Лабораторная работа № 1

Тема: Изучение отличий языка C++ от языка C

Цель: Изучить основные отличия языка C++ от языка C: библиотека ввода/вывода, объявление структур, объединений и перечислений, работа с динамической памятью, перегрузка имен функций и передача параметров по ссылке.

Краткая теория

Интегрированная среда разработки «MS Visual Studio .NET» предназначена для создания программного обеспечения на основе технологии .NET Framework, которая применяется в настоящее время для разработки приложений под ОС Windows XP/Vista/7 и выше. Одной из отличительных особенностей этой технологии разработки ПО является возможность написания программ на различных языках программирования. Основными из поддерживаемых языков программирования в .NET являются: C++, C# и Visual Basic. Поддержку того или иного языка программирования в данной IDE можно как включать, так и не включать, поэтому довольно часто при ссылке на эту среду программирования ее название пишут с указанием языка программирования, на котором будет вестись разработка (например, MS Visual C++ или MSVisual C#).

Запуск среды «MS Visual C++» осуществляется путем запуска файла devenv.exe из рабочего каталога среды: C:\ProgramFiles\MicrosoftVisualStudio 9.0\Common7\IDE. Как правило, на рабочем столе или меню программ ОС Windows располагается ярлык для запуска данной среды разработки (Microsoft Visual Studio 2008/2010). Прежде чем перейти непосредственно к разработке программ необходимо произвести некоторые настройки среды разработки (в принципе настройки могут быть проведены и раньше, но проверить все же стоит). Окно настроек среды разработки вызывается из главного меню: Tools → Options… Данная среда разработки является одной из наиболее мощных и развитых систем программирования, поэтому она имеет довольно большое количество параметров, изучение которых может потребовать длительного времени. Поэтому в рамках данной лабораторной работы мы ограничимся рассмотрением лишь базовых параметров, а изучение остальных параметров настройки данной среды разработки оставляется на усмотрение студента.

Сначала выполним настройку шрифта, которым будет отображаться текст программы.

Для этого в дереве параметров (левая часть окна) выберем пункт «Fonts and Colors» в ветви «Environment» и на экране получим окно настроек. В этом

окне необходимо установить шрифт Courrier New, а размер шрифта рекомендуется выбрать в пределах 12 – 14 пт. Остальные параметры можно оставить без изменений.

Далее перейдем к параметрам настройки расположения программных проектов. Для этого выберем пункт «General» в ветви «Projects and Solutions». В этом окне необходимо указать место расположения по умолчанию создаваемых проектов. В качестве такого места целесообразно указать каталог, расположенные на диске D (например, D:\16IT) в поле ProjectsLocation. В полях Userprojecttemplatelocations и Useritemtemplatelocations указываются места расположения временных файлов, создаваемых для нужд среды разработки. Значения этих полей можно оставить по умолчанию.

Дополнительно можно выполнить настройку запуска среды разработки. Для этого необходимо выбрать пункт «Startup» в ветви «Environment». В появившемся окне в поле «At startup» необходимо выбрать «Show empty environment», а также убрать флажок «Download content every».

Для начала разработки программы в среде «Visual C++» необходимо создать проект. Это выполняется в окне создания проекта, которое вызывается из главного меню программы: File → New → Project. В открывшемся окне необходимо выбрать тип создаваемого проекта.

В качестве типа проекта необходимо выбрать в дереве в левой части окна пункт «Win32» в ветви «Visual C++», которая может располагаться в ветви «Other Languages». В окне справа необходимо выбрать вид приложения «Win32 Console Application». В нижней части окна необходимо указать название проекта (в поле Name), а также, при необходимости, изменить расположение проекта (поле Location).

Дополнительно здесь можно указать необходимость создания отдельной директории для проекта и ее имя.

После нажатия на кнопку «OK» появится следующее окно, в котором можно открыть вкладку «Application Settings». В этом окне необходимо убрать флажок «Precompiled header». Для завершения мастера просто нажмите «Finish».

После того, как проект создан в главном окне среды разработки «Visual C++» появится окно инспектора проекта, в котором будут отображены все файлы проекта. В проекте могут присутствовать заголовочные файлы (ветвь Header Files), файлы ресурсов (ветвь Resourse Files) и файлы исходного кода (ветвь Source Files). Двойной клик левой кнопкой мыши по имени файла приведет к открытию редактора текста программы для этого файла.

