EUROPEAN UNIVERSITY OF LEFKE
Faculty of Engineering
Department of Computer Engineering

COMP218
OBJECT-ORIENTED PROGRAMMING

LAB WORK NO. 7

Prepared by **David O. Ladipo** (174574)
Submitted to Dr. Ferhun Yorgancıoğlu

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.

a. Write definitions of the member functions listed above.

```
#include <iostream>
using namespace std;

class Point{
  private:
      //data members
    int x1,y1;
  public:
      //default constructor
    Point(){}
    //parameterized constructor
    Point(int x_1=1,int y_1=1){
      set_x1(x_1);
      set_y1(y_1);
    }
    // copy constructor
    Point(const Point &copy):x1(copy.x1),y1(copy.y1){}
    // destructor
    ~Point(){}
    // set functions
    void set_x1(int x1){this->x1 = x1;}
    void set_y1(int y1){this->y1 = y1;}
    //get functions
    int get_x1()const{return x1;}
    int get_y1()const{return y1;}
    ////prints the point object using some format, e.g., [1,2]
    void print(){
      cout<<"["<<x1<<","<<y1<<"]"<<endl;
    }
};


class Line{
    // data composition
    Point p1;
    Point p2;
  public:
    // copy constructor
    Line(const Point& P1, const Point& P2): p1(P1), p2(P2){
```

```
   //   set_point1(); // calls the set point function
    //  set_point2();
    }
    // Destructor
    ~Line(){}

    // set functions of data composition, this set function could be called from
the copy constructor and set the values passed, but i preferred to pass two point
object from the main function.
    void set_point1(){
      p1.set_x1(5);
      p1.set_y1(20);
    }
    void set_point2(){
      p2.set_x1(10);
      p2.set_y1(50);
    }
    // calculates the slope value for each data composite member
    void get_slope(){
      int a,x,m,x1,x2,y1,y2;
      x1 = p1.get_x1();
      x2 = p2.get_x1();
      y1 = p1.get_y1();
      y2 = p2.get_y1();
      m=(y2 - y1);
      x= (x2 - x1);
      a = m/x;
      cout << a << endl;
    }
    void print(){
      //e.g., A line passing through [2,2] and [4,4] with slope = 1.0
      cout<<"A line passing through ";
      p1.print();
      cout << "and ";
      p2.print();
      cout<<"with slope = ";
      get_slope();
          }
};

int main() {
    // Prints out the main point object with 2 and 5 passed as parameters
  cout << "This is the main point object: " << endl;
  cout << "Point 1: ";
  Point mp(2,5);
```

```
  mp.print();
  cout << "Point 2: ";
  Point mp2(5,8);
  mp2.print();
  cout << "****************** " <<endl;
  cout<< "This is the Line object of two point objects passed as parameters: " <<
endl;
  // Line object initialized and i passed the two point object
  //created as parameters to the Line object
  Line lne(mp, mp2);
  lne.print();

}
```

"C:\Users\David\Desktop\OOP C++ Work\LAB-7\Lab-7-Task-1\bin\Debug\Lab-7-Task-1.exe"

```
This is the main point object:
Point 1: [2,5]
Point 2: [5,8]
******************
This is the Line object of two point objects passed as parameters:
A line passing through [2,5]
and [5,8]
with slope = 1

Process returned 0 (0x0)   execution time : 0.231 s
Press any key to continue.
```

b. Rewrite the program by separating the implementation file from the interface using a header file.

main.cpp

```
#include <iostream>
#include "Line.h"
#include "point.h"
using namespace std;

int main()
{
    // Prints out the main point object with 2 and 5 passed as parameters
  cout << "This is the main point object: " << endl;
```

```
  cout << "Point 1: ";
  Point mp(2,5);
  mp.print();
  cout << "Point 2: ";
  Point mp2(5,8);
  mp2.print();
  cout << "****************** " <<endl;
  cout<< "This is the Line object of two point objects passed as parameters: " <<
endl;
  // Line object initialized and i passed the two point object
  //created as parameters to the Line object
  Line lne(mp, mp2);
  lne.print();
    return 0;
}
```

Line.h

```
#ifndef LINE_H
#define LINE_H

#include "point.h"


class Line{
    // data composition
    Point p1;
    Point p2;
  public:
    // copy constructor
    Line(const Point& P1, const Point& P2);
    // Destructor
    ~Line();

    // set functions of data composition, This set function could be called from
    //the copy constructor and set the values passed, but i preferred to pass two
    //point object from the main function.
    void set_point1();
    void set_point2();


    // calculates the slope value for each data composite member
    void get_slope();
    void print();
```

```
};

#endif // LINE_H
```

point.h

```
#ifndef POINT_H
#define POINT_H


class Point{
  private:
      //data members
    int x1,y1;
  public:
      //default constructor
    Point();
    //parameterized constructor
    Point(int,int);
    // copy constructor
    Point(const Point &copy);
    // destructor
    ~Point();
    // set functions
    void set_x1(int);
    void set_y1(int);
    //get functions
    int get_x1();
    int get_y1();
    ////prints the point object using some format, e.g., [1,2]
    void print();
};

#endif // POINT_H
```

Line.cpp

```
#include <iostream>
#include "point.h"

#include "Line.h"
using namespace std;

Line:: Line(const Point& P1, const Point& P2): p1(P1), p2(P2){
   //   set_point1(); // calls the set point function
```

```
    //  set_point2();
    }
Line::~Line(){}
void Line:: set_point1(){
      p1.set_x1(5);
      p1.set_y1(20);
    }
void Line:: set_point2(){
      p2.set_x1(10);
      p2.set_y1(50);
    }
 void Line:: get_slope(){
      int a,x,m,x1,x2,y1,y2;
      x1 = p1.get_x1();
      x2 = p2.get_x1();
      y1 = p1.get_y1();
      y2 = p2.get_y1();
      m=(y2 - y1);
      x= (x2 - x1);
      a = m/x;
      cout << a << endl;
    }
void Line:: print(){
      //e.g., A line passing through [2,2] and [4,4] with slope = 1.0
      cout<<"A line passing through ";
      p1.print();
      cout << "and ";
      p2.print();
      cout<<"with slope = ";
      get_slope();
          }
```

point.cpp

```
#include <iostream>
#include "point.h"
using namespace std;

Point::Point()
{
}
Point:: Point(int x_1=1,int y_1=1){
      set_x1(x_1);
      set_y1(y_1);
    }
```

```
Point:: Point(const Point &copy):x1(copy.x1),y1(copy.y1){}

Point:: ~Point(){}

void Point:: set_x1(int x1){this->x1 = x1;}
void Point::  set_y1(int y1){this->y1 = y1;}

Point:: get_x1(){return x1;}

Point:: get_y1(){return y1;}
    ////prints the point object using some format, e.g., [1,2]
void Point:: print(){
      cout<<"["<<x1<<","<<y1<<"]"<<endl;
    }
```

"C:\Users\David\Desktop\OOP C++ Work\LAB-7\Lab-7-Task-2\bin\Debug\Lab-7-Task-2.exe"

```
This is the main point object:
Point 1: [2,5]
Point 2: [5,8]
*******************
This is the Line object of two point objects passed as parameters:
A line passing through [2,5]
and [5,8]
with slope = 1

Process returned 0 (0x0)   execution time : 0.218 s
Press any key to continue.
```