University of Asia Pacific

Department of Computer Science and Engineering

Program: B.Sc. in CSE

Final Examination **Spring-2023** **2nd year 1st Semester**

Course Code: CSE 203 Course Title: Object-Oriented Programming I: Java Credit: 3

Time: 3.00 Hours. Full Mark: 50

There are **Five** Questions. Answer all of them. Part marks are shown in the margins.

1. a. Polymorphism is one of the key features of OOP. Explain how polymorphism can be achieved in Java. **[6]** **[CO1]**

b. Differentiate between abstract classes and interfaces in Java. **[4]** **[CO1]**

2. a. Write a Java program which takes the user's name, height in meters, and weight in kilograms and calculates their BMI (Body Mass Index). The equation to calculate BMI is: **[5]** **[CO2]**

$$BMI = weight / (height)^2$$

After calculating the BMI of the user, provide a welcome message to the user with his/her name and show their BMI range according to the chart given below:

Underweight	$BMI < 18.5$
Normal weight	$18.5 \leq BMI < 24.9$
Overweight	$25 \leq BMI < 29.9$
Obesity I	$30 \leq BMI < 34.9$
Obesity II	$35 \leq BMI < 39.9$
Obesity III (Morbidly obese)	$BMI \geq 40$

b. Write a Java program which takes user input to create an integer array and returns the sum and average of the array elements. **[5]** **[CO2]**

3 a. Create an abstract class **Phone**. Add 3 attributes: ***manufacturer***, ***storage***, and ***batteryPower***. Create a constructor that will take parameters for all attributes and initialize the respective attributes. Add the following methods to this class. [6] [CO3]

- *public void call(String toPhNum)*
  - Inside the method, print "Calling **toPhNum**" where **toPhNum** is the parameter
- *public void sendMessage(String toPhNum, String msg)*
  - Inside the method, print the msg parameter and then print "Message sent to **toPhNum**" where **toPhNum** is the parameter.

b. Create a **concrete** subclass "**SmartPhone**" of the above abstract class **Phone (Q#3a)** and also implement the **SmartDevice** interface below. Add an additional attribute, ***os*** (Operating System) to this class. Create a constructor that will take parameters for all 4 attributes and initialize the respective attributes properly. Override the necessary methods. Add an overloaded method of the *call()* method of the parent class. In the overloaded **call()**, pass an **additional** parameter **"usingApp"** to pass the name of the app such as WhatsApp, Messenger, etc. Inside the ***call*** method, call the ***runApp*(...)** method and pass the **usingApp,** and then call the *call*(...) method of the parent class. [4] [CO3]

```
public interface SmartDevice {
        void runApp(String appName);
}
```

4. a. Assume there is an **abstract** parent class "**Student**" which has an attribute ***highestCgpa***. **Student** class has 2 subclasses: "**UnderGraduateStudent**" and "**GraduateStudent**". [5] [CO3]

Now, declare a static method named **setHighestCgpa** (Assume the method is inside the **UAP** class) which will take only one parameter that can hold both an **UnderGraduateStudent** object and a **GraduateStudent** object. Inside the method set the ***highestCgpa*** to 4 if the parameter is an object of the **UnderGraduateStudent** class or to 5 if it is a **GraduateStudent** object. For example, in the code segment below, the first method call will set the ***highestCgpa*** of that student to 4 and the second method call will set the ***highestCgpa*** of that Graduate student to 5.

```
UAP.setHighestCgpa(new UnderGraduateStudent());//this sets highestCgpa to 4
UAP.setHighestCgpa(new GraduateStudent()); //this sets the highestCgpa to 5
```

b. Carefully observe the code of the **BankAccount** class and **SouthEastBank** class below. Identify the **errors** in the code below and fix the errors. You are not allowed to delete any line of code. You can only add new lines or edit existing lines. Write the output after fixing the errors. [5] [CO4]

```java
package sp23final;

public class BankAccount {
    String name, accNum;
    private double balance;
    public BankAccount(String name, String accNum, double bal) {
        this.name = name;
        this.accNum = accNum;
        this.balance = bal;
    }

    public void deposit(double amt) {
        balance += amt;
    }

    public void withdraw(double amt) {
        balance += amt;
    }

    public void display() {
        System.out.println(this.name+":"+accNum+":"+balance);
    }
}
```

```java
package sp23final;

public class SouthEastBank {

    public static void main(String[] args) {
        BankAccount b1 = new BankAccount("Mahi", "11111", 1000);
        BankAccount b2 = new BankAccount("Arnob", "22222");
        transfer(b1, b2, 500);
        b1.display();
        b2.display();

    }

    public void transer(BankAccount a1, BankAccount a2, double amt) {
        a1 = new BankAccount("Rafi", "33333", 2000);
        a1.withdraw(amt);
        a2.deposit(amt);
        System.out.println(a2.balance);
    }
}

```

5. • **Answer (a,b) or (c,d)**

a. Create a user-defined exception named **InvalidTemperatureRangeException** that takes two parameters, **minTemp** and **maxTemp**, in the constructor. Inside the constructor, set the exception message to "Temperature should be between **minTemp** and **maxTemp** degrees." [4] [CO5]

b. Define a static method "**runAirCooler**" which will take a parameter ***temp***. Inside the method, if the ***temp*** is between 10 to 28, print "Running at ***temp***". Otherwise, throw the **InvalidTemparatureException (Q5a)** and pass 10 and 28 as the value of **minTemp** and **maxTemp**. [6] [CO5]

Now from the main method, call the **runAirCooler** method twice and pass 15 and 30 respectively. Take appropriate measures in the main method to handle the exception properly.

Or

c. Create a multi-threaded program running 3 threads in parallel. Create the threads by **implementing** the **Runnable** interface. Each Thread will take 3 integers: **n, min,** and **max,** generate **n** random numbers between **min** and **max,** and **print** those numbers. The **first** thread will print 10 random numbers between **1 to 100, 2nd thread** will print 5 random numbers from **101 to 200,** and **3rd thread** will print 8 random numbers from **201 to 300.** [5] [CO5]

Note: Do not create 3 different classes for 3 threads, rather create one class and pass different parameters for different threads.

d. Assume your OOP course teacher is storing the final scores (out of 100) in a txt file where each line contains the **id** of the student followed by the score as shown in the sample file below. Write a Java program to read the file (one line at a time), determine if the student has passed or failed, and write that info in the Console. Sample input and output are given below. [5] [CO5]

Sample input file	Expected output
22101001 90	22101001 P
22101002 80	22101002 P
22201005 45	22201005 P
22201007 88	22201007 P
22201017 20	22201017 F
22201019 40	22201019 P