Среда Visual C++ генерирует исходный код, содержащий описание функции _tmain, которая является точкой входа в консольное приложение. Это описание не соответствует стандарту языка C++, поэтому его необходимо изменить. Для этого сначала необходимо закомментировать или удалить подключение библиотеки stdafx.h. Затем объявление функции _tmain следует исправить на стандартное, как и в языке C:

```
int main(int argc, char *argv[])
```

Для осуществления ввода и вывода данных в языке C++ используется библиотека iostream. Подключение библиотеки осуществляется также, как и в языке C: используя директиву #include. Дополнительно необходимо определить пространство имен, которое будет использоваться в программе. Для этого сразу после подключения библиотеки iostream необходимо вставить строку using namespace std. После этих преобразований можно приступать к написанию остальной программы.

После написания самой программы ее необходимо скомпилировать и запустить.

Построение программы осуществляется с помощью элементов меню Debug главного меню программы. В этом меню содержатся следующие пункты:

- Build Solution (F6) – компиляция и сборка решения;
- Rebuild Solution – перекомпиляция и сборка решения;
- Clean Solution – очистка временных, объектных и т.п. файлов для данного решения;
- Build <имя проекта> (Shift+F6) – компиляция и сборка проекта;
- Rebuild <имя проекта> – перекомпиляция и сборка проекта;
- Clean <имя проекта> – очистка временных, объектных и т.п. файлов для проекта.

В среде разработки Visual C++ в рамках одного решения (solution) может быть несколько проектов.

Если программа написана с ошибками, то компилятор выдаст соответствующие сообщение в нижнем поле «Error List» главного окна на вкладке «Errors», предупреждения выводятся на вкладке «Warnings», а остальные сообщения на вкладке «Messages». Если компиляция программы прошла успешно, то можно осуществить запуск программы. Это действие осуществляется с помощью меню Debug главного окна среды разработки. В этом меню есть два пункта:

- Start Debugging (F5) – запустить программу в режиме отладки;

- Start Without Debugging (Ctrl+F5) – запустить программу без отладки.

Режим отладки обладает довольно широкой функциональностью. Все действия режима отладки доступны в меню Debug главного меню программы:

- Continue (F5) – продолжить выполнение до следующей точки останова или конца программы;
- Stop Debugging (Shift+F5) – завершить отладку;
- Restart (Ctrl+Shift+F5) – перезапустить программу;
- Step Into (F11) – вход внутрь функции или переход к следующей строке;
- Step Over (F10) – переход через функцию или переход к следующей строке;
- Step Out (Shift+F11) – выход из функции (без дальнейшей ее трассировки).

Установка точек останова осуществляется командой Breakpoint → Insert Breakpoint контекстного меню редактора текста программы, удаление точки останова осуществляется командой Breakpoint → Delete Breakpoint, отмена (запрет) точки останова осуществляется командой Breakpoint → Disable Breakpoint.

Во время отладки в нижней части экрана выводятся окна, содержащие отладочную информацию. Так по умолчанию там присутствуют вкладки:

- Locals – содержит список локальных переменных и их текущие значения,
- Watch – содержит список переменных (и их текущих значений), указанных пользователем.

Добавление переменной в список Watch окна отладочной информации осуществляется командной Add контекстного меню данного окна или путем непосредственного ввода имени переменной в свободную ячейку этой вкладки.

В языке C++ ввод и вывод осуществляется также, как и языке C через потоки. Но в языке C++ используется библиотека iostream, в которой описаны классы istream и ostream. Класс istream предназначен для организации ввода данных, у этого класса перегружена операция >>, с помощью которой осуществляется ввод данных в переменные. Класс osteam предназначен для организации вывода данных, у этого класса перегружена операция <<, с помощью которой осуществляется вывод значений переменных. В любой консольной программе автоматически доступны объекты cin (класса istream) – стандартный поток ввода, cout (класс ostream) – стандартный поток вывода. Так как классы istream и ostream, а также объекты cin и cout описаны в

пространстве имен std, то при написании программы целесообразно включить в текст программы using namespace std.

Пример:

```
#include <iostream>
using namespace std;
int main(int argc, char* argv[])
{
int x = 0, y = 0;
  cout << "Введите X: ";
  cin >> x;
  cout << "Введите Y: ";
  cin >> y;
  cout << "Сумма " << x << '+' << y << " = " << x+y << '\n';
return 0;
}
```

Второе отличие языка C++ от языка C заключается в том, что в языке C++ не нужно использовать оператор typedef для создания новых типов структур, объединений или перечислений.

