National University
of computer and emerging sciences

CL-1004

Object Oriented Programming- Lab

Spring' 2023

BS-SE

Lab Manual 07

Task 1

You are required to create a class called Laptop which represents a Laptop. This class should include four pieces of information as instance variables:

- Laptop model number (String).
- MAC Address (String).
- Company name (String).

To initialize the instance variables, you need to create a constructor. You should also provide get and set methods for each instance variable. Additionally, you should create a method called getLaptopInfo that returns the description of the laptop as a string, including all the information about the laptop.

String getLaptopInfo()

In your implementation, use the "this" keyword in member methods and constructors. Also create two constructors: a parameterized constructor and a non-parameterized constructor. Call the parameterized constructor from the non-argument constructor and pass some arguments.

Task 2

1. Write a class Course that includes the following members:

Data Members	Properties Description
courseCode	Stores course code, like “CS 103”
courseTitle	Stores course title, “Computer Programming"
creditHours	Stores credit hours of the course e.g. 1,2,3 & 4.
section	Stores section in which student is registered. E.g. A, B, C etc
repeatCount	Stores repeat count of the course for the student. Like 1,2,3

Member Functions:

a. Getters and setters for each data members.

b. Default constructor

c. Copy constructor

Course (Course & c)

d. Parameterized constructor

Course (string cc, string ct, int ch, char s, int rc)

2. Write a class Semester that includes the following members:

Data Members	Properties Description
semesterCode	Stores the code of semester, like “Spring 2018"
courseCount	Stores the number of courses registered between one and five courses.
courses pointer of Course Class	Course array (dynamically created using courseCount)

Member Functions:

a. Getters and setters for each data members.

b. Default constructor.

c. Copy constructor

Semester (Semester & s)

d. Parametrized constructor

Semester (string sc,int c, Course *courseArr)

After defining the classes, write the following global functions in Semester.cpp file:

1. **int GetCreditHoursCount(Semester sem)** : receives a semester as parameter and returns the total number of credit hours registered in it.

3. **bool FindCourseInSemesterRegistration(Semester sem, string sc) :** receives a semester, and a course code as parameters and returns true if the course is registered in the semester.

Task 3

Write a class called Line that represents a line segment between two Points. Your Line objects should have the following methods:

- Line(Point &p1, Point &p2)

Constructs a new Line that contains the given two Points.

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

Constructs a new Line that contains the given two Points.

- Line(Line &copy) // it's a copy constructor

Constructs a new Line form given line.

- Point getP1()

Returns this Line's first endpoint.

- Point getP2()

Returns this Line's second endpoint.

- double getSlope()

Returns the slope of this Line. Remember The slope of a line between points (x1, y1) and (x2, y2) is equal to (y2-y1)/(x2-x1).

- String toString()

Returns a String representation of this Line, such as "[(22, 3), (4,7)]" .

Hint: create a separate Point class with int x, and int y data members with constructor and getter setter functions.

Task 4:

You are required to design a class that represents a bank account. The only two instance data members for this class are the accountNumber(long) and balance(double). Although the balance can be initialized by the user of the class during creation of an instance, the account number cannot be entered by the user because it must be unique.

To prevent duplication of the account number, you need to use a static data member, called base(int), that is initialized to 0 and incremented with the opening of each new account. We add a static member function because we need to access it during increment.

i) Account() // Default Constructor

Initializes balance to zero and update the accountNumber according to given instructions

ii) Account(double bal) // Parameterized Constructor

If bal is negative, program must terminate with a message otherwise update balance and accountNumber accordingly

iii) void checkBalance () const; // Accessor

Display account details and balance

iv) void deposit (double amount); // Mutator

amount > 0.0 to perform this transaction, and display account details afterwards

v) void withdraw (double amount); // Mutator

balance >= amount to perfrom this transaction, and display account details afterwards

vi) ~Account()

Display account is closed with account details and a message that balance is transferred to the authorized person.