Основы языка программирования C++

Структура программы на языке C++

В состав программы входят:

- директивы препроцессора
- операторы (инструкции)
- комментарии

Пример программы на C++

```
// программа вводит с консоли число
// и выводит его на консоль
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout << "input number\n";
   cin >> a;
   cout << "number = " << a << "\n";
   return 0;
}
```

Пример программы на C++

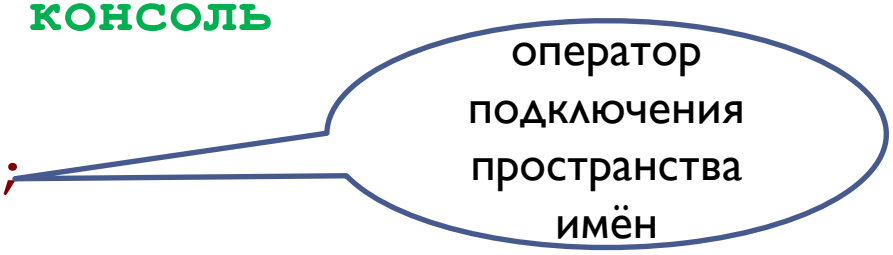
```
// программа вводит с консоли число
// и выводит его на консоль
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout << "input number\n";
   cin >> a;
   cout << "number = " << a << "\n";
   return 0;
}
```

Пример программы на C++

```
// программа вводит с консоли число
// и выводит его на консоль
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout << "input number\n";
   cin >> a;
   cout << "number = " << a << "\n";
   return 0;
}
```

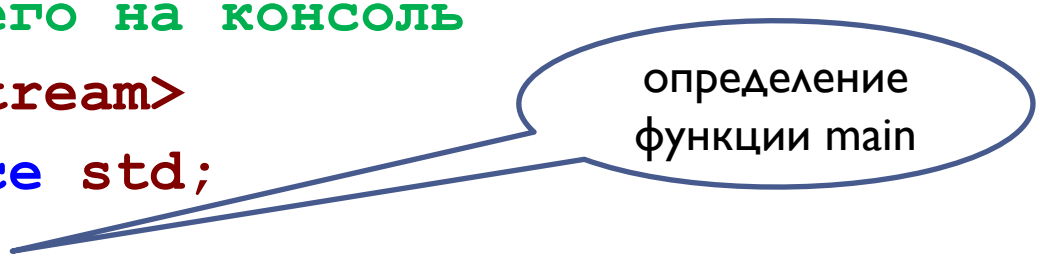
Пример программы на C++

```
// программа вводит с консоли число
// и выводит его на консоль
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout << "input number\n";
   cin >> a;
   cout << "number = " << a << "\n";
   return 0;
}
```



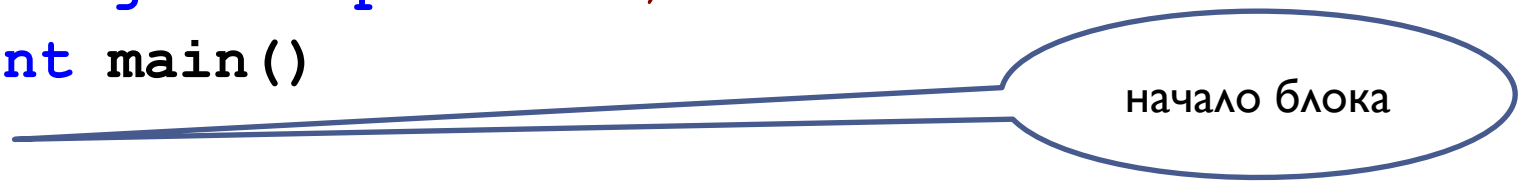
Пример программы на C++

```
// программа вводит с консоли число
// и выводит его на консоль
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout << "input number\n";
   cin >> a;
   cout << "number = " << a << "\n";
   return 0;
}
```



