Лекция № 1

История появления языка С++

Сотрудник фирмы Bell Labs Денис Ритчи создал язык С (читается “си”) в 1972 г. во время совместной работы с Кеном Томпсоном над операционной системой UNIX. Прообразом послужил язык B (“би”), разработанный Томпсоном. А свое начало язык B берет от языка APL (“эй пи эль”). Язык С был разработан как инструмент для программистов-практиков. Большая часть ОС UNIX была написана на этом языке. В соответствие с потребностью решать такую главную задачу как написание кода ОС, целью авторов было создание удобного и полезного языка.

Эти критерии, конечно, учитывались и при разработке множества других языков. Но разработка других языков преследовала и другие цели, например:

- Pascal — язык на основе которого можно было бы обучать фундаментальным основам и принципам программирования;
- Basic — синтаксис языка близок к английскому языку; предназначен для быстрого освоения программирования непрофессионалами.

В эпоху развития объектно-ориентированного программирования и появления объектно-ориентированных языков и такой универсальный язык как С тоже получил развитие в этой области. Новый язык, включающий в себя объектно-ориентированное расширение, получил название С++ (“си плас плас” от английского “c plus plus”; “си плюс плюс”).

Достоинства языка С++

К основным достоинствам языка С++ необходимо отнести следующие моменты:

- С++ — современный язык. Он включает в себя управляющие конструкции, рекомендуемые теоретическим и практическим программированием.
- С++ — эффективный. Структура позволяет наилучшим образом использовать возможности современных ЭВМ. Программы отличаются компактностью и быстротой исполнения.
- С++ — переносимый или мобильный. Программа, написанная для одной вычислительной системы, может быть перенесена почти без изменений на другую. Компиляторы реализованы почти на 40 типах вычислительных систем, начиная от 8-ми разрядных процессоров и кончая CRAY-1 один из мощных суперкомпьютеров.
- С++ — мощный и гибкий. Большая часть ОС UNIX написана на С.
- С++ — удобный язык. Достаточно структурирован, чтобы поддерживать хороший стиль программирования, в то же время не накладывает больших ограничений.
- С++ — язык " компилирующего" типа.

Структура программы

Исходная программа на языке С состоит из следующих частей:

- директив препроцессора;
- указаний компилятору;
- объявлений;
- определений.

Эти части имеют разное предназначение в тексте программы:

Директивы — специфицируют действия препроцессора по преобразованию текста программы перед компиляцией.

Указания — это специальные инструкции, которым компилятор С++ следует во время компиляции.

Объявления — задают имя и атрибуты данных, их начальные значения явно или по умолчанию.

Определение — функции специфицирует тело функции, которая представляет собой составной оператор (блок операторов), содержащий объявления и операторы.

Объявления типа — позволяет программисту создавать собственный тип данных. Для типа понятия объявления и определения совпадают.

Исходная программа может содержать произвольное число директив, указаний, объявлений и определений. Но при этом их порядок существенен.

Константы в Си

Константы используются для задания постоянных величин непосредственно в тексте программы. В языке С++ различают несколько типов констант.

Целые константы специфицируют положительные значения. Состоят из последовательности цифр. Знак "–" рассматривается как унарная арифметическая операция.

Примеры: 134 2 67894

Длинные целые обозначаются как <цифра>…<цифра>L. Если константа велика для типа **int**, то она считается длинной.

Примеры: 134L 2L 67894L

Помимо этого существуют правила для записи константы в формате 8-ых и 16-ых системах счисления:

- если число начинается с цифры <**0**> (ноль) то это целое число задано в 8-ой системе счисления.
  Примеры: 037 037L 12345L
- если число начинается с **0x** то это целое число задано в 16-ой системе счисления.
  Примеры: 0x3EA7 0x3D7L 0x12AF4L

Замечание. Между цифрами числовых констант пробелы *недопустимы*.

Константы с плавающей запятой или *вещественные* константы всегда представляются числом с двойной точностью, то есть как тип **double**. В полном формате эти константы состоят из следующих частей:

целой части — последовательность цифр;
десятичной точки;
дробной части — последовательность цифр;
символа экспоненты **e** и **E**;
экспоненты в виде целой константы (может быть со знаком "–").

Одна часть (но не обе сразу) из нижеследующих пар может быть опущена:

- *целая* или *дробная* часть
- *десятичная точка* или *символ* **e**(**E**) *и экспонента* в виде целой части.

Примеры: 345. 3.1415926 2.1E5 .123E3 4037e–5

Экспонента обозначает, что мантисса числа (вещественное число до символа **e**) должна быть умножена на 10 в степени этой степени. Так запись 12.98e–3 будет эквивалентна $12.98 \cdot 10^{-3}$ в привычной записи на бумаге. Следует отметить, что это же число можно было записать и как: 1.298e–2 129.8e–4 0.01298 и т.п.

