00014

Student ID:	
Machine No:	

Sri Lanka Institute of Information Technology

B.Sc. Honours Degree in Information Technology

Specialized in Information Technology

Final Examination (Computer Base)
Year 2, Semester 1 (2023)

IE2021 – Object Oriented Programming

Duration: 3 Hours

June 2023

Instructions to Candidates:

- This paper contains four questions. Answer All Questions.
- Marks for each question given on the paper. Total Marks: 100.
- Create a folder on the Desktop with your student registration number and store all your program files inside that.
- Create a separate Project for each question. The name of the project is provided.
- This paper contains 08 pages including the Cover Page.

Instructions to Candidates when submitting:

- Save all your work.
- Delete all unnecessary files from each project folder (There should be 4 folders name as Question01, Question02, and Question03 inside your ID folder, and each folder should contain the answer (.JAVA files ONLY)).
- Zip the Student ID folder (Zipped folder also should be named with Student ID number).
- Upload the zipped folder into the correct link.

Question 1 (20 marks)

This question is based on the **OOP concepts**.

Implement the necessary classes, along with their required attributes and methods, based on the following description.

You have been assigned the task of developing a simple system to manage the employees of a company. The system should consist of two classes, **Employee** and **Manager**. The Employee class should have three data members, ***EmpId, name***, and ***address***, which are all of string data type. It should also have a constructor to initialize these attributes and two methods, **Read()** and **Print()**. The **Read()** should accept inputs for the above mentioned attributes through the keyboard and **Print()** should display their values.

The **Manager** class should extend from the **Employee** class and should have four data members ***Department, ProductNo1, ProductNo2***, and ***ProductNo3***, where product numbers are integers and ***Department*** is a string. It should have a constructor to initialize all these attributes and a method, **Read(),** to input the values for these attributes along with ***EmpId, name,*** and ***address*** by calling the **Read()** method of the **Employee** class. The **Read()** method of the Manager class should use a Try Catch block to validate the input of numbers for the three product numbers. Additionally, it should have a **Print()** method to display the Manager details along with the Employee details by calling the **Print()** method of the **Employee** class.

Furthermore, you need to create a class called **EmployeeApp** with a main method that uses an **ArrayList** to store two **Employee** Class Objects and two **Manager** Class Objects. The program should **only** allow the **Employee** class and the descendants of the **Employee** class to be stored in the ArrayList. Finally, display all the four objects stored in the ArrayList by calling each object's **Print()** method.

Hint: Refer to the sample output given below.

Save the project as **Question01**

Sample Output:

```
Employee ID is : EMP101
Employee Name is : Kamal Perera
Employee Adress is : Colombo

Employee ID is : EMP102
Employee Name is : Sunil Fernando
Employee Adress is : Kandy

Employee ID is : EMP103
Employee Name is : Nimali Herath
Employee Adress is : Kadawatha
Employee Department is : Marketing
Leading first product is : 10001
Leading second product is : 10002
Leading third product is : 10003

Employee ID is : EMP104
Employee Name is : Hansika Perera
Employee Adress is : Ratnapura
Employee Department is : Marketing
Leading first product is : 20001
Leading second product is : 20002
Leading third product is : 20003
```

Question 2 (30 marks)

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

a) Write a program that takes input from the user and creates an ArrayList of integers. The program should then remove all the odd numbers from the ArrayList and display only the even numbers in the ArrayList.

i) Create an empty ArrayList of **integers** using the ArrayList class. Then prompt the user to enter numbers, one at a time, until they enter 0. Each number the user enters must be added to the ArrayList. [6 marks]

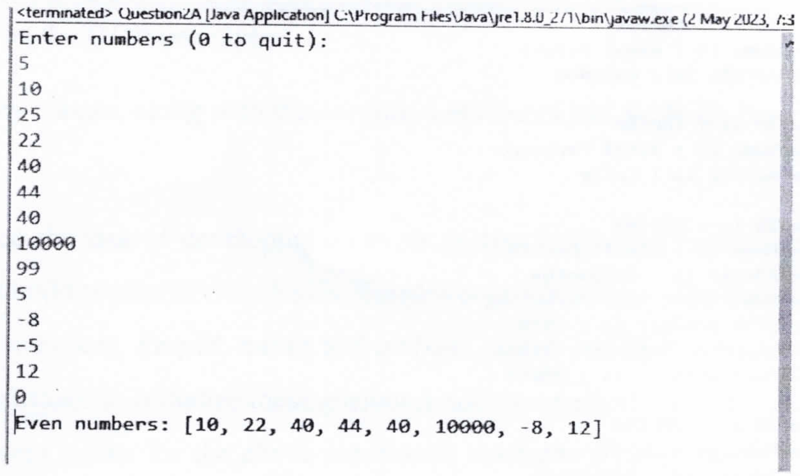
ii) Once the user has entered all the numbers, remove any odd numbers from the ArrayList. [7 marks]

iii) Display the remaining even numbers in the ArrayList [4 marks]

Hint: Refer to the sample output given below.

