Department of Information Technology

Course Code and Name : IT 2103 (Object-Oriented Programming)

Session : Odd Semester 2023 – 2024 [July 2023 – December 2023]

Assignment 1

Session : Odd Semester 2023 – 2024 [July 2023 – December 2023]

Date of Handing of Assignment : 29-09-2023

Date of Submission : 06-10-2023

Q1

Consider the description of class named ‘**MyTime**’ as follows:

1. The class has three object variables (instance fields) namely ‘**hours**’ (type:int, range 0-23), ‘**minutes**’ (type:int, range 0-59) and ‘**seconds**’ (type:int, range 0-59). All object variables have local class scope.
2. The class has a class variable of type integer named ‘**nTimes**’ with local scope initialized to 0 and its value gets incremented each time an instance of ‘**MyTime**’ is created.
3. Any instance of ‘**MyTime**’ class can be created by following any of the following ways:
   (i) Without any parameter (No Argument). In this case all instance field values are set to 0.
   (ii) With one integer type parameter having value of the ‘**hours**’ field only. The other instance fields (‘**minutes**’ and ‘**seconds**’) values are set to 0s.
   (iii) With two integer type parameters having values of ‘**hours**’ and ‘**minutes**’ fields. The other instance field (‘**seconds**’) value is set to 0.
   (iv) With three integer type parameter having value of ‘**hours**’, ‘**minutes**’ and ‘**seconds**’ fields.
4. The class provide an object method as well as a class method to compare any two instances of ‘**MyTime**’ class for greater than, less than and equality comparisons. Suppose ‘T1’ and ‘T2’ are two ‘**MyTime**’ class instances. These can be compared by either by invoking object method as T1.compareTo(T2) or by invoking class method as MyTime.compareTo(T1, T2). Both the methods return 1 if T1 > T2, -1 if T1 < T2 and 0 if T1 = T2. For relationship, first the ‘**hours**’ field values are checked, then ‘**minutes**’ field values are checked (if ‘**hours**’ values of T1 and T2 are same) and in the end ‘**seconds**’ values are checked (if ‘**hours**’ and ‘**minutes**’ values of T1 and T2 are same). [Use name of the method as compareTo(..) with return type as ‘int’].
5. The class supplies accessor method for every object variable.
6. The class supplies a global scope object method to display values of instance fields in AM/PM format. Use ‘**displayTime()’** as name of the method with global scope and return type as void.

You must write Java Code for the class ‘MyTime’ as per above-mentioned description.
Write a Driver class named ‘Test’ and create any five instances of class ‘MyTime’ and invoke various methods of the class.

Q2.

A Java program reads the values of attributes related to an exam of a student such as (i) name of student (type: String) (ii) marks in object-oriented programming (type: integer, range 0-100) (iii) marks in data structures and algorithms (type: integer, range 0-100) and (iv) marks in computer system architecture (type: integer, range 0-100) from a single command line argument. The values of various attributes are provided in a single command line argument (in string form) by user using a specialized format as follows:

- The value of student name is enclosed between a pair of two ‘$’ characters such as $Dora$, $Nitin$ etc.
- The value of marks in object-oriented programming is enclosed between a pair of two ‘&’ characters such as &100&, &34& etc.
- The value of marks in data structures and algorithms is enclosed between a pair of two two ‘#’ characters such as #89#, #56#.
- The value of marks in computer system architecture is enclosed between a pair of two ‘@’ characters such as @78@, @67@ etc.

For example, suppose the name of the driver class (class with public static void main(String args[]) method) is ‘OOP’. The values of various attributes are as follows (if the program is executed with specified commands)

- If executed with command ‘**java OOP &78&@100@$Mohan$#89#**’, then the value of the name of student is ‘**Mohan**’, the value of the marks in object-oriented programming is ‘**78’**, the value of the marks in data structures and algorithms is ‘**89’** and the value of the marks in computer system architecture is ‘**100’**.
- If executed with command ‘**java OOP ‘@30@$Ayush$#67#&89&**’, then the value of the name of student is ‘**Ayush’**, the value of the marks in Object-oriented programming is ‘**89’**, the value of the marks in data structures and algorithms is ‘**67’** and the value of the marks in computer system architecture is ‘**30’**.

[Note: Assume the marks values are in integer form and are supplied by user in proper numeric format. No need to check any type of exceptions or error related with type/format of values].

You have to complete the java code of driver class named ‘**OOP**’ which extracts the values of various attributes from the single argument passed from the command line and displays the extracted values on System.out.

```
class OOP
{
        public static void main(String args[])
        {
                        String    values = args[0];
                // Complete the remaining part by using the variable ‘values’ of the above statement
        }// End of Method
}// End of class OOP
```

Q3

Consider the description of the class named ‘Matrix’ as follows:

- The class has three private instance fields as (i) ‘**elements’** of type int[][], (ii) ‘**rows**’ of type int and (iii) ‘**columns**’ of type int.
- The class supplies a parameterized constructor which receives the values for ‘**rows**’ and ‘**columns**’ fields. [Scope : Package Private]
- Supply an accessor and a mutator method for every instance field. [Scope : Public]
- The class has an object and a class method to add two Matrices. Suppose ‘M1’ and ‘M2’ are any two Matrix instances then ‘M1’ and ‘M2’ can be added either through M1.addMatrix(M2) (Alternatively M2.addMatrix(M1)) or through Matrix.addMatrix(M1,M2). The resultant matrix is returned as an output. [Scope : Package Private]

- The class has an object and a class method to check to equality of two matrices. The boolean type value true or false is returned. Two matrices are equal if they have same rows and columns and same elements values. [Name the Method as ‘equalsMatrix’]. [Scope : Package Private]
- Supply an object and a class method to get determinant value of a given matrix. [Scope : Package Private]
- Supply a suitable toString() Method which displays the elements of a given matrix on System.out in Row and Column Format. [Scope : Package Private]
- Supply an Object and a class method to compare any two matrices say ‘M1’ and ‘M2’ for equality, greater than or less than. Two matrices are equal if their determinant values are equal. Matrix ‘M1’ is greater than matrix ‘M2’ if M1’s determinant value is more than M2’s determinant value and vice versa. [Scope : Package Private]
- Supply an object method to get the transpose of a Matrix. [Note : Transpose of a matrix is another matrix which is retrieved by exchanging the elements of rows and columns.

Q4

Suppose the code of the Matrix class of Q3 is available here. Complete the methods as per commented specification:

```
class MatrixTest
{
public   static   ArraList<Matrix> getMatricesWithDeterminant(ArrayList<Matrix> matrixList, int detV)
{
/*
This method adds the matrices from ‘matrixList’ with determinant value equal to detV to new Arraylist<Matrix> and returns it.
*/
}// End of Method
public   static   void   main(String args[])
{
   /*  Complete this method which performs the following tasks sequentially.
          1.    Create an ArrayList<Matrix> named ‘matList’.
          2.    Add any 10 elements to matList.
          3.    Create an Iterrator<Matrix> instance named itrM’ for ‘matList’
          4.    Use ‘itrM’ instance to display the matrices and their determinant values on System.out
          5.    Retrieve the matrices from ‘matList’ which have determinant value 100.
    */
}// End of Method
}// End of class
```