00188

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 (2022)
Paper Version B

IT2030 – Object Oriented Programming

Duration: 3 Hours

November 2022

Instructions to Candidates:

- This paper contains four 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.
- Store all your program files in the Desktop Folder provided
- This paper contains 08 pages including the Cover Page.

Instructions to Candidates when submitting:

- Save all your work.
- Create a folder from your student ID. Inside that, create 4 separate folders from the project name provided.
- Copy each project answer source codes (Only the .java files) in to respective folders (There should be 4 folders name as Question01, Question02, Question03 and Question04 inside your ID folder, and in 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 (25 marks)

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

a) Implement an abstract class called **Customer** to do the following.

i) Store the following properties
id (int), name (String)

(02 mark)

ii) Implement a constructor to get two properties as parameters and initialize them.

(02 marks)

iii) Implement a method called **display()** to display these two properties.

(02 mark)

iv) Define an abstract method called **calculateBill().**

(01 mark)

b) Implement a sub class of Customer class called **RegisteredCus** to do the following.

i) Store the following properties
rewardPoints (double), netAmonut (double)

(02 mark)

ii) Implement a constructor to get two properties (id, name) as parameters and initialize them. The **rewardPoints** and **netAmonut** should be assigned as 0.

(02 marks)

iii) Override the **calculateBill()** method to calculate total amount of the customer's bill amount.

I. Get the bill amount of the customer as keyboard inputs.
[Hint: Use Scanner class]
II. If the user entered bill amount is greater than Rs. 950.00 add 15% from the bill amount as **rewardPoints.**
III. Give a 7% discount for all the registered customers and save the **netAmonut** by deducting the discount from bill amount.

(06 marks)

iv) Override a method called **display()** to display all the properties of a registered customer.
[Hint: Call the **calculateBill()** *method within the display() method and then print* **rewardPoints** *and* **netAmonut** *values]*

(04 marks)

c) Implement a class called **MainApp** which has the **main()** to perform followings.

i) Craete an ArrayList of **RegisteredCus** class and add two **RegisteredCus** objects to the arraylist.

(02 marks)

ii) Call the **display()** method for both the objects using a foreach loop.

(02 marks)

Save the project as **Question01**

Sample Output:

```
Customer ID: 101
Customer Name: Kamal
Enter the bill amount: 1050.00
Customer reward points: 157.5
Customer Net amount to pay: 976.5


Customer ID: 105
Customer Name: Sunil
Enter the bill amount: 550.00
Customer reward points: 0.0
Customer Net amount to pay: 511.5
```

Question 2 (15 marks)

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

a) Implement a generic class called **Employee**.

i) Include two properties called **name**, and **empId**. These properties should be able to hold any data type. For example, **empId** can be a String or even an Integer value.

(04 marks)

ii) Include an overloaded constructor that takes in two parameter to initialize the two properties.

(03 marks)

iii) Include another method called **getEmpId()** that will return the **empId** of the Employee object.

(02 marks)

b) Implement a class called **Company** with the **main** method.

i) Create a **HashMap** object called **empList** that uses Integer value as the Key and an **Employee** object as the value.

(02 marks)

ii) Create a **Employee** object with a String value as the **empId** and add to the **empList**

empId: "EMP12345"

Name: "Nishan Perera"

(2 marks)

iii) Create another **Employee** object with integer value as the **empId** and add to the **empList**.

empId: 123456

Name: "Krishan Gamage"

(2 marks)

Save the project as **Question02**

Refer the partial code of the main method given below.

```
class Company{
  public static void main(String[] args) {
    //start your code here

    //End your code here
    for (int id: empList.keySet()) {
      Employee value = empList.get(id);
      System.out.println("Employee number: "+ id + "and the Employee ID is: " + value.getEmpId());
    }
  }
}
```

Question 3 (30 marks)

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

A Wrapping paper Art is printed using computer program and which is drawn using two concurrent Threads. You are allowed to enter pattern styles through keyboard inputs and you should select number of occurrences (count) to be printed the style. Each thread should print patterns one after the other and you should print the inverted triangle shape using given style. Set names for the Threads as **Pattern01**, and **Pattern02**.

*[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]*

a) Implement the **Pattern01** class as below.

i) Overload the **Pattern01** constructor with a lock (for synchronization), and use Triangle **pattern** (String), **count** (int) as parameters.

(01 mark)

ii) Override the **run()** method and implement the pattern print logic.

(10 marks)

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

(01 mark)

b) Implement the **Pattern02** class as below

i) Overload the **Pattern02** thread constructor with a lock (for synchronization), and use Art work **pattern** (String), and **count** (int) as parameters.

