

# Язык C++

---

Прикладное программирование

# Структура программы

---

```
(Глобальная область)  
  
int main ()  
{  
    return 0; |  
}
```



# Структура программы

---

- Библиотека `<conio.h>` - консольный ввод-вывод, для того чтобы программа задержалась на экране,
- :
- 1) Перед функцией `int main` написать:
  - `#include <conio.h>`

# Структура программы

---

- Файл `<conio.h>` определяет функцию `_getch()`.
- Перед `return 0;` написать функцию `_getch();` - которая задерживает нашу программу, чтобы мы на нее посмотрели.



# Структура программы

---

(Глобальная область)

```
#include <conio.h>
```

```
int main ()
```

```
{
```

```
    getch(); |
```

```
    return 0;
```

```
}
```

# Структура программы

---

- Для вывода текстов используем директиву стандартного потока ввода и вывода. Описание этих функций находится в файле `iostream`.



# Структура программы

---

- `cin >>` для ввода данных, `>>` - оператор перенаправления потока ввода данных
- `cout <<` для вывода данных на экран, `<<` - оператор перенаправления потока вывода данных

# Структура программы

---

## Using namespace std

- Пространство имен – это группа определенных функций, классов и т.п. Практически все стандартные библиотеки C++ используют пространство имен std, именно поэтому практически всегда нужно писать `using namespace std`, иначе вам придется каждый раз вызывать функцию из этих библиотек, подставляя префикс пространства имен `std::`.



# Структура программы

---

- И теперь после знака {на следующей строчке напишем: `cout << "Hello world!";`
- Часть `cout` обеспечивает вывод информации на консоль. Она принадлежит стандартному пакету. Символ `<<` является оператором перенаправления потока данных. В кавычках пишется строка, которую мы хотим вывести.

# Структура программы

---

```
(Глобальная область) main()
#include <iostream> //Пространство имен std объявляется в iostream
#include <conio.h>

using namespace std;
int main ()
{
    cout <<"Hello, world!";
    getch();
    return 0;
}
```

Выводит строку:  
Hallo, world!



# Структура программы

---

- Для того чтобы у нас появилась русская надпись, нам нужно сделать следующее:
- Подключаем заголовочный файл `locale` с помощью `#include`
- А перед `cout` напишем:

```
setlocale (LC_ALL, "Russian");
```

# Структура программы

---

(Глобальная область)

```
#include <iostream>
#include <conio.h>
#include <locale>

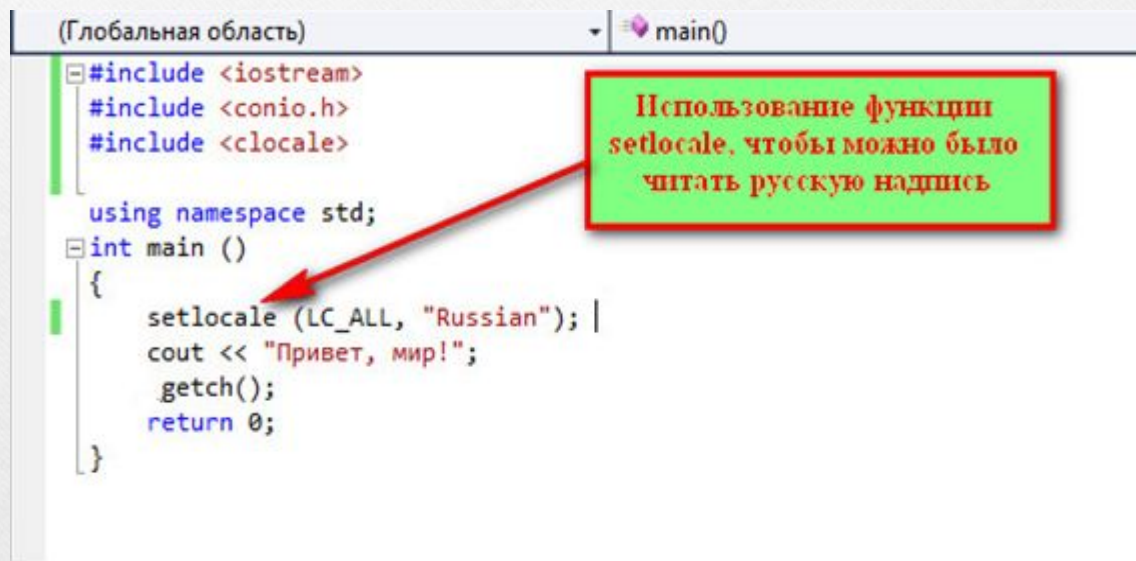
using namespace std;

int main ()
{
    cout << "Привет, мир!";
    getch();
    return 0;
}
```

Подключили  
заголовочный файл  
clocale



# Структура программы



```
(Глобальная область) main()
#include <iostream>
#include <conio.h>
#include <ctype.h>

using namespace std;

int main ()
{
    setlocale (LC_ALL, "Russian"); |
    cout << "Привет, мир!";
    getch();
    return 0;
}
```

Использование функции  
setlocale, чтобы можно было  
читать русскую надпись

# Типы данных в C++

---

- Тип – это множество допустимых значений величины.
- Все типы данных одномоментно не освоить, лучше изучать этот момент постепенно, поэтому для объяснения типов данных давайте решим, такую задачу. Допустим, нам надо написать программу, которая просто складывала бы два числа, к примеру, сложить числа *a* и *b*.