Save the project as **Question02A**

Sample Output:



```
<terminated> Question2A [Java Application] C:\Program Files\Java\jre1.8.0_271\bin\javaw.exe (2 May 2023, 7:3
Enter numbers (0 to quit):
5
10
25
22
40
44
40
10000
99
5
-8
-5
12
0
Even numbers: [10, 22, 40, 44, 40, 10000, -8, 12]
```

b) Write a generic class called **"Pair<T, U>"** that represents a pair of two values, one of type **T** and another of type **U**. The class should have the following methods. [2 marks]

   i) Write a parameterized constructor that initializes the first and second values of the pair. [3 marks]

   ii) Write a method called **"getFirst()"** which returns the first value of the pair. [2 marks]

   iii) Write a method called **"getSecond()"** which returns the second value of the pair. [2 marks]

   iv) Write a class called **"MainApp"** which contains the main method. Create two objects where the first object accepts a **String** as the first value, **Integer** as the second value of the pair. The second object should accepts a **Double** as the first value and a **String** as the second value. [4 marks]

Hint: Refer to the sample output given below.

Save the project as **Question02B**

Sample Output:

```
Printing First Pair
First value: Test 1
Second value: 42

Printing Second Pair
First value: 3.14
Second value: Test 2
```

00014

Question 3 **(25 marks)**

This question is based on the **Exceptions**.

a) Write a program that prompts the user to enter a username and password, then performs the following three validations using three different custom exception classes as follows.

   i) **InvalidUserName**, is an exception class that prints out the error message "The username must be at least 6 characters long" if the number of characters in the user input for the username is less than 6. [6 marks]

   ii) **InvalidPasswordLength**, is an exception class that prints out the error message "The password must be at least 8 characters long" if the number of characters in the user input for the password is less than 8. [6 marks]

   iii) **InvalidPassword**, is an exception class that prints out the error message "The password must contain at least one uppercase letter, one lowercase letter, and one digit" if the number of characters in the user input for the password is not containing at least one uppercase letter, one lowercase letter, and one digit. [6 marks]

b) Write another class called **DemoApp** to get the user inputs for the username, password and then validate those for above three conditions. If the input has one or more custom exception, you should have a proper try catch statements to handle the exceptions.

[7 marks]

Save the project as **Question03**

Sample Output 1:

```
Enter a username: Kamal
Enter a password: Kamal123
Error: Username must be at least 6 characters long.
```

Sample Output 2:

```
Enter a username: KamalPerera
Enter a password: Ka1
Error: Password must be at least 8 characters long.
```

Sample Output 3:

```
Enter a username: KamalPerera
Enter a password: kamal123
Error: Password must contain at least one uppercase letter, one lowercase letter, and one digit.
```

Question 4 **(25 marks)**

This question is based on the **Design Patterns** implementation. You have to draw the class diagram for the below scenario based on the **Command** and **Singleton** patterns.

As a software developer, you have been tasked with implementing a WAR system for launching missiles. The system should allow for two types of missiles (Heat and Rocket) to be launched and blasted upon providing the source and destination locations. To achieve this, you need to implement the **MissileSystem** interface and create two classes, **HeatMissileSystem** and **RocketMissileSystem**, both of which should implement the **MissileSystem** interface and override the launch and blast methods. Missile systems and controllers are created according to the **Singleton pattern** and overall application developed according to the **Command pattern.**

Additionally, you should refer to the **MissileOperation** interface and declare the initiateOperation method. Then, implement this interface in **BlastMissile** and **LaunchMissile** classes and override the initiateOperation method in each class.

Finally, you need to create the **MissileController** class, which sets two operations - launch and blast. This class should work according to the Singleton design pattern. You will also need to implement the performLaunching and performBlasting methods in this class.

Refer the below screenshot of the Main program and check the Console Outputs.

Use the empty space given below to draw the class diagram.

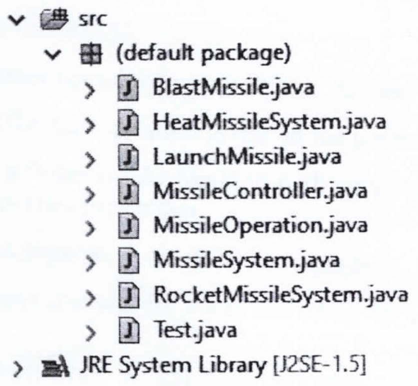
Main Program:

```java
public class Test {

    public static void main(String[] args) {
        MissileSystem heatMissileSystem = HeatMissileSystem.getInstance("Heat Missile");
        MissileSystem rocketMissileSystem = RocketMissileSystem.getInstance("Rocket Missile");
        MissileController missileController = MissileController.getInstance();

        LaunchMissile launchHeatMissile = new LaunchMissile(heatMissileSystem);
        BlastMissile blastHeatMissile = new BlastMissile(heatMissileSystem);
        missileController.setOperations(launchHeatMissile, blastHeatMissile);
        missileController.performLaunching("Colombo");
        missileController.performBlasting("Flight MH370");

        LaunchMissile launchRocketMissile = new LaunchMissile(rocketMissileSystem);
        BlastMissile blastRocketMissile = new BlastMissile(rocketMissileSystem);
        missileController.setOperations(launchRocketMissile, blastRocketMissile);
        missileController.performLaunching("Kandy");
        missileController.performBlasting("Flight MH420");
    }
}
```

Console Output:

```
Initialize Heat Missile System...
Initialize Rocket Missile System...
Initialize Missile Controller
Heat Missile launch from Colombo and Heat Missile blast Flight MH370
Rocket Missile launch from Kandy and Rocket Missile blast Flight MH420
```

Package Hierarchy:



```
v src
   v (default package)
      > BlastMissile.java
      > HeatMissileSystem.java
      > LaunchMissile.java
      > MissileController.java
      > MissileOperation.java
      > MissileSystem.java
      > RocketMissileSystem.java
      > Test.java
> JRE System Library [J2SE-1.5]
```

00014