Sri Lanka Institute of Information Technology

B. Sc. Honours Degree
in
Information Technology
Mock Paper - 2020
IT2030 – Object Oriented Programming

Duration: 3 Hours

Instructions to Candidates:

- This paper contains **Three** questions. **Answer All** Questions.
- Marks for each question are given in the paper.
- Total Marks: 100.
- Create a separate Project for each question. The name of the project is provided. Save each Java program using the class name given.
- This paper contains **7** pages with the Cover Page.
- You can use any IDE to implement the solution.

Instructions to Candidates when submitting:

- Save all your work.
- Create a folder from your student ID.
- Inside that, create 3 separate folders from the project name provided.
- Copy each project answer source codes(Only the .java files) in to respective folders.(There should be 3 folders name as **Question01, Question02, Question03** inside your ID folder, and in each folder should contain the answer.(.JAVA files ONLY).
- Zip the Student ID folder(Zip folder also should be the Student ID number).
- Upload into the correct link.

Question 1 (35 marks)

This question is based on the **Object-Oriented Programming (OOP) concepts**. You are going to control two types of satellites called Drone Satellite and Navigational Satellite from one location called Satellite Center.

a) You can refer the output is given in **SatelliteDemo** class and adjust your code accordingly

```
3  public class SatelliteDemo {
4
5      public static void main(String[] args) {
6
7          ISatellite navigationalSatellite = new NavigationSatellite("Ravana-01");
8          IGeoLocation locationTracker1 = new SatelliteLocation("Sri Lanka");
9          ISatellite droneSatellite = new DroneSatellite("Ravana-02");
10         IGeoLocation locationTracker2 = new SatelliteLocation("Russia");
11
12         ISatellite [] satelliteArray = new ISatellite[]{navigationalSatellite, droneSatellite};
13         IGeoLocation [] trackerArray = new IGeoLocation[]{locationTracker1, locationTracker2};
14
15         SatelliteCenter satelliteCenter = new SatelliteCenter(0, satelliteArray,trackerArray);
16         satelliteCenter.startService();
17         satelliteCenter.stopService();
18         satelliteCenter.locationService();
19
20         SatelliteCenter remoteController2 = new SatelliteCenter(1, satelliteArray,trackerArray);
21         remoteController2.startService();
22         remoteController2.stopService();
23         remoteController2.locationService();
24     }
25 }
```

Console | Javadoc | Problems | Declaration | Servers | Data Source Explorer | Debug

```
<terminated> SatelliteDemo [Java Application] C:\Program Files\Java\jre1.8.0_20\bin\javaw.exe (Sep 2, 2019, 9:06:47 PM)
Ravana-01 navigational satellite activate
Ravana-01 navigational satellite deactivate
Satellite Location is = Sri Lanka

Ravana-02 drone satellite activate
Ravana-02 drone satellite deactivate
Satellite Location is = Russia
```

i). First implement the **ISatellite** interface and declare **activate()** and **deactivate()** methods.
(03 marks)

ii). Then implement the **IGeoLocation** interface and declare the method called **displayLocation()**
(03 mark)

iii). Create two classes called **DroneSatellite** and **NavigationSatellite** and implement the **ISatellite** interface in each class and override necessary methods in each. You should overload the constructor to pass the name of the satellite in both classes.
(4 X 2 = 08 mark)

iv). Similarly create a class called **SatelliteLocation** and implement the **IGeoLocation** interface with in the class and **override the displayLocation()** method. Then overload the constructor to pass the location of the satellite.

(03 marks)

b) Satellite center maintain multiple satellites and multiple Geo Location trackers. To activate each satellite and the tracker the option can be used as a switch.

i). Create the **SatelliteCenter** class and implement the properties **option(int),** and array of **ISatellite (ISatellite [ ])** and the array of **IGeoLocation (IGeoLocation [ ])** tracker.

(03 marks)

ii). Overload the constructor of the same class and initialize the above properties.

