FAST School of Computing

Object Oriented Programming – Spring 2023

Cyber Security Department

LAB 07

Recursion in C++

Learning Outcomes

In this lab you are expected to learn the following:

- Classes
  - ✗ Constructors (Default, Parameterized, Copy)
  - ✗ Static data members and methods
  - ✗ Constant functions
  - ✗ Initializer lists with constructors.
  - ✗ Member Functions and Non Member Functions

Note: Plagiarism(from some else or internet) in any 1 question will lead to zero marks in the whole lab task.

Problem 1:

Declare a class **Block.** A block as you all know is something a cubical container. It has following attributes

Length Width Height

In addition to these, a block can be made of different **materials** e.g. wood, card, metal etc. Further, more a block can have different **colors**. Declare them also as member variables of class. You also need to maintain a **count of Blocks** being created. So make **int static countofBlocks** as public.

The class has the following **member functions:**

1. Provide a **default Constructor**, a **parameterized Constructor** for the Block that takes all necessary values as arguments with the material as optional (if it is omitted the Block is considered to be made of **Card** – default value for the material.)

2. Provide **getters** for all attributes and **setters** for each too except for the material (material of block cannot be changed after when it has been created!!!). Hint: use const for material
3. Provide a function getVolume() that calculates and returns the value of the volume of the Block. **Volume=length × breadth × height.**

4. Also provide another function **getSurfaceArea()**, that calculates and returns the surface area of a Block. **surface area = 2*(l*w + l*h + w*h)**.

5. Provide a **print** function that Prints the following about the Block

Length Width, Height, Material, Color, Volume, Surface Area

6. Print **countofblocks** in main().

Problem 2:

Design a class Complex for handling Complex numbers and include the following **private** data members:

- **real:** a double
- **imaginary:** a double

The class has the following **member functions.**

1. A **constructor** initializing the number with **default parameters.**
2. Overloaded Constructors.
    - Complex(double r, double i) //Use initializer list
    - Complex(Complex & copy) // copy constructor

3. **Getters** and **Setters** of the class data members as given below
    - void setReal(double r)
    - double getReal()
    - void setImaginary(double i)
    - double getImaginary()

4. Write functions to perform the following operations:
    - **Complex addComplex(double r)**

      It adds r of type double to real part of complex number while imaginary part remains same. And returns newly generated complex number.

    - **Complex addComplex(Complex &c1)**

      It adds both complex numbers and returns newly generated complex number.

    - **Complex subComplex(double r)**

      It subtracts r of type double from real part of complex number while imaginary part remains same. And returns newly generated complex number.

- **Complex subComplex(Complex &c1)**
  It subtracts both complex numbers and returns newly generated complex number.

- **Complex mulComplex(double n)**
  It's a scalar multiplication. Real and imaginary parts are multiplied by n. and returns newly generated complex number.

- **Complex mulComplex(Complex &c1)**
  It multiplies both complex numbers and returns newly generated complex number. (a+bi)(c+di) = (ac−bd) + (ad+bc)i

Problem 3:

Design a class **Holiday** that represents a holiday during the year. This class has three **private** data members:

- **name**: A string that represents the name of holiday.
- **day**: An integer that holds the day of the month of holiday.
- **month**: A string that holds the month the holiday is in.

The class has the following **member functions:**

1. Write a default constructor that initializes each data member of class such that:
   name with **NULL**, **day** with **0** and **month** with **NULL** **Holiday()**
2. Write a constructor that accepts the arguments for each data member such that:
   string n assigned to **name**, **int d** to **day** and **string m** to **month**.
   Holiday(string &n, int d, string &m) // Use initialzer list
3. A copy constructor **Holiday(Holiday & copy)**
4. Generate getter setter of each member variable: such that **name** should never be greater than 50 characters, **day** should never be negative and **month** should not be greater than 10 characters.

5. Write a function **inSameMonth** (outside class) which takes two Holiday objects as arguments, compares two objects of the class Holiday, and returns true if they have the same month otherwise false.

   bool inSameMonth (Holiday &a, Holiday &b)

6. Write a function **avgDate** (outside class) which takes an **array** of type Holiday and its **size** as its argument and returns a **double** value that is the average of the entire day data member in the Holiday array **arr**. You may assume that the array is full (i.e. does not have any NULL entries).

   double avgDate(Holiday arr[], int size)

Submission Details:

1. Save single .cpp file with your roll no and lab number e.g. i22-XXXX_Lab7.cpp
2. Take screen shot of running test cases of tasks.
3. Zip the .cpp file and screen shots (Do not create .rar file) with roll no and lab no. e.g. i22-XXXX_Lab7.zip.
4. Submit the zip file on google class room.