Assignment 7

Submission Date- 09/12/2024

Title – Study of constructor, operator overloading in C++.

Objective- 1. To learn Implementation of Constructor, operator overloading in C++.

Problem Statement-

Implement a class Quadratic that represents degree two polynomials i.e., polynomials of type ax2+bx+c. The class will require three data members corresponding to a, b and c. Implement the following operations:

1. A constructor (including a default constructor which creates the 0polynomial).
2. Overloaded operator+ to add two polynomials of degree2.
3. Overloaded << and >> to print and read polynomials. To do this, you will need to decide what you want your input and output format to look like.
4. A function eval that computes the value of a polynomial for a given value of x.
5. A function that computes the two solutions of the equationax2+bx+c=0.

Software & Hardware requirements- any Text editor and Terminal in Linux/ Turbo C++ Compiler installed on PC.

Theory-

Constructors:

Constructors are the special type of member function that initializes the object automatically when it is created Compiler identifies that the given member function

is a constructor by its name and return type. Constructor has same name as that of class and it does not have any return type.

Constructor can be overloaded in similar way as function overloading. Overloaded constructors have same name (name of the class) but different number of argument passed. Depending upon the number and type of argument passed, specific constructor is called. Since, constructor are called when object is created. Argument to the constructor also should be passed while creating object.

A object can be initialized with another object of same type.- Default Copy Constructor. If you want to initialize a object A3 so that it contains same value as A2.

```
Area A3 (A2);     /* Copies the content of A2 to A3 */
                        OR,
Area A3=A2;       /* Copies the content of A2 to A3 */
```

Operator Overloading:

The meaning of operators are already defined and fixed for basic types like: int, float, double etc in C++ language. For example: If you want to add two integers then, + operator is used. But, for user-defined types (like: objects), you can define the meaning of operator, i.e, you can redefine the way that operator works. For example: If there are two objects of a class that contain string as its data member, you can use + operator to concatenate two strings. Suppose, instead of strings if that class contains integer data member, then you can use + operator to add integers. This

feature in C++ programming that allows programmer to redefine the meaning of operator when they operate on class objects is known as operator overloading.

To overload a operator, a operator function is defined inside a class as:

```
class class_name
{
........ ...... ......
public:
   return_type operator sign (argument/s)
   {
    ...... ..... ......
   }
........ ...... ......
};
```

The return type comes first which is followed by keyword **operator**, followed by operator sign, i.e., the operator you want to overload like: +, <, ++ etc. and finally the arguments is passed. Then, inside the body of you want perform the task you want when this operator function is called. This operator function is called when, the operator (sign) operates on the object of that class class_name.

Code:

```
#include <iostream>
#include <cmath>
using namespace std;
class Quadratic
{
   public:
      int a,b,c,s;
      Quadratic()
      {
            a=0;
```

```
        b=0;
        c=0;
    }

    Quadratic operator+(Quadratic &d)
    {
        Quadratic c4;
        c4.a=a+d.a;
        c4.b=b+d.b;
        c4.c=c+d.c;
        return c4;
}
    friend void operator<<(ostream &o,Quadratic &e)
    {
        o<<e.a<<"x\u00B2"<<"+"<<e.b<<"x"<<"+"<<e.c;
    }
    friend void operator>>(istream &i,Quadratic &q)
        {
            i>>q.a>>q.b>>q.c;
        }
    void eval(int se)
    {
      s=(a*se*se)+(b*se)+c;
      cout<<"The result of polynomial for a given x: "<<s;
    }
    int sol()
    {
```

```
            Cout <<"\n The roots of quadratic equation of polynomial is: ";
            Cout <<"\nRoot1: "<<((-b)+(sqrt(b*b-4*a*c)))/(2*a);
        cout<<"\n Root2: "<<((-b)-(sqrt(b*b-4*a*c)))/(2*a);
    }
};

int main() {
    Quadratic c1,c2,c3,c6;
    int x;
    cout<<"Initially polynomial is: ";
    cout<<c1;
    cout << "\n Enter first polynomial: " << endl;
    cin>>c1;
    cout << "Enter second polynomial: " << endl;
     cin>>c2;
    c3=c1+c2;
  cout << "First polynomial is: " << endl;
  cout<<c1;
  cout << "\n Second polynomial is: " << endl;
  cout<<c2;
cout << "\n Addition of two polynomial: " << endl;
  cout<<c3;
  cout << "\n Enter the value of x to evaluate polynomial 1: " << endl;
  cin>>x;
  c1.eval(x);
  c1.sol();
  cout << "\n Enter the value of x to evaluate polynomial 2: " << endl;
```

```
    cin>>x;
    c2.eval(x);
    c2.sol();
    return 0;
}
```

O/P- Execute a code and write Out Put.

Conclusion-

Well Understanding concept of constructor, operator overloading and its implementation in C++.