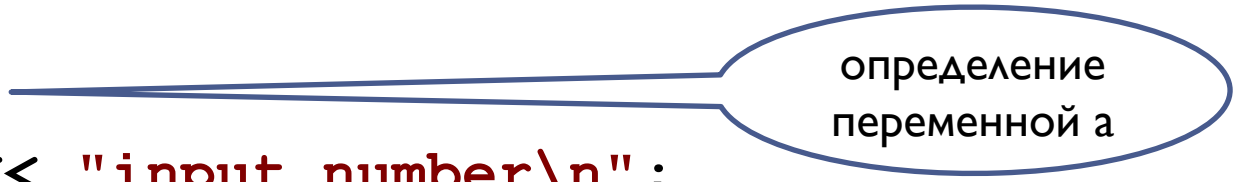
Пример программы на C++

```
// программа вводит с консоли число
// и выводит его на консоль
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout << "input number\n";
   cin >> a;
   cout << "number = " << a << "\n";
   return 0;
}
```



Пример программы на C++

```
// программа вводит с консоли число
// и выводит его на консоль
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout << "input number\n";
   cin >> a;
   cout << "number = " << a << "\n";
   return 0;
}
```

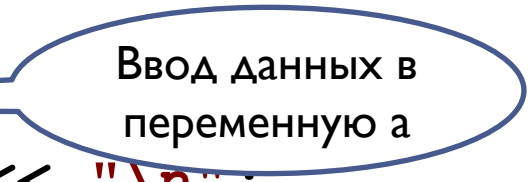


Пример программы на C++

```
// программа вводит с консоли число
// и выводит его на консоль
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout << "input number\n";
   cin >> a;
   cout << "number = " << a << "\n";
   return 0;
}
```

Пример программы на C++

```
// программа вводит с консоли число
// и выводит его на консоль
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout << "input number\n";
   cin >> a;
   cout << "number = " << a << "\n";
   return 0;
}
```

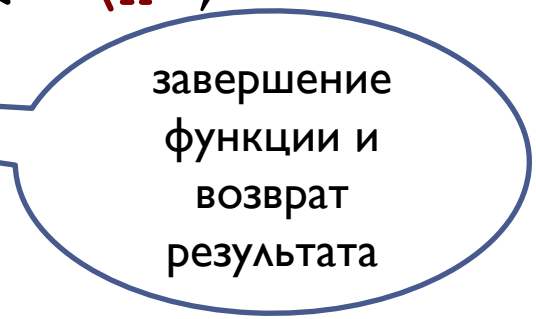


Пример программы на C++

```
// программа вводит с консоли число
// и выводит его на консоль
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout << "input number\n";
   cin >> a;
   cout << "number = " << a << "\n";
   return 0;
}
```

Пример программы на C++

```
// программа вводит с консоли число
// и выводит его на консоль
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout << "input number\n";
   cin >> a;
   cout << "number = " << a << "\n";
   return 0;
}
```



Пример программы на C++

```
// программа вводит с консоли число
// и выводит его на консоль
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout << "input number\n";
   cin >> a;
   cout << "number = " << a << "\n";
   return 0;
}
```

конец блока

Целочисленные типы данных C++

- **int** (целый, его размер определяется компилятором, обычно – 2 или 4 байта);
- **char** (символьный, как правило, 1 байт);
- **wchar_t** (предназначен для хранения набора символов, для которых недостаточно 1 байта, например, для кодировки Unicode. Как правило, занимает 2 байта);
- **bool** (предназначен для хранения логических величин, 0 интерпретируется как false, а любое ненулевое значение – как true. при преобразовании из типа **bool** к другому значение true переводится в 1);

Кроме того, можно указывать *спецификаторы типа*, которые уточняют внутреннее представление и диапазон значений типа:

- **short**;
- **long**;
- **long long**;
- **signed**;
- **unsigned**.

