University of Asia Pacific

Department of Computer Science and Engineering

Program: B.Sc. in CSE

Final Examination | **Fall-2022** | 2nd year 1st Semester

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

Time: 3.00 Hours. | Full Mark: 150

Instructions:

1. There are **Six (6)** Questions. Answer all of them. All questions are of equal value. Part marks are shown in the margins.
2. Non-programmable calculators are allowed.

1. a. Discuss the Java identifier rules with examples. *[8]* *CO1*

   b. Explain the major difference between Java arrays and *C/C++* arrays. *[8]* *CO1*

   c. Describe Java Garbage Collection with examples. *[9]* *CO1*

2. a. Write a Java Program to take 5 integer inputs from the user, then sum up only the odd numbers, and then print the sum. *[10]* *CO2*

   b. Write a Java Program to take 6 integer inputs from the user, then sum up only the prime numbers, and then print the sum. [15] CO2

3. a. Create an **abstract** class named "**Professional**" and add the following to this class. [12] CO3

   i) Add 5 attributes: ***name, age, specialty, designation, and salary***.

   ii) Create a constructor and pass parameters for all attributes. Inside the constructor, initialize all attributes with the parameters passed to the constructor.

   iii)Add the following methods

   - A complete/concrete method name "***public void promotion(String newDesignation, double newSalary)***"
     - Inside the method, set the ***designation*** to ***newDesignation*** and ***salary*** to ***newSalary*** where ***designation*** and ***salary*** are the attributes whereas ***newDesignation*** and ***newSalary*** are the parameters passed to the constructor.
   - An **abstract** method named **jobDescription()** which does not return anything.
   - Override ***toString()*** method and return the concatenated value of name, age, and specialty in String format.

   b. Create a subclass of the above "**Professional**" class and name this class **Programmer**". [13] CO3
   Add an additional attribute **rank**. Create constructor and pass ***rank*** and all attributes of parent class except ***specialty***. Implement the constructor in the proper way and set the value of ***specialty*** to "ICT". Override the abstract method and print "Do professional programming" inside the method.

OR

a. Create an Interface named "**Engine**" and add the following methods. **[10] CO3**

```
void start()
void running(int min)
void stop()
```

b. Create a class named "**Car**" using the "**Engine**" interface and override the necessary methods. Additionally, add 2 more attributes named ***model*** and ***speed*** and a parameterized constructor to this class. Implement the **Car** class in such a way that the code below shows the Expected Output shown in the right column. **[15] CO3**

Code (just the main method)	**Expected Output**

`public static void main(String[] args) {`
`  Car car = new Car("Toyota Corolla", 30);`
 	
`  /* Following line will show "Toyota Corolla started at speed=30."*/`
`  car.start();`
 	
`  /* Output of method call below will be`
`  Running at speed=30.`
`  Distance covered=300.*/`
`  car.running(10);`
 	
`  car.speed = 40;`
 	
`  /* Output of method call below will be`
`  Running at speed=40.`
`  Distance covered=200.*/`
`  car.running(5);`
 	
`  // Following line will display "Stopped."`
`  car.stop();`
`}`
	Toyota Corolla started at speed=30.
	Running at speed=30.
	Distance covered=300.
	Running at speed=40.
	Distance covered=200.
	Stopped.

4. a. What is the output of the code on the next page if you enter the last 2 digits of your registration no, your first name, and your phone no as the 3 inputs id, name, and phNo respectively? Show the detailed steps of output calculation. **[10] CO4**

```java
package fexam;

public class Person {
    String name, phoneNo;

    public Person(String name, String phoneNo)
    {
        this.name = name;
        this.phoneNo = phoneNo;
    }

    public int findMagicNumber(int id) {
        int index = id%11;
        char d = phoneNo.charAt(index);
        int digit = Integer.parseInt(""+d);
        return 2*digit;
    }
}
```

```java
package fexam;
import java.util.Scanner;

public class TestPerson {

    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        int id = scan.nextInt();
        String name = scan.next();
        String phNo = scan.next();

        Person p = new Person(name, phNo);
        int output = p.findMagicNumber(id);
        System.out.println(output);
        scan.close();
    }
}
```

b. 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. *[15] CO4*

```java
public class TestError {
    public static void main(String[] a) {
        Student st1 = new Student("Abir", "111", 3.5f);
        Student st2 = new Student("Hasan", "111", 3.5f);
        System.out.println(st1);
    }
}

class Student {
    private String name, id;
    private float cgpa;
    public final static String univName = "UAP";
    public int studentCount = 0;

    public void Student(String name, String id, float cgpa) {
        this.name = name;
        this.id = id;
        this.cgpa = cgpa;
        studentCount++;
    }

    public static void increaseStudentCount(int inc_amt) {
        studentCount += inc_amt;
    }

    public static void setUnivName(String newName) {
        univName = newName;
    }
}
```

5. a. Given the following partial class "**Voter**", implement encapsulation for the "***name***" attribute and implement the **read-only** feature for the "***age***" attribute. Also, create an **overloaded** method of the ***grow()*** method. [19] CO3

```
public class Voter {
	private String name;
	private int age;

	public Voter(String name, int age) {
		this.name = name;
		this.age = age;
	}

	public void grow() {
		age++;
	}
}
```

b. Carefully observe the member inner class below. Create a class with the main method. Inside the main method, do the following. [6] CO5

i) Create an object of the **House** class (pass parameters as needed), call the ***getArea***() method, and print the output of the ***getArea***() method.

ii) Create an object of the Inner class **Room** (pass parameters as needed), call the ***getArea***() method, and print the output of the ***getArea***() method.

```
public class House {
	int width, length;
	int noOfRooms;
	public House(int width, int length, int nOfRooms) {
		this.width = width;
		this.length = length;
		this.noOfRooms = nOfRooms;
	}

	public int getArea() {
		return length*width;
	}

	// Inner class with 2 attributes width and length.
	public class Room{
		int width, length;
		public Room(int width, int length) {
			this.width = width;
			this.length = length;
		}

		public int getArea() {
			return length*width;
		}
	}
}
```

6. a. Create a class with multi-level try-catch i.e., nested try-catch where the exception is not handled in the inner level and handled by the outer level. **[15] CO5**

OR

Create a multi-threaded program with 4 threads where each thread will add even numbers to an array. The 1st thread will add even numbers from 20 to 30, the 2nd thread from 40 to 50, the 3rd thread from 60 to 70, and the 4th thread from 80 to 90. **[15] CO5**

b. Create a user-defined exception named **InvalidCgpaException** which will take 2 parameters **minCgpa** and **maxCgpa** as parameters of the constructor and set the exception message to "Valid CGPA should be between **minCgpa** and **maxCgpa**" where **minCgpa** and **maxCgpa** are the parameters. **[10] CO5**

OR

Create a multi-threaded program with 3 threads where each thread will add 5 prime numbers from 50 to 80 to an array. **[10] CO5**