Замечание. Использовать пробел при записи констант *запрещается*.

Символьные константы — состоят из одного текстового символа заключенного в одинарные апострофы: 'x', 'o', 'Z'. Некоторые символы не имеют графического представления (специальные символы, которые невозможно ввести с клавиатуры), можно набрать, используя специальные комбинации цифровые или символьные. Такие комбинации начинаются с обратной косой черты, а число указывается в 8-ой системе счисления:

'\007' — код символа в 8-ой системе счисления;
'\n' — код символа новая строка;
'\t' — код символа табуляции;
'\0' — код символа со значением 0;

‘\\’ — код символа обратная наклонная черта;
‘\’’ — код символа одиночная кавычка;
и т.д.

Вся эта комбинация состоит из пары символов или более, но компилятор заменяет ее на один символ.

Строчные константы — это последовательность символов заключенная в двойные кавычки.

Пример: “Томск – город студентов.”

Необходимо понимать, что конструкции ‘x’ и “x” формируют *разные* константы. Дело в том, что *строчная константа* при размещении в памяти заканчивается символом <**0**> (ноль), как указатель окончания текста. Поэтому конструкция из одного символа, но указанная как строчная константа, потребует памяти в два байта — для самого символа и для завершающего ноля.

Пример простой программы

Простейшая *программа* на языке С++, как и на других языках программирования, выглядит достаточно несложной и имеет аналогичные операторы в других языках таких как, например, Pascal или Basic.

Таблица

Примеры простейших программ на языках программирования С++ и Pascal

Программа на С++	Комментарии	Программа на Pascal
	Заголовок программы	`Program First;`
`#include <stdio.h>`	Подключение заголовочных файлов | Подключение модулей	`Uses math;`
	Описание переменных	`var a, b, c;`
`void main ()`	Заголовок главной функции	
`{`	Скобка	**`Begin`**
`int a = 5, b = 6, c;`	Описание переменных и инициализация переменных	
	Инициализация переменных	`a := 5; b := 6;`
`c = a + b;`	Вычисление суммы	`c := a + b;`
`printf ("Привет, мир !\n");`	Вывод текстовой строки	`WriteLn (‘Привет, мир!’);`
`printf ("c = %d\n", c);`	Вывод результата суммирования	`WriteLn (‘c = ’, c);`
`}`	Скобка	**`end.`**

Даже простейшая программа, как правило, требует подключения дополнительных ресурсов, например, библиотек ввода–вывода. В С++ эти ресурсы располагаются в библиотеках и .OBJ файлах. Но при подключении этих библиотек требуется указывать, как обращаться к подпрограммам расположенным в них. Для этого используются специальные текстовые файлы — заголовочные. Такой файл имеет расширение .h и называется заголовочным или хидер-файлом (header file) и хранит описание различных функций и макроопределений, которые программист может использовать в программе.

Процесс обработки исходного модуля с текстом программы проходит в два этапа (поэтому процесс компиляции в С++ называют двухпроходным). На первом этапе компиляции программного текста компилятор занимается только обработкой самого текста, как это определено командами препроцессора. Команды препроцессора занимают как минимум одну строку и начинаются с символа “#”. Наиболее часто используется команда подключения `#include`.

Препроцессор — часть компилятора, которая осуществляет некоторую предварительную обработку текста программы перед началом компиляции. Символ “#” указывает, что строка должна обрабатываться препроцессором языка С++.

На месте строки оператора `#include <имя.h>` (по окончание работы препроцессора) оказывается содержимое файла, имя которого было указано в угловых скобках. Обычно таким образом в программу вводят описание библиотечных процедур.

В круглых скобках, в строке `main ()`, в общем случае содержится информация, передаваемая функции. В нашем примере такой информации нет. Необходимо отметить, что `Main ()` и `main ()` — разные функции: `main ()` — главная или головная функция программы и эта функция выполняется первой.

Следует отметить, что знак “`;`” является частью оператора на С++, а не разделителем операторов, как на языке Pascal. Тело функции заключается в блоковые скобки `{ }` (фигурные скобки).

Любая программа должна выводить результат вычислений. В этой программе с этой целью использован оператор `printf ()` — функция вывода, а в скобках указывается передаваемая ей информация. Параметры здесь состоят из двух частей — строки формата (в двойных кавычках) и списка переменных, значение которых требуется вывести (здесь лишь одна переменная `c`). В данном формате указано, какой текст требуется вывести и как выводить переменные. При печати на месте символа `%d`, будет выведено число, то есть `%d` служит для указания места, куда необходимо вставить значения `c` при печати; причем `%` указывает, что в этом месте необходимо выводить переменную, `d` — указывает формат, то есть, что эту переменную необходимо воспринимать как целочисленное данное.

Продолжение следует…