Object Oriented Programming Lab

SPRING - 2024

LAB 07

FAST National University of
Computer and Emerging Sciences

Learning Outcomes

In this lab you are expected to learn the following:

- Static Data Members
- Copy Constructor

Static Data Member:

Static data members are class members that are declared using **static** keyword. Only one copy of that member is created for the entire class and is shared by all the objects of that class, no matter how many objects are created.

Syntax: static data_type data_member_name;

For example,

```
class Room{
    private:
        int length;
        int height;
    public:
    static int objectCount;
    Room(){
        objectCount++;
    }
};
int Room::objectCount=0;
```

Here, we defined a class named **Room**.
The variables **length**, **breadth**, and **height** declared inside the class are known as **data members**. And, the objectCount data member is a static data member that contains the number of objects created of class Room. Room() is a constructor that increments objectCount each time a new class object is created.

Copy Constructors:

A **copy constructor** is a member function that initializes an object using another object of the same class.
For Example,

```
class Room{
    private:
        int length;
        int height;
    public:
    Room(int l,int h){
        length=l;
        height=h;
    }
    Room(Room &r){
        length=r.length;
        height=h.height;
    }
};
int main(){
    Room r1(5,6);
    Room r2(r1);
}
```

Lab Tasks

Submission Instructions:

1. Create a new folder with name *ROLLNO_SEC_LAB01* **e.g. i22XXXX_A_LAB07**
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. Define a class **Circle** with the following private data members:

- int radius
- static int countOfCircles (You also need to maintain a count of circles being created)

Then implement member functions;

1. Write a default constructor
2. Write a parameterized constructor
3. Create setter and getter methods for all data members.
4. Static Member function for countOfBlocks.
    - **static int getcountOfCircles()**
5. Write a function to compute area of circle.
    - **double getArea()**
    - Area of Circle: 3.14 * radius * radius
6. Write a function to compute the perimeter of circle.
    - **double getPerimeter()**
    - Perimeter of Circle: 2 * 3.14 * radius.
7. Write a destructor that will decrement static countOfCircles.
8. Print **countOfCircle in main**

Output:

- CountOfCircles = 0

1. **Example 1:**

   Create a pointer to object type Circle and assign values using a parametrized constructor.

   - Radius=6
   - Area=113.04
   - Perimeter=37.68

2. **Example 2:**

   Create a pointer to object type Circle and use Setter to assign values

   - Radius=10
   - Area=314
   - Perimeter=62.8

3. **Example 3:**

   Create a pointer to object type Circle and assign values using a parametrized constructor.

   - Radius=12.5
   - Area=490.625
   - Perimeter=78.5

- CountOfCircles = 3
- Deallocate memory and delete all objects
- CountOfCircles = 0

Q 2. Write a class **Matrix** with the following private data members:

- int rows
- int columns
- int **arr

Then implement member functions;

1. Write a default constructor
1. Overloaded constructors
   - Matrix(int r, int c, int** p) // parameterized constructor
   - Matrix(Matrix &copy) // copy constructor
2. Create setter and getter methods for all data members
3. Write following member function in the class

- **void multiplyMatrix(int)**

  It perform scalar multiplication which takes integer as argument and multiple it with each element of matrix.

- **void addMatrix(int)**

  It perform scalar additon which takes integer as argument and add it with each element of matrix.

- **void display()**

  It prints the Matrix.

Output:

1. **Example 1:**

   Use Copy Constructor to assign values

   - Initialize the value of Matrix

     Arr[3][3]={ {1,5,7}, {4,8,3}, {6,4,1}}

   - Resultant Matrix after calling function multiplyMatrix(4)

     Arr[3][3]={ {4,20,28}, {16,32,12}, {24,16,4}}

   - Resultant Matrix after calling function addMatrix(3)

     Arr[3][3]={ {4,8,10}, {7,11,6}, {9,7,2}}

Q 3.

Your goal in this question is to write a program to manage a ShoppingList. For this you will need to create a class ShoppingList that should be able to store all the details of shopping items with the following functionalities:

Your ShoppingList has a fixed capacity 10 of ShoppingItems. For each ShoppingItem, you will store its name, its quantity, and its price. A new ShoppingItem cannot be added to ShoppingList if the list is already full. Whenever you buy a new item you will add it to the ShoppingList if the capacity is not full.

Identify all data members of these classes and write them. Your classes must provide following member functions.

ShoppingList	
Member Function	**Description**
AddItem	Add a new shopping item to the shopping list.
Print	Should print the current list of added items to list
TotalCost	Should print the total cost of shopping list
ShoppingItem	
Member Function	**Description**
InputItem	Add details for the current item.
Display	Should display the information of item