка – это переменная, которая является новым именем для существующего участка памяти.

```
#include <stdio>

int main() {
    int a = 10;
    // Создадим ссылку r на переменную a
    int& r = a;
    // Теперь, если изменить r, то поменяется и a
    r += 5;
    printf("%d\n", a);
}
```

Ссылки часто используются для тех же целей, что и указатели (только со ссылкам работать удобнее). В отличии от указателей, ссылки:

1. Должны всегда инициализироваться при создании
2. Не могут никуда не ссылаться (т.е. не могут равняться `NULL`)
3. Их нельзя переприсвоить. При использовании оператора `=` со ссылками изменяется та переменная, на которую ссылка ссылается, а не сама ссылка.

- **Задача 8:** Сделайте задание в файлах `04ref.cpp`, `05ref.cpp` и `06ref.cpp`.

## Перегрузка функций

- **Задача 9:** Сделайте задание в файлах `07function_overload.cpp`, `08nulptr.cpp` и `09default_arguments.cpp`.

## Перегрузка операторов

- **Задача 10:** Сделайте задание в файлах `10operator_overload.cpp`, `11iostream.cpp` и `12iostream_overload.cpp`.