# Целые типы данных

---

<u>Сводная таблица знаковых целых типов данных</u>		
<u>Тип данных</u>	<u>Диапазон значений</u>	<u>Размер, байт</u>
char	-128...127	1
short	-32 768...32 767	2
int	-2 147 483 648 ... 2 147 483 647	4
long	-2 147 483 648 ... 2 147 483 647	4
long long	-9 223 372 036 854 775 808...9 223 372 036 854 775 807	8

# Целые типы данных

---

<u>Сводная таблица беззнаковых целых типов данных</u>		
<u>Тип данных</u>	<u>Диапазон значений</u>	<u>Размер, байт</u>
unsigned char	0...255	1
unsigned short	0 ... 65 535	2
unsigned int	0 ... 4 294 967 295	4
unsigned long	0 ... 4 294 967 295	4
unsigned long long	0 ... 18 446 744 073 709 551 615	8



# Целые типы данных

---

- Для решения задачи нам понадобятся две переменные `a` и `b`, которые мы будем складывать. Зададим им тип `int`. Теперь возникает вопрос, а в какой области программы их написать. И тут возможны 2 варианта.

# Целые типы данных

---

slozhenie.cpp X

(Глобальная область)

```
#include <iostream>
#include <conio.h>
#include <locale.h>

using namespace std;

int a; //глобальная переменная

int main ()
{
    setlocale (LC_ALL, "Russian");

    getch ();

    return 0;
}
```

Глобальная переменная



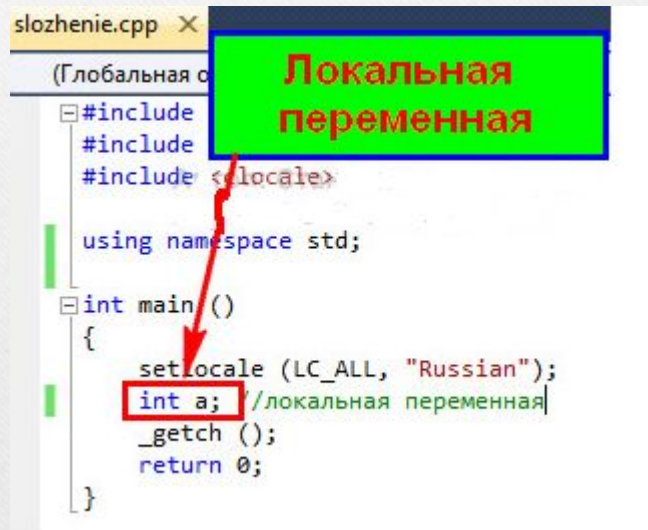
# Целые типы данных

---

- В данном случае эта переменная называется глобальной, описана вне блока и изначально обнуляется. Эта переменная живет и действует от начала выполнения программы до конца.

# Целые типы данных

---



The screenshot shows a code editor window titled 'slozhenie.cpp'. The code includes standard headers and uses the 'std' namespace. Inside the 'main' function, a red box highlights the declaration 'int a;', which is annotated with a green comment '/\*локальная переменная\*/'. A red arrow points from a green box labeled 'Локальная переменная' to this declaration.

```
slozhenie.cpp X
(Глобальная о
#include
#include
#include <locale>

using namespace std;

int main ()
{
    setlocale (LC_ALL, "Russian");
    int a; /*локальная переменная*/
    _getch ();
    return 0;
}
```

**Локальная переменная**



# Логический тип данных

---

- Язык C++ предоставляет тип, специально созданный для хранения логических значений True или False, оба из которых являются зарезервированными ключевыми словами C++. Этот тип полезен при хранении параметров и флагов, которые могут быть установлены или сброшены, существовать или отсутствовать, могут быть доступными или недоступными. Объявление переменной логического типа:  
  
`Bool AlwaysWord=false;`

# Типы с плавающей точкой

---

- Или вещественные типы. Эти числа могут быть положительными или отрицательными, могут содержать десятичные значения.
- К типам с плавающей точкой относятся типы `float` и `double`.
- Примеры описания типов:
- `Float Pi=3.14;`
- `Double My=22/7;`



# Условный оператор

---

- Условное выполнение кода в C++ на базе конструкции If... else....выглядит следующим образом:

if (условное выражение)

выполнить нечто, когда условное выражение возвращает true;

else //необязательная часть

выполнить нечто другое, когда условное выражение возвращает false.

