Laboratory Work No. 7

Object Composition

This laboratory work covers the following concepts in C++ programming language:

- class declaration (access specifiers: public, private)
- data members, member functions, constructors, destructor
- object composition

⇨ Create a Win32 Console application and an empty C++ source file in Visual Studio IDE to be able to start typing programs.

Task-1: In Cartesian coordinate system, a linear equation represents a line passing through two discrete points, such as, (1, 1) and (4, 2), respectively. Hence, one can easily describe the "point" object with a C++ class as given below.

```
class Point {
    public:
        Point();                    //default constructor
        Point(…);                   //parameterized constructor
        Point(…);                   //copy constructor
        ~Point();                   //destructor
        … set_x(…);                 //some member functions
        … set_y(…);
        … get_x(…);
        … get_y(…);
        … print();                  //prints the point object using some format, e.g., [1,2]
    private:
        int x, y;                   //data members
};
```

On the other hand, in order to create a "line" object, one can think of executing data composition technique where the line object is composed of two discrete point objects. Hence, one can write the following:

```
class Line {
    public:
        Line();                     //default constructor
        Line(…);                    //parameterized constructor
        Line(…);                    //copy constructor
        ~Line();                    //destructor
        … set_point1(…);            //some member functions
        … set_point2(…);
        … get_slope();              //calculates the slope value
        … print();                  //prints the line object using some format
                                    //e.g., A line passing through [2,2] and [4,4] with slope = 1.0
    private:
        Point p1, p2;               //data composition!
};
```

a. Write definitions of the member functions listed above.
b. Rewrite the program by separating the implementation file from the interface using a header file.

⇨ Provide a driver program to test each implementation.