National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

Department of Computing

CS212: Object Oriented Programming

Class: BSCS-5AB

Lab 6: Operator Overloading

Date: 9-04-2015

Time: 09:00 am – 12:00 pm / 2 – 5 pm

Instructor: Ms. Hirra Anwar

Lab 6: Operator Overloading

Introduction

Operator overloading in C++ allows us to use some of the existing C++ operators with user defined classes.

Objectives

To understand the concept of operator overloading by implementing unary operator overloading.

Tools/Software Requirement

You will need Visual Studio.

Description

C++ allows you to specify more than one definition for a function name or an operator in the same scope, which is called function overloading and operator overloading respectively. An overloaded declaration is a declaration that had been declared with the same name as a previously declared declaration in the same scope, except that both declarations have different arguments and obviously different definition (implementation).

When you call an overloaded function or operator, the compiler determines the most appropriate definition to use by comparing the argument types you used to call the function or operator with the parameter types specified in the definitions. The process of selecting the most appropriate overloaded function or operator is called overload resolution.

Function overloading in C++:

You can have multiple definitions for the same function name in the same scope. The definition of the function must differ from each other by the types and/or the number of arguments in the argument list. You cannot overload function declarations that differ only by return type.

Following is the example where same function print() is being used to print different data types:

```
#include <iostream>

using namespace std;


class printData
```

```
{
   public:
      void print(int i) {
        cout << "Printing int: " << i << endl;
      }
   void print(double  f) {
        cout << "Printing float: " << f << endl;
      }
    void print(char* c) {
        cout << "Printing character: " << c << endl;
      }
};
int main(void)
{
   printData pd;

   // Call print to print integer
   pd.print(5);
   // Call print to print float
   pd.print(500.263);
   // Call print to print character
   pd.print("Hello C++");

   return 0;
}
```

When the above code is compiled and executed, it produces the following result:

```
Printing int: 5
Printing float: 500.263
Printing character: Hello C++
```

Lab Tasks

Write your functions using the concepts of passing by reference.

Lab Task 1

Create a class Date and overload the ++ and – operators to increment or decrement a particular date by day.

Lab Task 2

Re-implement Date class, this time instead of incrementing by day overload the operator to increment the Date object by month.

Lab Task 3

Create a Distance class which has which has feet and inches. Overload the ++ and – operators to increment and decrement a distance object.

Lab Task 4

Find the size of Date & Distance classes' objects.

Deliverables

Source code of all the tasks with your names & reg. no along with comments in all programs should be shown to the lab engineer and upload on LMS.