Пример:

```
#include <iostream>
using namespase std;
int main(int argc, char* argv[])
{
  struct Student{
char fio[3][16];
     int kurs;
float rate;
};
Student st;
cout << "Введите в формате \" Имя Фамилия Отчество\
курс успеваемость\"\n";
  cin >> st.fio[0] >> st.fio[1] >> st.fio[2] >> st.kurs >>
st.rate;
 cout << "Имя: " << st.fio[0] << ' ' << st.fio[1] << ' '
<< st.fio[2] << '\n';
cout << "Курс: " << st.kurs << '\n';
cout << "Успеваемость: " << st.rate << '\n';
return 0;
}
```

В языке C++ при объявлении функции без параметров допускается оставлять пустые скобки, не указывая ключевое слово void, как это делается в языке C.

```
#include <iostream>
using namespase std;
void Func();
int main(int argc, char* argv[])
```

```
{
  Func();
return 0;
}
void Func()
{
  cout << "Функция без параметров!\n";
}
```

В языке C++ реализован механизм передачи параметров по ссылке. Для этого используется описание параметра в виде:

тип &имя

Пример реализации функции возведения целочисленного значения в квадрат:

```
#include <iostream>
using namespase std;
void Sqr(int &v) { v *= v;}
int main(int argc, char* argv[])
{
int z = 4;
  cout << "Перед Sqr: " << z << '\n';
  Sqr(z);
  cout << "После Sqr: " << z << '\n';
return 0;
}
```

При передаче в параметрах функции большого значения (структура) целесообразно использовать передачу константной ссылки:

```
#include <iostream>
using namespase std;
struct Student{
char fio[3][16];
int kurs;
float rate;
};
void PrintStudent(const Student &s)
{
  cout << "Имя: " << s.fio[0] << ' ' << s.fio[1] << ' '
<< s.fio[2] << '\n';
cout << "Курс: " << s.kurs << "\nУспеваемость: "
<< s.rate << '\n';
}
int main(int argc, char* argv[])
{
  Student st = {{"Иванов", "Иван", "Иванович"},3,6.5};
  PrintStudent(st);
 return 0;
 }
```

Для работы с динамической памятью в языке C++ можно использовать функции из библиотеки stdlib.h (calloc, malloc, realloc, free), но целесообразнее использовать операторы new и delete (особенно при создании объектов).

Выделение памяти: указатель = new тип;

Освобождение памяти: delete указатель;

Примеры:

```
int *ptr = new int;
cin >> *ptr;
cout << "Value: " << *ptr << endl;
delete ptr;
ptr = new int (10);
cout << "Value: " << *ptr << endl;
delete ptr;
```

Выделение памяти под массив: указатель = new тип[размер];

Освобождение памяти: delete [] указатель;

Пример:

```
int *ptr = new int [10];
for(int i=0;i<10;i++) cin >> ptr[i];
for(int i=0;i<10;i++) cout << ptr[i] << ' ';
cout << endl;
delete [] ptr;
```

В языке C++ можно описывать функции с параметрами по умолчанию. Если при вызове функции значение данного параметра не указано, то используется значение по умолчанию.

Пример:

```
double Volume(double l, double =1.0, double =1.0);
int main(int argc, char* argv[])
{
  cout << "Объем: " << Volume(2,3,4) << endl;  //вывод: 24
  cout << "Объем: " << Volume(2,3) << endl;     //вывод: 6
  cout << "Объем: " << Volume(2) << endl;        //вывод: 2
return 0;
}
double Volume(double l, double w, double h)
{
return l*w*h;
}
```

В языке C++ допускается перегрузка функций – возможность использования одного и того же идентификатора для именования нескольких функций. Перегруженные функции должны различаться по количеству, порядку и типу формальных параметров.

Пример:

```
double Square(double);
double Square(double,double);
int main(int argc, char* argv[])
{
  cout << "Square of circle: " << Square(5) << endl;
cout << "Square of rectangle: " << Square(2,6) << endl;
return 0;
}
double Square(double r) { return 3.1415*r*r; }
double Square(double a, double b) { return a*b; }
```

Ход работы

В лабораторной работе необходимо выполнить два задания (работа с очередью и стеком). При выполнении задания необходимо реализовать дружественный интерфейс: при вводе(выводе) данных выводится приглашение, которое содержит описание вводимой (выводимой) величины (назначение и тип).

В данной лабораторной работе предполагается, что все значения могут вводиться некорректно. Поэтому необходимо осуществлять проверку на корректность ввода с использованием операторов управления.