(01 mark)

ii) Override the **run()** method and implement the pattern print logic

(10 marks)

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

(01 mark)

c) Implementation the **Main** method as below

i) Use Keyboard inputs to get the Pattern styles

(03 marks)

ii) Implement two Threads and set names **Pattern_01**, and **Pattern_02 Thread** and pass necessary parameters

(03 marks)

Save the project as **Question03**

Sample Output:

```
Enter Pattern 1 = +
Enter Pattern 2 = -
Enter count = 8
========Threads start printing patterns.=======
Pattern 02 Thread =   - - - - - - - -
Pattern 01 Thread =   + + + + + + + +
Pattern 02 Thread =    - - - - - - -
Pattern 01 Thread =    + + + + + + +
Pattern 02 Thread =     - - - - - -
Pattern 01 Thread =     + + + + + +
Pattern 02 Thread =      - - - - -
Pattern 01 Thread =      + + + + +
Pattern 02 Thread =       - - - -
Pattern 01 Thread =       + + + +
Pattern 02 Thread =        - - -
Pattern 01 Thread =        + + +
Pattern 02 Thread =         - -
Pattern 01 Thread =         + +
Pattern 02 Thread =          -
Pattern 01 Thread =          +
```

Question 4 (30 marks)

This question is based on the **Design Patterns** implementation.

a) You are going to implement the **Strategy Design Pattern** based on the scenario for meal preparation of a Restaurant. You can prepare three meals for the day (**Breakfast, Dinner, and Lunch**) with time duration of (**60 minutes, 45 minutes,** and **30 minutes** for each).

i). Implement two interfaces **IPrepareQuickly** and **IPrepareDeliciously**. Each interface you should declare methods (in **IPrepareQuickly** interface declare the method **void deliveryTime()** and in **IPrepareDeliciously** interface declare methods **void addFlavour()** and **double getCost()**)

(02 marks)

ii). Then create 3 classes **ChickenFlavour**, **FishFlavour**, and **EggFlavour** and those classes should implement the **IPrepareDeliciously** interface and override all methods with in the class.

(06 marks)

iii). Similarly create another 3 classes **OneHour**, **ThirtyMinutes**, and **FourtyFiveMinutes** and those classes should implement the **IPrepareQuickly** interface and override the method as well.

(03 marks)

iv). Create an **Abstract** class called **Meal** and aggregate two interfaces **(IPrepareQuickly,** and **IPrepareDeliciously),** you should set those two behaviors with using two set methods **setFlavour()** and **setDuration()**. (Those "set" methods are used to dynamically add prepare **quickly** and **prepare deliciously** features to meal)

(06 marks)

b) Now for the above three meals you can add different flavors such as **chicken, fish**, and **egg** and the cost of each flavored meal respectively chicken - 100/=, Fish – 80/=, and egg – 60/= rupees. Based on the flavor cost should be different.
[Assumption: You can't add more than one flavor per meal]

i). Then implement another two methods called **mealWithFlavour(),** and **mealInDuration()** and you should call relevant **addFlavour()** and **deliveryTime()** method respectively through the declared interfaces of the Meal class

(02 marks)

ii). Apart from that with in the **Meal** class you should add two **abstract** methods **displayMeal() and displayCost()**

(01 mark)

iii). Now **extends** the **Meal** class in the **Breakfast, Lunch and Dinner** classes. Implement all abstract methods. Within the **displayMeal()** method you should call for the **mealWithFlavour(), mealInDuration(), and displayCost() methods.**

(10 marks)

Refer the below sample java code of the Main program and check the Console Outputs. Write a java code to display the same output.

Save the project as **Question04**

Sample Output:

ThirtyMinutes.java | EggFlavour.java | Breakfast.java | Meal.java

```
1  package oop.design.patterns;
2
3  public class TestStratergy {
4
5      public static void main(String[] args) {
6
7          Meal meal = new Breakfast();
8          meal.setFlavour(new ChickenFlavour());
9          meal.setDuration(new ThirtyMinutes());
10         meal.displayMeal();
11
12         Meal meal2 = new Lunch();
13         meal2.setFlavour(new FishFlavour());
14         meal2.setDuration(new OneHour());
15         meal2.displayMeal();
16
17         Meal meal3 = new Dinner();
18         meal3.setFlavour(new EggFlavour());
19         meal3.setDuration(new FourtyFiveMinutes());
20         meal3.displayMeal();
21
22     }
23
24 }
25
```

Console

```
<terminated> TestStratergy [Java App

Preparing Breakfast......
Added Chicken for the meal
Meal is ready in 30 minutes
Cost for the meal is = 100.0

Preparing Lunch....
Added fish for the meal
Meal is ready in 60 minutes
Cost for the meal is = 80.0

Preparing Dinner.......
Added egg for the meal
Meal is ready in 45 minutes
Cost for the meal is = 60.0
```