# Литература по курсу


1. Подбельский В.В. Язык Си++: Учебное пособие.- М.: Финансы и статистика, 2009.- 560 с.
2. Подбельский В.В., Фомин С.С. Программирование на языке Си: Учебное пособие.- М.: Финансы и статистика, 2009.- 600 с.
3. Керниган Б., Ритчи Д. Язык программирования С, 2-е издание.: Пер. с англ.- М.: Издательский дом «Вильямс», 2011. 304 с.
4. Румянцев П.В. Азбука программирования в Win 32 API.- М.: Горячая Линия - Телеком, 2004. – 312 с.

# *Тема № 1. Основные элементы языка программирования Си++*

# *Введение. Историческая справка*

Разработан в США сотрудниками фирмы Bell Laboratories в начале 70-х годов, использован для разработки ОС UNIX.

Первое описание языка дано его разработчиками – Б. Керниганом и Д. Ритчи. После было разработано десятки реализаций языка Си, поддерживающий разный диалект.

В 1983 г. при Американском Национальном Институте стандартов (ANSI) образован комитет по стандартизации языка Си, в 1989 г. ANSI C или C89.

В 1999 г. стандарт пересмотрен в международной организации по стандартизации ISO (International Organization for Standardization) принят стандарт ISO 9899:1999 или просто «C99».

8 декабря 2011 опубликован новый стандарт для языка Си (ISO/IEC 9899:2011) или просто «C11».

# Особенности языка Си:

- язык Си – язык программирования «среднего» уровня: поддерживает операции низкого уровня (операции над битами), базовые типы отражают те же объекты, что и язык Ассемблера (байты, машинные слова, символы, строки) в то же время имеет основные управляющие конструкции присущие языкам «высокого» уровня;
- язык со слабой типизацией, с одной стороны слабый контроль за преобразованием типов повышает эффективность программ, с другой, затрудняет отладку;
- относительно мал по объему, в нем отсутствуют встроенные операторы ввода-вывода, динамического распределения памяти, управления процессами и т.д., однако, в системное окружение входят различные библиотеки функций, что позволяет отделить особенности архитектуры компьютера от реализации языка.

# Переход к Си++

Начало 70-х годов появилась новая концепция программирования объектно-ориентированное программирование (ООП).

Суть процедурного программирования – программа (процедура) состоит из последовательности операторов, основной оператор присваивания, служащий для изменения содержимого памяти.

Объект= данные + методы (функции) для обработки данных.

80-е годы тенденция объединения стилей – в процедурные языки программирования вводятся объектно-ориентированные средства, например, СП Turbo Pascal v. 5.5, Си++.

Язык Си++ появился в 80-х годах первоначально назывался «Си с классами», название Си++ появилось в 1983 г. (в Си существует операция ++ (инкремент – увеличение на 1). Один из создателей Бьёрн Страуструп.

В 1998 г. принят стандарт языка Си++ ISO/IES 14882.

Следующий этап развития Си++ появление инструментальных средств визуального программирования на базе языка Си++ (Borland C++ Builder, Microsoft Visual C++ и т. д.)

# 1.1. Алфавит языка Си++

- прописные и строчные буквы английского алфавита: A,…,Z, a,…,z;
- арабские цифры 0, 1,…, 9;
- специальные символы:
  - пробельные символы: пробел, табуляция, перевод строки, новая строка, возврат каретки, новая страница, вертикальная табуляция,
  - разделители: , . ; : ? ‘ ! “ / | \ ~ _ ( ) { } [ ] > < # % & ^ - = + *.

Символы алфавита находятся в первой половине кодовой таблицы (первые 128 символов) кодировки ASCII.

Из символов алфавита строятся лексемы.

Лексема – единица текста программы, имеющая для компилятора самостоятельный смысл.

# 1.2. Идентификаторы (имена) языка Си++

Идентификатор(имя) – это лексема, представляющая собой последовательность букв английского алфавита, десятичных цифр, символа подчеркивания, начинающаяся не с цифры (в некоторых компиляторах разрешается внутри идентификатора использовать $).

Примеры идентификаторов: x, X, abc124, A_c3, x4er, x_, __d.

Неверные идентификаторы: 1c, 4sd.

Следует отметить, что большие и малые буквы в идентификаторах отличаются, т.е. x и X разные идентификаторы.

# 1.3. Ключевые слова

Ключевые слова – это идентификаторы, зарезервированные для специального использования.

Названия стандартных типов, управляющих конструкций, и т.п.

int double char float long …. if else for while do …. struct

Программист не может их использовать в качестве имен своих объектов.

