Object Oriented Programming Lab

SPRING - 2024

LAB 10

FAST National University of

Computer and Emerging Sciences

Learning Outcomes

In this lab you are expected to learn the following:

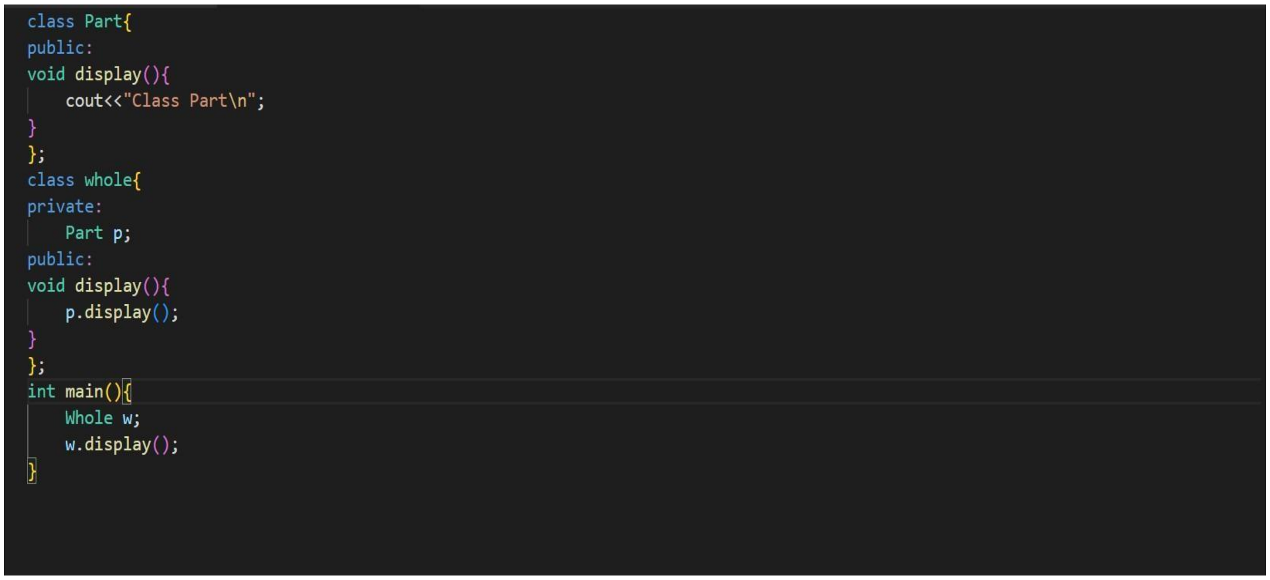
- Composition and Aggregation

Composition:

Composition is a relationship between two classes in which one class, known as the composite class, contains an object of another class, as a member variable. Composition models "part-of" relationships. These relationships are part-whole relationships. Composition is often used to model physical relationships, where one object is physically contained inside another.

- Heart is a part-of body
- Fish is a part-of pond

For example,



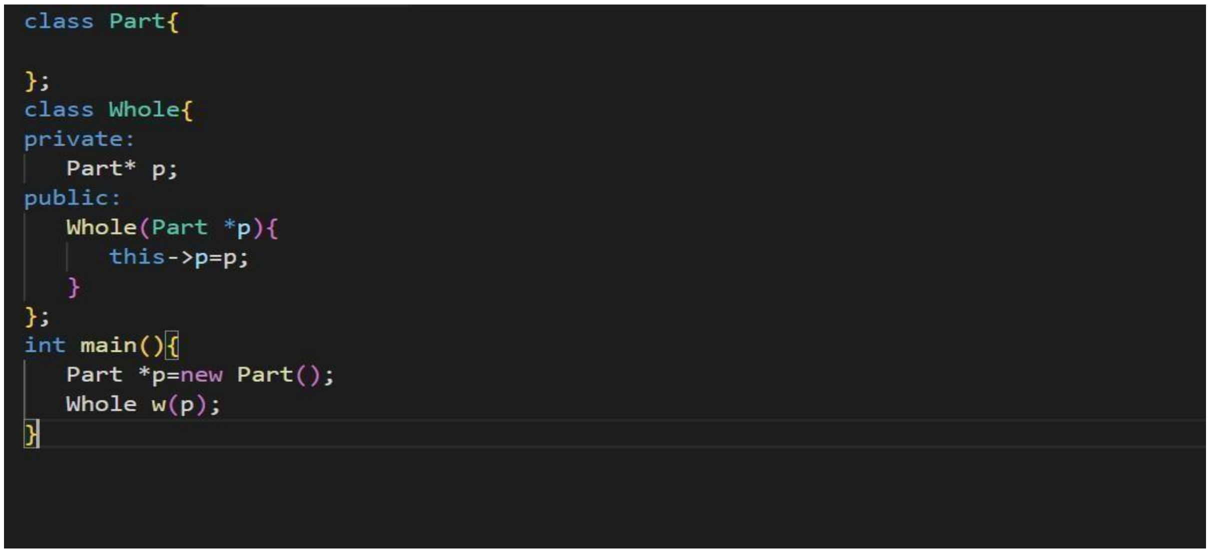
```
class Part{
public:
void display(){
    cout<<"Class Part\n";
}
};
class whole{
private:
    Part p;
public:
void display(){
    p.display();
}
};
int main(){
    Whole w;
    w.display();
}
```