# Применение условного оператора

```
polsee_iz_treh.cpp X
(Глобальная область)
#include <iostream>
#include <conio.h>
#include <locale>

using namespace std;

int main ()
{
    setlocale (LC_ALL, "Russian");
    1) cout << "Введите число a b и c для определения наибольшего из них\n";
    2) int a, b, c, max;
    3) cin >> a >> b >> c;
    if (a>b && a>c)
        max=a;
    else
        if (b>c)
            max=b;
        else
            max=c;
    4) cout << "Максимальное из трех - это число "<<max;
    getch ();
    return 0;
}
```

Строго по блок-схеме



# Условная обработка с использованием конструкции switch-case

---

- Задача конструкции switch-case в том, чтобы сравнить результат некоего выражения с набором возможных констант и выполнить разные действия, соответствующие каждой из этих констант.
- Ключевые слова C++, которые используются в такой конструкции, - это switch, case, default, break.

# Условная обработка с ИСПОЛЬЗОВАНИЕМ конструкции switch-case

- Конструкция switch-case имеет следующий синтаксис:

```
switch (выражение)
```

```
{
```

```
    case метка А:
```

```
        действие 1;
```

```
        break;
```

```
    case метка В:
```

```
        действие 2;
```

```
        break;
```

```
    // и так далее....
```

```
    default:
```

```
        Сделать Нечто Если Выражение Не Соответствует Ничему Выше;
```

```
        break;
```

```
}
```



# Операторы цикла

---

- Операторы цикла служат для организации повторяющегося процесса. В C++ можно использовать три вида циклов:
1. Цикл с предусловием (while)
  2. Цикл с постусловием (do while)
  3. Цикл с параметром (for)

# Оператор цикла с предусловием (while) в C++

---

- Структура цикла while в C++ такова:  
while (выражение) оператор;
- В условии можно использовать следующие операции:

> <      больше меньше

>= <=    больше или равно    меньше или равно

==      равно

!=      не равно



# Пример использования цикла с предусловием

---

```
int main ()
{
    setlocale (LC_ALL, "Russian");
    cout << "Введите натуральное число\n";
    int k=0, n;
    cin >> n;
    while (n!=0) //Пока n не равно 0 делай {...}
    {
        k++; //k увеличиваем на 1
        n=n/10; //n присваиваем целую часть частного, т.е. уменьшаем
    }
    cout << "Число цифр в этом числе " << k; //вывод на экран числа k
    getch();
    return 0;
}
```

# Оператор цикла с постусловием (do-while) в языке C++

---

- В языке C++ цикл с постусловием имеет следующий вид:

**do оператор while выражение;**

- Если для решения задачи вам необходимо использовать несколько операторов, то они также как и в любом цикле или группе выполнения последовательных операторов заключаются в фигурные скобки.



# Пример использования цикла с постусловием

(Глобальная область)

```
#include <iostream>
#include <conio.h>
#include <locale>

using namespace std;

int main ()
{
    setlocale (LC_ALL, "Russian");
    cout << "Введите два числа для нахождения НОД\n";
    int x, y, nod;
    cin >> x >> y;
    do
    {
        if (x>y) x=x % y;
        else
            y=y % x;
    }
    while (x!=0 && y!=0);
    nod=x+y;
    cout << "НОД этих чисел равен " << nod;
    getch ();
    return 0;
}
```

Группа операторов

Если условие истинно,  
то цикл выполняется  
еще раз.

# Цикл с параметром (for) в языке C++

---

- Цикл с параметром будет иметь следующий формат:

for (инициализация; выражение;  
модификация) оператор;



# Применение цикла с параметром (for)

---

```
int main ()
{
    setlocale (LC_ALL, "Russian");
    int sum=0;
    for (int i=1; i<=10; i++)
        sum+=i;
    cout <<sum;
    getch ();
    return 0;
}
```

# Объявление массива

---

```
using namespace std;  
  
const int N = 5; //константа  
  
int main ()  
{  
    setlocale (LC_ALL, "Russian");  
    int a[N]; //размер массива задан через константу  
    getch ();  
    return 0;  
}
```



# Ввод элементов массива вручную

```
const int N = 5; //константа

int main ()
{
    setlocale (LC_ALL, "Russian");
    int a[N];           //размер массива задан через константу
    cout << "Введите элементы целочисленного массива: \n";
    for (int i=0; i<N; i++)
        cin >> a[i];

    cout << "Получен целочисленный массив со следующими элементами: \n";
    for (int i=0; i<N; i++)
        cout << "a["<<i<<"]="<< a[i] << ' ';

    getch ();
    return 0;
}
```

**Ввод элементов  
одномерного массива**

**Вывод на экран**

# Заполнение массива случайными числами

```
#include <ctime>

using namespace std;

const int N = 10; //константа

int main ()
{
    setlocale (LC_ALL, "Russian");

    int a[N];          //размер массива задан через константу

    cout << "Массив заполняется числами из диапазона [-30;100]: \n";

    srand(time(NULL)|clock());
    for (int i=0; i<N; i++)
        a[i]=-30+rand()%131;

    cout << "Получен целочисленный массив со следующими элементами: \n";

    for (int i=0; i<N; i++)
        cout << "a["<<i<<"]="<< a[i] << ' ';

    getch ();
    return 0;
}
```

**Заполнение массива случайными числами**



