

# Введение в C++

{

# Структура программы на C++



# Структура программы на C++

# директивы препроцессора

.....

# директивы препроцессора

функция а ( )

операторы

функция в ( )

операторы

void main ( ) //функция, с которой начинается выполнение программы

операторы

описания

присваивания

функция

пустой оператор

составной

выбора

циклов

перехода

# Состав языка



# Состав языка

*Алфавит языка СИ++, который включает*

- ▣ прописные и строчные латинские буквы и знак подчеркивания;
- ▣ арабские цифры от 0 до 9;
- ▣ специальные знаки "{, | []()+-/ %\*.\' : ; & ? < > = ! # ^
- ▣ пробельные символы (пробел, символ табуляции, символы перехода на новую строку).

Из символов формируются **ЛЕКСЕМЫ** языка:

- ▣ **Идентификаторы** – имена объектов СИ-программ. В идентификаторе могут быть использованы латинские буквы, цифры и знак подчеркивания. Прописные и строчные буквы различаются, например, PROG1, prog1 и Prog1 – три различных идентификатора. Первым символом должна быть буква или знак подчеркивания (но не цифра). Пробелы в идентификаторах не допускаются.
- ▣ **Ключевые (зарезервированные) слова** – это слова, которые имеют специальное значение для компилятора. Их нельзя использовать в качестве идентификаторов.
- ▣ **Знаки операций** – это один или несколько символов, определяющих действие над операндами. Операции делятся на унарные, бинарные и тернарную по количеству участвующих в этой операции операндов.
- ▣ **Константы** – это неизменяемые величины. Существуют целые, вещественные, символьные и строковые константы. Компилятор выделяет константу в качестве лексемы (элементарной конструкции) и относит ее к одному из типов по ее внешнему виду.
- ▣ **Разделители** – скобки, точка, запятая пробельные символы.

# Константы в C++

*Константа* – это лексема, представляющая изображение фиксированного числового, строкового или символьного значения.

Константы делятся на 5 групп:

▣ целые;

▣ вещественные (с плавающей точкой);

▣ перечислимые;

▣ символьные;

▣ строковые.

Целые константы:

**Десятичные:** последовательность десятичных цифр, начинающаяся не с 0, если это число не 0 (примеры: 8, 0, 192345)

**Восьмеричные:** это константа, которая всегда начинается с 0. За 0 следуют восьмеричные цифры (примеры: 016 – десятичное значение 14, 01)

**Шестнадцатеричные:** последовательность шестнадцатеричных цифр, которым предшествуют символы 0x или 0X (примеры: 0xA, 0X00F)

Вещественные константы:

**С фиксированной точкой:** Вид константы с фиксированной точкой: [цифры].[цифры] (примеры: 5.7, .0001, 41.)

**С плавающей точкой:** Вид константы с плавающей точкой: [цифры][.][цифры]E|e[+|-][цифры] (примеры: 0.5e5, .11e-5, 5E3)

Перечислимые константы:

```
enum { one=1, two=2, three=3, four=4};
```

`enum {zero,one,two,three}` – если в определении перечислимых констант опустить знаки = и числовые значения, то значения будут приписываться по умолчанию. При этом самый левый идентификатор получит значение 0, а каждый последующий будет увеличиваться на 1.

```
enum { ten=10, three=3, four, five, six};
```

```
enum {Sunday, Monday, Tuesday,  
Wednesday, Thursday, Friday, Saturday} ;
```

## Символьные константы:

Для представления символов, не имеющих графического отображения, например:

`\a` – звуковой сигнал,

`\b` – возврат на один шаг,

`\n` – перевод строки,

`\t` – горизонтальная табуляция.

Для представления символов: `\, \' , ? , \"` (`\\ , \' , \? , \"`).

Для представления символов с помощью шестнадцатеричных или восьмеричных кодов (`\073`, `\0xF5`).

Строковая константа – это последовательность символов, заключенная в кавычки. Внутри строк также могут использоваться управляющие символы.

Например: `“\nНовая строка”`,

`“\n”` Алгоритмические языки программирования высокого уровня `“”`.

# Типы данных в C++

<u>int</u> (целый)	}	целочисленные
<u>char</u> (символьный)		
<u>wchar_t</u> (расширенный символьный)		
<u>bool</u> (логический)		
<u>float</u> (вещественный)	}	с плавающей точкой (число=мантисса $\times 10^k$ )
<u>double</u> (вещественный с двойной точностью)		

Существует 4 спецификатора типа, уточняющих внутреннее представление и диапазон стандартных типов

- short (короткий)
- long (длинный)
- signed (знаковый)
- unsigned (беззнаковый)

# Операции в C++

## Унарные операции:

- `&` – операция взятия адреса.
- `*` – операция обращения по адресу.
- `-` – унарный минус.
- `+` – унарный плюс.
- `!` – отрицание.
- `++` – автоувеличение на 1.
- `--` – автоуменьшение на 1.
- `sizeof` – операция вычисления размера в байтах.

# Операции в C++

## Бинарные операции:

- + – бинарный плюс.
- – бинарный минус.
- \* – умножение.
- / – деление (при делении двух целых чисел получается целая часть от частного).
- % – получение остатка от деления.

# Операции в C++

## Операции присваивания:

= – присвоить операнду из левой части значение выражения из правой части.

+= – присвоить операнду из левой части сумму операндов левой и правой частей.

-= – присвоить операнду из левой части разность операндов левой и правой частей.

/= – присвоение частного от деления.

%= – присвоение остатка от деления.

# Операции в C++

## Операции присваивания:

= – присвоить операнду из левой части значение выражения из правой части.

+= – присвоить операнду из левой части сумму операндов левой и правой частей.

-= – присвоить операнду из левой части разность операндов левой и правой частей.

/= – присвоение частного от деления.

%= – присвоение остатка от деления.

# Операции в C++

## Операции сравнения:

- < – меньше.
- > – больше.
- <= – меньше или равно.
- >= – больше или равно.
- == – равно.
- != – не равно.

# Операции в C++

## Логические бинарные операции:

`&&` – логическое И.

`||` – логическое ИЛИ.