(03 marks)

iii). Implement the method called **startService()** and you should invoke the **activate()** method of the satellite class by using the option as switch. [E.g.: - if option = 0 activate Navigation Satellite if option = 1 activate drone satellite]

(03 marks)

iv). Implement the method called **stopService()** and you should invoke the **deactivate()** method.

(03 marks)

v). Then develop the **locationService()** method and based on the given option tracker should invoke the **displayLocation()** method

(03 marks)

vi). Extends the **SatelliteDemo** class by adding another Drone Satellite and the tracker. Display your modified output again in the console

(03 marks)

Save the project as **Question01**

Question 2 (30 marks)

This question is based on the **Collection Framework and Generics**.

a) You should implement an array list of Students and Lecturers and use one Generic class called **GenericPerson** to display elements in both array lists. Please refer the **GenericPersonDemo** Test class and its execution output to fine-tune your results.

```
15 public class GenericPersonDemo {
16
17     public static void main(String[] args) {
18         ArrayList<Student> students = new ArrayList<>();
19         students.add(new Student("STD1111", 6));
20         students.add(new Student("STD2222", 7));
21         students.add(new Student("STD3333", 12));
22         students.add(new Student("STD4444", 11));
23         students.add(new Student("STD5555", 10));
24
25         ArrayList<Lecturer> lecturers = new ArrayList<>();
26         lecturers.add(new Lecturer("EMP0000", "IT"));
27         lecturers.add(new Lecturer("EMP1111", "SE"));
28         lecturers.add(new Lecturer("EMP2222", "CSN"));
29         lecturers.add(new Lecturer("EMP3333", "EE"));
30         lecturers.add(new Lecturer("EMP4444", "IS"));
31
32         GenericPerson genericPerson = new GenericPerson();
33         genericPerson.displayElements(students);
34         genericPerson.displayElements(lecturers);
35     }
36 }
```

Console | Javadoc | Problems | Declaration | Servers | Data Source Explorer

```
<terminated> GenericPersonDemo [Java Application] C:\Program Files\Java\jre1.8.0_20\bin\java
Student = STD1111, Grade = 6
Student = STD2222, Grade = 7
Student = STD3333, Grade = 12
Student = STD4444, Grade = 11
Student = STD5555, Grade = 10

Lecturer = EMP0000, Department = IT
Lecturer = EMP1111, Department = SE
Lecturer = EMP2222, Department = CSN
Lecturer = EMP3333, Department = EE
Lecturer = EMP4444, Department = IS
```

i). Implement an interface **IPerson** and declare the method **displayDetails()** should return the output in **String** type.

(03 marks)

ii). Create a class called **Student** and implement the two properties called **studetID** (String) and **grade** (int) and values should be assigned through the **overloaded constructor**.

(03 marks)

iii). Implement the **IPerson** interface in the **Student** class and override the method **displayDetails()** to print the student ID and the grade.

(03 marks)

iv). Create a class called **Lecturer** and implement the two properties called **employeeID** (String) and **department** (String) and the values should be assigned through the **overloaded constructor**.

(03 marks)

v). Implement the **IPerson** interface in the **Lecturer** class and override the method **displayDetails()** to print the **employeeID** and the **department**.

(03 marks)

vi). Now create the generic class called **GenericPerson** and implement the method **displayElements** should support passing **generic array list** (either Lecturers array list or Students array list). The **displayElements()** method should have an iteration and within the iteration, the each element should call the **displayDetails()** method to print the Lecturer and Student details as per the given output.