Aggregation:

Aggregation is a relationship between two classes in which one class, known as the aggregate class, contains a pointer or reference to an object of another class. Aggregation is also a part-whole relationship. It models has-a relationship. Parts can belong to more than one object at a time. The whole is not responsible for the creation and deletion of parts.

- Every person has an address

For Example,



```cpp
class Part{

};
class Whole{
private:
    Part* p;
public:
    Whole(Part *p){
        this->p=p;
    }
};
int main(){
    Part *p=new Part();
    Whole w(p);
}
```

Lab Tasks

Submission Instructions:

1. Create a new folder with name *ROLLNO_SEC_LAB10* **e.g. i23XXXX_A_LAB10**
2. Move all of your **.cpp and .h files** to this newly created directory and compress it into a **.zip file**.
3. Now you have to submit this zipped file on Google Classroom.
4. If you don't follow the above-mentioned submission instruction, you will be marked **zero.**
5. Plagiarism in the Lab Task will result in **zero** marks in the whole category.

Q1.

Write a class **Point** that has the following data members.

- **X_Coordinate**: x coordinate of type integer
- **Y_Cooridnate**: y coordinate of type integer

The Point class has following member functions

1. A default constructor that initializes the data members to zero.
   Point()

2. A parameterized constructor that accepts the parameters for each member variable.
   Point(int , int)

3. A copy constructor that takes a previously constructed object as an argument.
   Point(const Point &p)

4. Write accessors for each data member.
   int getX_Coordinate() const

   int getY_Coordinate() const

Write a class **Line** that represents a line segment between two Points hence it composes **Point** class.

The Line class has the following data members.

Point_1: a point P1 of type Point
Point_2: a point P2 of type Point

The Line class has the following member functions.
Note*: Use member initializer list for all constructors

1. A default constructor that initializes the coordinates of 2 points to 1,5 and 10, 15.

Line()

2. A parameterized constructor.

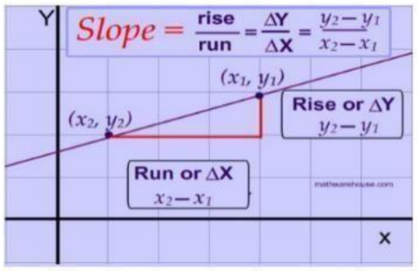
   Line(int x1, int y1, int x2, int y2)

3. A copy constructor that takes two previously constructed Point objects as argument.

   Line(const Point &p1, const Point &p2)

4. A member function findSlope that returns the slope of the length.

   float findSlope()


Y
$Slope = \frac{\text{rise}}{\text{run}} = \frac{\Delta Y}{\Delta X} = \frac{y_2 - y_1}{x_2 - x_1}$
(x_1, y_1)
(x_2, y_2)
Rise or ΔY
$y_2 - y_1$
Run or ΔX
$x_2 - x_1$
mathwarehouse.com
X

5. A member function findLength that returns the length of the line segment using distance formula.

   float findLength()

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

6. A member function findMidPoint that returns the midpoint of the line segment.

   Point& findMidPoint()

Midpoint Formula:

The midpoint (x,y) of a segment with endpoints (x_1, y_1) and (x_2, y_2) has coordinates

$$(x, y) = \left(\frac{x_1 + x_2}{2}, \frac{y_1 + y_2}{2} \right)$$

Output:

1. **Example**
   Use Parameterized Constructor of class Line to assign the values
   - x1=1, y1=5, x2=10, y2=15
   - Slope= 1.11
   - Length= 13.45
   - Mid Point= (5,10)

Q2.