# 1.4. Константы (литералы)

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

Типы констант:

- целые:
  - два типа:
    - тип int (по умолчанию) в MS DOS – 2 байта, в Win32 – 4 байта, примеры: 1245, 6, 175, 5,  1425;
    - тип long – 4 байта, если диапазон выходит за тип int, или явно указывается тип с помощью суффикса l (L), примеры: 12l, 14567L, 125234;
  - три формы представления:
    - в десятичной СС – 1234, 378l, 346;
    - в шестнадцатеричной  СС – 0x10, 0x10acd, 0XFFFF;
    - в восьмеричной СС – 010, 070, 01237;

    константы могут быть представлены типами unsigned int и unsigned long, для явного указания можно использовать суффикс u (U);

- вещественные:
  - три типа:
    - тип double (по умолчанию) – 8 байт, примеры: 12.5, .123, 0.5, 1. ;
    - тип float – 4 байта, используется суффикс f (F), примеры: 10.5f, 0.123F ;
    - тип long double – 10 байт, используется суффикс l (L), примеры: 10.5l, 0.9L;
  - две формы представления:
    - форма с точкой, примеры: 10.125, 1. , .125 , 0.125 ;
    - форма со знаком экспоненты: 1e -5 , 12.23E4F ;
- символьные:
  - односимвольные (занимают памяти 1 байт): это буква, цифра, знак пунктуации или специальный символ, заключенный в апостроф.

‘a’, ‘d’, ‘1’, ‘.’, ‘ ‘, представлены в памяти типом char;

- многосимвольные (до 4-х символов): ‘asdf’, ‘GR’, представлены в памяти типом int (первый символ – младший байт).

Последовательность литер, начинающихся с ‘\’ называется эскейп-последовательностью.

‘\\’, ‘\’’, ‘\””, ‘\n’ – перевод строки (новая строка) (код 10), ‘\r’ – возврат каретки (код 13), ‘\a’ – звуковой сигнал (7), ‘\b’ – возврат на шаг (забой) (8),

Допустимо использовать коды символов от 0 до 255,

заданные в восьмеричной или шестнадцатеричной СС. ‘\127’, ‘\10’, ‘\xff’, ‘\xf’.

– строковые константы – последовательность символов, заключенных в кавычки (не в апострофы), внутри строковых констант допускается использовать эскейп-последовательности. Пример:

“начало строки\nтекст с новой строки”

# 1.5. Операции (операторы)

```
[ ]    ( )      .      ->      ++      --
&       *
 +      -       ~      !     sizeof    /
%       <<
 >>     <       >      <=      >=      ==
!=      ^
 |      &&      ||     ?:      =       *=
/=      %=
 +=     -=      <<=    >>=     &=      ^=
|=      ,       #      ##
```

# 1.6. Комментарии

Два вида комментариев:

- // комментарий до конца строки
- /* Комментарий на
  несколько
  строк */

# 1.7. Структура программы на Си (Си++)

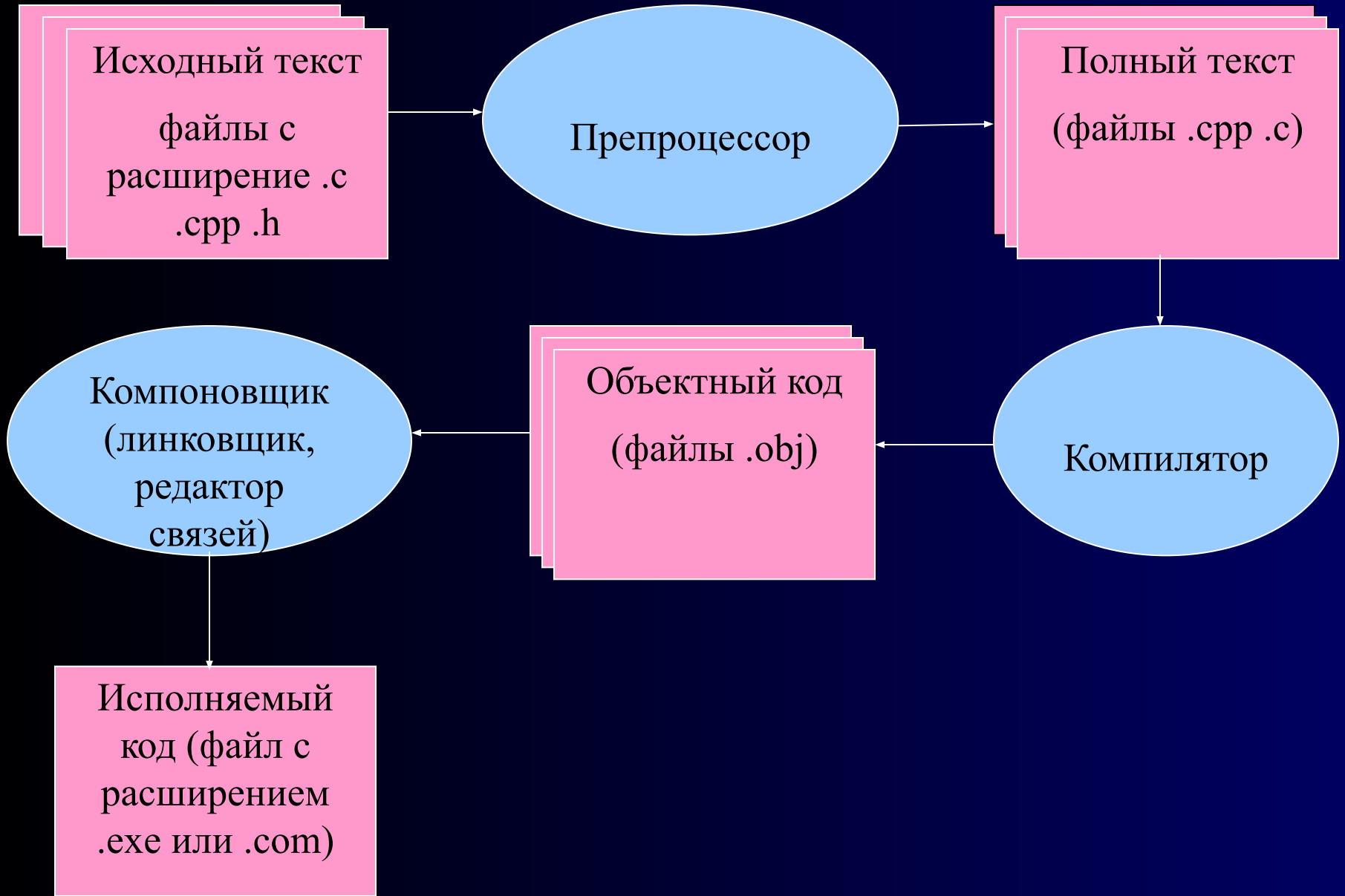
Основная программная единица текстовый файл с расширением .c или .cpp . (.cc)