Другие стандартные типы данных C++

- **float** (числа с плавающей точкой длиной 4 байта);
- **double** (числа с плавающей точкой длиной 8 байт);
- **long double** (числа с плавающей точкой длиной 10 байт);
- **void** (т.н. ”пустой” тип, используется для определения функций, которые не возвращают значений или не имеют аргументов, обозначения абстрактных указателей и для некоторых других целей).

Самоопределённые константы (литералы)

Тип константы	Пример
Целая десятичная	`8, 0, -5, 4U, 3000L`
Целая восьмеричная	`077, 0111`
Целая шестнадцатеричная	`0xFFA2, 0X00FF`
Вещественная	`3.5, 0.2e6, .1E10`
Символьная	`'s', 'П', '\n', 'ss', '\0xFF', '\077'`
Строковая	`"Здесь был я", "ЗАО \"МММ\"", "C:\\autoexec.bat"`

Запись длинных строковых констант

```
"Эта строковая константа размещена \_
на нескольких строках \_
программы"
```

Красным цветом обозначен невидимый пробел!

Переменные

Переменная – это именованная область памяти, в которой хранятся данные определенного типа. Оператор описания переменной в общем случае выглядит так:

[класспамяти] [**const**] тип {имя [инициализатор]}…;

Класс памяти может быть задан с помощью слов **auto**, **extern**, **static** либо **register**.

Модификатор **const** позволяет задать именованные константы времени выполнения.

Инициализатор позволяет присвоить начальное значение переменной (и обязателен при описании константы). Его можно записать в двух формах:

= *значение*

или

(*значение*)

Значение выражения в инициализаторе должно быть вычислимо в процессе компиляции!

Примеры определения переменных

- `short int a = 1;`
- `const char CR = '\n';`
- `char s, sf('a'), st = '1';`
- `static unsigned int P;`

Операции

- Операции содержат знак операции (иногда слово) и операнды.
- Результат вычисления операции может быть использован далее при вычислении выражений.
- Некоторые операции изменяют значение одного из операндов *(в дальнейшем такие операнды обозначены красным)*

Арифметические операции

Запись операции (a, b, c – операнды)	Описание операции
a+ b	сумма
a – b	разность
a * b	произведение
a / b	частное (если оба операнда целые, выполняется целочисленное деление)
a % b	остаток от деления
a++	постфиксный инкремент (результат – старое значение операнда)
++a	префиксный инкремент (результат – новое значение операнда)
a – –	постфиксный декремент
– – a	префиксный декремент
– a	унарный минус – изменение знака

Присваивание и связанные с ним операции

Запись операции (a, b, c – операнды)	Описание операции
a = b	присваивание: a получает значение b
a += b	то же, что a = (a + b)
a –= b	то же, что a = (a – b)
a *= b	то же, что a = (a * b)
a / =b	то же, что a = (a / b)
a %= b	то же, что a = (a % b)

Операции сравнения

Запись операции (a, b, c – операнды)	Описание операции
a == b	Возвращает истину, если a равно b
a != b	Возвращает истину, если a не равно b
a < b	Возвращает истину, если a меньше b
a <= b	Возвращает истину, если a меньше или равно b
a > b	Возвращает истину, если a больше b
a >= b	Возвращает истину, если a больше или равно b

Логические и побитовые операции

Запись операции (a, b, c – операнды)	Описание операции
! a	Отрицание. Возвращает истину, если a – ложь, и наоборот
a || b	Логическое «или». Возвращает истину, если a или b истинно
a && b	Логическое «и». Возвращает истину, если a и b истинно
a | b	Побитовое «или». Операция «или» выполняется для всех битов операндов
a & b	Побитовое «и». Операция «и» выполняется для всех битов операндов
a ^ b	Побитовое «исключающее или». Операция «и» выполняется для всех битов операндов
~a	Инверсия битов операнда
a << b	Сдвиг значения a на b битов влево