A university owns a department (e.g., CS, Electrical Engineering), and each department has one professor. If the university closes, the department will no longer exist, but the professor in those departments will continue to exist. Therefore, a University can be seen as a composition of a department, whereas a department has an aggregation of a professor. In addition, a Professor could work in more than one department, but a department could not be part of more than one university.

Write a Class named **Professor** having the following attributes:
- name of type string
- employeeID of type int
- designation of type string

Write a Class named **Department** having the following attributes:
- name of type string
- deptID of type int
- pointer to object of type professor

Write a Class named **University** having following attributes:
- name of type string
- pointer to object of type Department

Write following functions.

1. Write appropriate getter setter of each data member for each Class
2. Add/delete/update Department in University class

   - bool addDepartment(Department D)

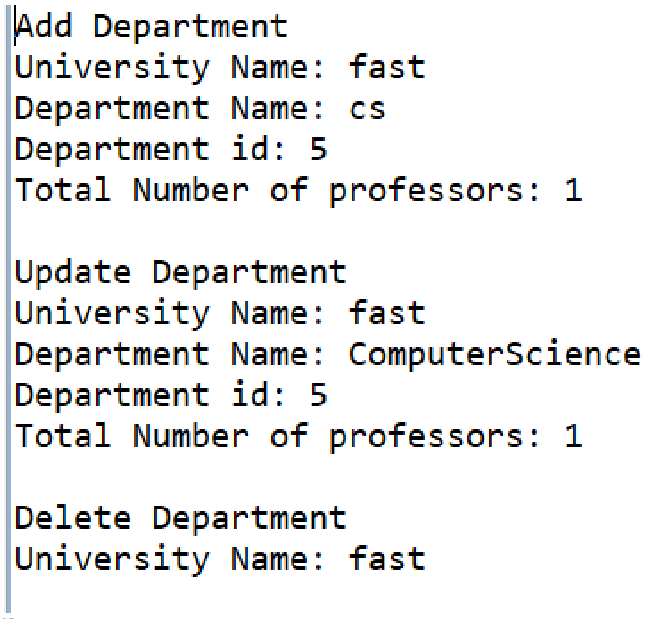
   - bool deleteDepartment(string name)

   - bool updateDepartment(int id, string name) //Update name of department given department id.

3. void Display() function to display university information. Also, display department information in this function.

4. Add/delete/update Professor in Department class
    - bool addProfessor(Professor *p)
    - bool deleteProfessor (int id)
    - bool updateProfessor (int id, string newDesignation) //Update designation of the professor given employee id
5. void Display() function to display department information. Also display professors information in this function.

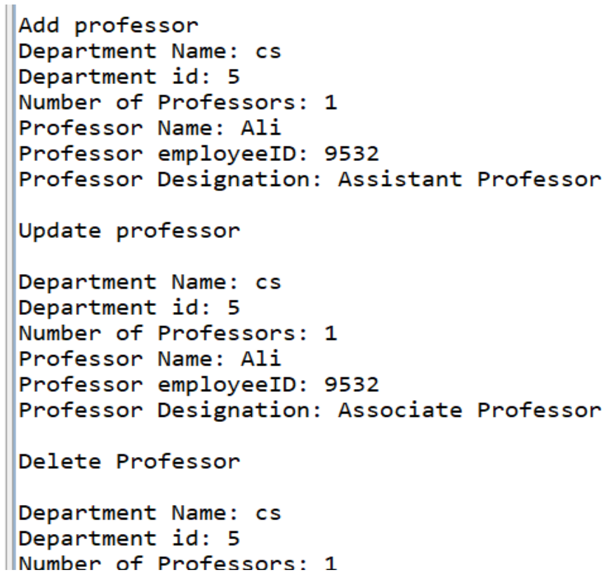
Output:



```
Add Department
University Name: fast
Department Name: cs
Department id: 5
Total Number of professors: 1

Update Department
University Name: fast
Department Name: ComputerScience
Department id: 5
Total Number of professors: 1

Delete Department
University Name: fast
```



```
Add professor
Department Name: cs
Department id: 5
Number of Professors: 1
Professor Name: Ali
Professor employeeID: 9532
Professor Designation: Assistant Professor

Update professor

Department Name: cs
Department id: 5
Number of Professors: 1
Professor Name: Ali
Professor employeeID: 9532
Professor Designation: Associate Professor

Delete Professor

Department Name: cs
Department id: 5
Number of Professors: 1
```