(08 marks)

b) You should create a class called **AscendingTable** and that should store elements as key, value pairs. Keys should be stored according to the Ascending order. Implement the **display()** method that should print keys and values according to the ascending order. Refer the **GenericDemo** Test class and console output to adjust your results accordingly

[07 marks]

```
18 public class GenericDemo {
19
20     public static void main(String[] args) {
21
22         AscendingTable<Integer, String> myTable = new AscendingTable<>();
23         myTable.add(40, "ddd");
24         myTable.add(10, "aaa");
25         myTable.add(30, "ccc");
26         myTable.add(20, "bbb");
27
28         AscendingTable<Integer, Double> myTableD = new AscendingTable<>();
29         myTableD.add(40, 10.123);
30         myTableD.add(30, 23.456);
31         myTableD.add(20, 34.567);
32         myTableD.add(10, 45.678);
33
34         AscendingTable.display(myTable);
35         AscendingTable.display(myTableD);
36     }
37 }
```

```
Console  Javadoc  Problems  Declaration  Servers  Data Source Explorer  Debug
<terminated> GenericDemo [Java Application] C:\Program Files\Java\jre1.8.0_20\bin\javaw.exe (Sep 2, 2019, 12:27:05 AM
10, aaa
20, bbb
30, ccc
40, ddd
10, 45.678
20, 34.567
30, 23.456
40, 10.123
```

Save the project as **Question02**

Question 3 (35 marks)

This question is based on the **Threads** implementation.

a) You are going to implement two threads to multiply numbers and add numbers called **MultiplyThread (Thread-1)** and **PlusThread (Thread-0)** respectively. **TestThread** class is given as below and both Threads should execute one after the other for the given range and check the given output to make your implementation ease.
[**Assumption: - Thread synchronization is essential and both threads should print the output as synchronized manner. Correct implementation of *wait()*, *notify()* methods is compulsory to obtain full marks**]

TestThreads.java

```
1  package paper.v1.Q3;
2
3  public class TestThreads {
4
5      public static void main(String[] args) {
6
7          Object lock = new Object();
8          Thread plusThread = new Thread(new PlusThread(lock, 2, 10));
9          Thread multiplyThread = new Thread(new MultiplyThread(lock, 2, 10));
10         plusThread.start();
11         multiplyThread.start();
12     }
13 }
```

Console | Javadoc | Problems | Declaration | Servers | Data Source Explorer | Debug

```
<terminated> TestThreads [Java Application] C:\Program Files\Java\jre1.8.0_20\bin\javaw.exe (Sep 1, 2019, 11:21:47 PM)
Thread-1 => 2 X 2 = 4
Thread-0 => 2 + 2 = 4
Thread-1 => 3 X 3 = 9
Thread-0 => 3 + 3 = 6
Thread-1 => 4 X 4 = 16
Thread-0 => 4 + 4 = 8
Thread-1 => 5 X 5 = 25
Thread-0 => 5 + 5 = 10
Thread-1 => 6 X 6 = 36
Thread-0 => 6 + 6 = 12
Thread-1 => 7 X 7 = 49
Thread-0 => 7 + 7 = 14
Thread-1 => 8 X 8 = 64
Thread-0 => 8 + 8 = 16
Thread-1 => 9 X 9 = 81
Thread-0 => 9 + 9 = 18
Thread-1 => 10 X 10 = 100
Thread-0 => 10 + 10 = 20
```

i). You have to overload the **PlusThread** constructor with a lock (for synchronization), **start** and **range** parameters.

(03 mark)

ii). Implement a method called **addNumbers(Object lock, int start, int range)** and pass parameters which are passed through the overloaded constructor.

(05 marks)

iii). In each iteration the Thread should **sleep 1 second** of time interval and it should print the thread name and given values as per the given output.

(04 marks)

iv). Override the **run()** method and call the **addNumbers** method within that.

(05 marks)

b) **MultiplyThread** should print the values as per the given console output and use iterator to limit the start and the range to be printed with displaying the name of currently running thread. [**Hint: - Thread.currentThread.getName()**]

i). You have to overload the **MultiplyThread** constructor with a lock (for synchronization), **start** and **range** parameters.

(05 mark)

ii). Implement a method called **multiplyNumbers(Object lock, int start, int range)** and pass parameters which are passed through the overloaded constructor.

(05 marks)

iii). In each iteration the Thread should **sleep 1 second** of time interval and it should print the thread name and given values as per the given output.

(03 marks)

iv). Override the **run()** method and call the **multiplyNumbers** method within that.

(05 marks)

Save the project as **Question03**