**National University of Computer and Emerging Sciences **

**Lab Manual**

**Object Oriented Programming**

| Course Instructor | Ms. Saira Karim |
| --- | --- |
| Lab Instructor (s) | Ms. Mamoona Akbar Ms. Sonia Anum |
| Section | BCS-2B |
| Semester | Fall 2022 |

Department of Computer Science

FAST-NU, Lahore, Pakistan

**Lab Manual 7**

**Objectives:**

Operator overloading using non-member functions

Overloading assignment operator

Static data member and member function

Concept of composition

**TASK 1:**

A polynomial P1(x) = x4+2x2+5 has three terms: x4, 2x2and 5. Coefficients of these terms are 1, 2 and 5 respectively while exponents are 4, 2 and 0 respectively. To work with Polynomials, a definition of class Polynomial is given below and memory configuration for P1 is shown as follows:

| **class Polynomial**  **{**  **private:**  int totalTerms;//Total terms in a Polynomial int\* coeff;//to save array of coefficients int\* exp; //to save array of exponents **};** | totalTerms 3  coeff 1 2 5  exp 4 2 0  **P1(x)** |
| --- | --- |

**Your task is to complete the definition of Polynomial class such that the main program runs successfully.** Make sure that your program doesn’t consume extra memory space and it should not leak any memory.

| **void main()**  **{**  int coeff\_P1[] = {1,2,5}; //Coefficients for Polynomial P1  int exp\_P1[] = {4,2,0}; //Exponents for Polynomial P1  int coeff\_P2[] = {4,3}; //Coefficients for Polynomial P2  int exp\_P2[] = {6,2}; //Exponents for Polynomial P2  Polynomial P1(3, coeff\_P1, exp\_P1);//Creates P1 with 3 terms (P1 = 1x^4 + 2x^2 + 5x^0 )  Polynomial P2(2, coeff\_P2, exp\_P2);//Creates P2 with 2 terms (P2 = 4x^6 + 3x^2)  cout<<"P1 = "<<P1<<endl; //Prints P1 = x^4+2x^2+5  cout<<"P2 = "<<P2<<endl; //Prints P2 = 4x^6+3x^2  if(!P1)  cout<<”P1 is zero”<<endl; /\*if polynomial has only 1 term and its coeff and exp are zero. i.e. if p1 = 0.\*/  if(P1 != P2)  cout<<”P1 is Not Equal to P2”<<endl; |
| --- |

| cout<<++P1<<endl; //adds 1 in all the coefficient.  cout<<P1<<endl;  cout<<P1++<<endl; //adds 1 in all the coefficient.  cout<<P1<<endl;  Polynomial P3 = P1+P2; //Adds P1 and P2 and saves result in P3.You may consume extra space for resultant Polynomial in Add function  cout<<"P3 = "<<P3<<endl; //Prints P3 = 4x^6+x^4+5x^2+5  P3 = 2 + P1; //Assume P1 already has a constant term, add 2 in it.  cout<<"P3 = "<<P3<<endl;  P4 = P1;  cout<<"P4 = "<<P4<<endl;  **}** |
| --- |

**TASK 2: (Static member and Function)**

Implement a class called **Box**. The Box class will have three data members: ∙ int length;

∙ int breadth;

∙ int height;

You have to implement the following:

1. Implement all getters/setters.

2. There should be a static data member

∙ static int objectCount; // Increases every time object is created 3. Write an overloaded and default constructor.

4. Write member functions as follow:

∙ static int getCount();

∙ double Volume();

∙ double Area();

5. Write a suitable main() function to test the functionality of the static members and functions.

**TASK 3: (Composition)**

a) Define and implement a class Point in files Point.h and Point.cpp, respectively. This class should provide:

∙ Two private integer data members x and y which will store the x and y coordinates of a point

∙ A default constructor which takes two parameters to initialize the x and y coordinates and prints “Point() called” on the screen.

∙ A function print() which prints out the point on the screen in the format (x,y) ∙ A destructor which prints “~Point() called” on the screen.

b) Now define and implement a class Circle in files Circle.h and Circle.cpp. This class should contain:

∙ A private data member center which will be an instance of the Point class ∙ A private float data member radius that will store the radius of the circle ∙ A constructor which takes three parameters (x and y coordinates of the center of the

circle, and the radius) and initializes the data members accordingly and also prints “Circle() called” on the screen.

∙ A destructor which prints “~Circle() called” on the screen.

∙ A function print() which prints the information (center and radius) of the circle on the screen

To call the constructor of class Point from the constructor of class Circle, you can use the following syntax.

Circle::Circle(int x, int y, float r): center(x,y) { … };

Add another file Lab.cpp in your project. Copy the following piece of code in that file, compile and then execute. Note down the output of the program and write it in comments in the code.

#include "Circle.h"

void main()

{

Circle c (3,4,2.5);

c.print();

}