Программа может включать в себя следующие основные элементы:

– директивы препроцессора;
– указания компилятору;
– описания;
– определения.

Для того чтобы программа могла выполняться она должна содержать хотя бы одно определение – определение функции main (WinMain) (точка входа в программу).

# 1.8. Последовательность обработки программы на Си (Си++)



# Код программы

- /----------
- #include <vcl.h> - Заголовочный модуль
- #pragma hdrstop
- //----------
- #pragma argsused
- int main(int argc, char* argv[]) – Точка входа в программу
- { *составляется программа разработчиком*
- return 0;
- }
- //----------

***printf***

Синтаксис:

**int** printf (Формат, *СписокПеременых) ;*

Выводит на экран значения переменных. Формат вывода задается в строке форматирования, которая помимо спецификатора формата может содержать текст и управляющие символы. Значение первой переменной выводится в соответствии с первым спецификатором формата, второй — со вторым, и т. д.

Спецификаторы формата (необязательный параметр n задает ширину поля вывода).

Заголовочный файл: <stdio.h>

```
printf("Вычисление площади прямоугольника\n");
```

***scanf***

Синтаксис:

**int** scanf **(const chsur*** *Формат,*
*СписокАдресовПеременных) ;*

*Вводит с клавиатуры значения переменных, в соответствии с указанным спецификатором формата. Первая переменная получает значение в соответствии с первым спецификатором формата, вторая — со вторым и т. д.*

*Заголовочный файл: <stdio.h>*

***Puts***

Синтаксис:

puts(const char* *Строка);*

*Выводит на экран строку символов и переводит курсор в начало следующей строки экрана. В качестве параметра функции можно использовать строковую константу или строковую переменную.*

Заголовочный файл: <stdio.h>

***Gets***

Синтаксис:

**char *gets (chau:* s) ;**

*Вводит с клавиатуры строку символов. Вводимая строка может содержать пробелы.*

Заголовочный файл: <stdio.h>

***Putch***

Синтаксис:

**int putch(int c);**

*Выводит на экран символ.*

Заголовочный файл: <conio.h>

***Getch***

Синтаксис:

**int getch(void);**

Возвращает код символа нажатой клавиши. Если нажата служебная клавиша, то функция getch возвращает 0. В этом случае, для того, чтобы определить, какая служебная клавиша нажата, нужно обратиться к функции getch еще раз.

*Замечание*

*Функция getch не выводит на экран символ, соответствующий нажатой клавише.*

Заголовочный файл: <conio.h>

***Cputs***

Синтаксис:

cputs(**const char*** *Строка);*

Выводит на экран строку.

Цвет выводимых символов можно задать при помощи функции textcolor , цвет фона — при помощи функции textbackground.

*Замечание*

*Для перехода к началу следующей строки вместо \n следует использовать символы \n \r , иначе курсор лишь переводится'на новую строку, но не возвращается к левой границе окна.*

Заголовочный файл: <conio.h>

Пример:

Написать программу вычисления площади параллелограмма.

*Ниже приведен рекомендуемый вид экрана во время выполнения программы (данные, вводятся пользователем).*

Вычисление площади прямоугольника

Введите исходные данные:

Длина (см) -> **9**

Ширина (см) -> 7.5

Площадь параллелограмма: 67.50 кв.см.

```
float l,w; // длина и ширина прямоугольника
float s; // площадь прямоугольника
printf("Вычисление площади прямоугольника\n");
printf("Введите исходные данные:\n");
printf("Длина (см.) -> " );
scanf("%f", &1);
printf("Ширина (см.) -> ") ;
scanf("%f", &W);
S = 1 * w;
printf("Площадь прямоугольника: %10.2f кв.см.\n", s);
printf("\n\n.  Для завершения нажмите <Enter>");
getch();
```