I. Разработать программу согласно варианту задания. Необходимо реализовать следующие функции для работы с динамической очередью (последовательного или связного хранения):

- создание очереди;
- добавление элемента в очередь;
- удаление элемента из очереди;
- вывод очереди на экран;
- очистка очереди.

Используя разработанные функции выполнить обработку N очередей (число N вводит пользователь).

Варианты заданий:

1 Удалить из очереди все четные числа.

2 Удалить из очереди все отрицательные числа.

3 Поменять местами крайние элементы очереди.

4 Поменять местами минимальные и максимальные элементы очереди.

5 Удалить из очереди каждый второй элемент.

6 Удалить из очереди все элементы, расположенные до минимального элемента очереди.

7 Поменять местами наибольший среди отрицательных и наименьший среди положительных элементов очереди.

8 Поместить максимальный элемент очереди на первую позицию.

9 Поменять местами минимальные и первый элементы очереди.

10 Поменять местами первый и последний элементы очереди.

11 Удалить первый и последний элементы очереди.

12 Удалить из очереди все элементы, расположенные между минимальным и максимальным элементами очереди.

13 Удалить из очереди все элементы, стоящие после максимального элемента.

14 Найти среднее значение всех элементов очереди и удалить все элементы, которые меньше среднего значения.

15 Найти среднее значение всех элементов очереди. Поместить ближайший к среднему значению элемент очереди на первую позицию.

16 Дана очередь целых чисел. Удалить наибольший элемент очереди.

17 Дана очередь символов. Удалить каждый n-ый элемент очереди.

18 Дана очередь целых чисел. Удалить все четные числа из очереди.

19 Дана очередь символов. Удалить каждый четный элемент очереди.

20 Дана очередь вещественных чисел. Удалить числа, меньшие среднего арифметического.

21 Дана очередь целых чисел. Удалить наименьший элемент очереди.

22 Дана очередь символов. Удалить каждый нечетный элемент очереди.

23 Дана очередь целых чисел. Удалить все нечетные числа из очереди.

24 Дана очередь вещественных чисел. Удалить из очереди числа из заданного пользователем диапазона.

25 Дана очередь с вещественными числами, упорядоченными по убыванию. Добавить в очередь среднее арифметическое элементов очереди, не нарушая упорядоченности.

26 Дана очередь символов. Преобразовать очередь, оставив в нем из группы подряд идущих одинаковых элементов только один.

27 Дана очередь целых чисел. Удалить все локальные минимумы, при этом первый и последний элемент не обрабатывать.

28 Дан набор из чисел. Создать две очереди: первая должна содержать числа из исходного набора с нечетными номерами (1, 3, ...), а вторая — с четными (2, 4, ...).

29 Разделить созданную очередь целых чисел на две: в первую поместить положительные числа, во вторую – отрицательные.

30 Дана очередь целых чисел. Необходимо поменять местами крайние элементы.

31 Дана очередь целых чисел. Необходимо удалить элементы, заканчивающиеся на цифру 5.

32 Дана очередь вещественных чисел. Необходимо поменять местами элементы, содержащие максимальное и минимальное значения.

33 Дана очередь вещественных чисел. Перенести в новую очередь все элементы, находящиеся между первым и максимальным элементом.

34 Дана очередь целых чисел. Перенести в новую очередь все элементы, находящиеся между первым и минимальным элементом.

35 Дана очередь вещественных чисел. необходимо определить количество и удалить все элементы, находящиеся между минимальным и максимальным элементами.

36 Дана очередь целых чисел. Определить количество элементов, меньших среднего арифметического значения всех четных элементов, и удалить эти элементы.

37 Дана очередь целых чисел. Необходимо вычислить среднее арифметическое элементов, стоящих на четных позициях, и заменить им последний элемент.

38 Дана очередь вещественных чисел. Необходимо вычислить среднее арифметическое и заменить им все значения элементов, стоящих на нечетных позициях.

39 Дана очередь символов. Необходимо заменить все цифры на последний символ в очереди.

40 Даны две упорядоченные очереди вещественных чисел. Необходимо объединить эти очереди в одну, сохранив упорядоченность.

II. Разработать программу согласно варианту задания. Необходимо реализовать следующие функции для работы со стеком динамического хранения (последовательного или связного):

- создание стека;
- добавление элемента в стек;
- удаление элемента из стека;
- вывод стека на экран;
- очистка стека.

Используя разработанные функции выполнить обработку N стеков (число N вводит пользователь).

Варианты заданий:

1 Дан стек вещественных чисел. Удалить числа, меньшие среднего арифметического.

2 Дан стек целых чисел. Удалить наименьший элемент стека.

3 Дан стек символов. Удалить каждый нечетный элемент стека.

4 Дан стек целых чисел. Удалить все нечетные числа из стека.

5 Дан стек вещественных чисел. Удалить из стека числа из заданного пользователем диапазона.

6 Дан стек вещественных чисел, упорядоченных по убыванию. Добавить в стек среднее арифметическое элементов, не нарушая упорядоченности.

7 Дан стек символов. Преобразовать стек, оставив в нем из группы подряд идущих одинаковых элементов только один.

8 Дан стек целых чисел. Удалить все локальные минимумы, при этом первый и последний элемент не обрабатывать.

9 Дан набор из чисел. Создать два стека: первый должен содержать числа из исходного набора с нечетными номерами (1, 3, ...), а второй — с четными (2, 4, ...).

10 Разделить созданный стек вещественных чисел на два: в первый – положительные числа, во второй – отрицательные.

11 В созданном стеке символов поменять местами крайние элементы.

12 Из стека целых чисел удалить элементы, заканчивающиеся на цифру 5.

13 В стеке вещественных чисел поменять местами элементы, содержащие максимальное и минимальное значения.

14 Перенести из созданного стека целых чисел в новый стек все элементы, находящиеся между первым и максимальным элементом.

15 Перенести из созданного стека вещественных чисел в новый стек все элементы, находящиеся между первым и минимальным элементом.

16 В стеке целых чисел определить количество и удалить все элементы, находящиеся между минимальным и максимальным элементами.

17 В стеке вещественных чисел определить количество элементов, имеющих значения, меньше среднего арифметического значения всех элементов, и удалить эти элементы.

18 В стеке вещественных чисел вычислить среднее арифметическое и заменить им первый элемент.

19 В стеке вещественных чисел вычислить среднее арифметическое и заменить им все четные значения элементов.

20 В стеке целых чисел заменить все положительные цифры, отрицательными. И наоборот.

21 Объединить два целочисленных стека в новый.

22 Удалить из стека целых чисел все четные числа.

23 Удалить из стека вещественных чисел все отрицательные числа.

24 Поменять местами крайние элементы стека целых чисел.

25 Поменять местами минимальный и максимальный элементы стека вещественных чисел.

26 Удалить из стека символов каждый второй элемент.

27 Удалить из стека целых чисел все элементы, расположенные до минимального элемента стека.

28 Поменять местами наибольший среди отрицательных и наименьший среди положительных элементов целочисленного стека.

29 Поместить максимальный элемент стека вещественных чисел на первую позицию.

30 Поменять местами минимальный и первый элементы стека целых чисел.

31 Поменять местами первый и последний элементы стека вещественных чисел.

32 Удалить первый и последний элементы стека целых чисел.

33 Удалить из стека вещественных чисел все элементы, расположенные между минимальным и максимальным элементами.

34 Удалить из стека целых чисел все элементы, расположенные после максимального элемента.

35 Найти среднее арифметическое значение элементов целочисленного стека и удалить элементы, меньшие среднего значения.

36 Найти среднее арифметическое значение элементов стека вещественных чисел. Поместить ближайший к среднему значению элемент стека на первую позицию.

37 Дан стек целых чисел. Удалить наибольший элемент стека.

38 Дан стек символов. Удалить каждый n-ый элемент стека.

39 Дан стек целых чисел. Удалить все четные числа из стека.

40 Дан стек символов. Удалить каждый второй четный элемент стека.

Защита лабораторной работы

Для защиты лабораторной работы студент предоставляет рабочую программу и исходный код. Показывает и рассказывает преподавателю о выполненном задании.

Если все требования, указанные в пункте «Ход работы» выполнены в полной мере, получены ответы на вопросы преподавателя и, при необходимости, сделано дополнительное задание, то студент получает 60 баллов.

За невыполнение задания в полной мере преподаватель отнимает баллы:

- 10 баллов за каждую ошибку в программе;
- 10 баллов за невыполнение одного из требований лабораторной работы;
- 5 баллов за неполный ответ на вопрос;
- 3 балла за каждый недочет в работе программы.

В случае, если студент применил творческий подход к выполнению работы, преподаватель может поставить до 20 бонусных баллов.

Лабораторная работа считается не защищенной в случаях, если студент:

- не разбирается в коде программы;
- не может ответить на вопросы преподавателя по работе программы;
- не выполняет дополнительное задание (схожее с уже написанным);
- пытается сдать чужой вариант.

В таких случаях преподаватель выдает *новый вариант* и отнимает до *30 баллов* рейтинга.