United International University
Department of Computer Science and Engineering

CSE 1115: Object Oriented Programming   Mid: Fall 2025

Total Marks: 30   Time: 1 hour and 30 minutes

Any examinee found adopting unfair means will be expelled from the trimester / program as per UIU disciplinary rules.

Answer all ***five(5)*** *questions. The numbers on the right of the questions denote their marks.*

1. The CSE department wants to store student information in a jagged 2D array of objects, where each row can contain a different number of Student objects. (5)

Student.java

```java
import java.util.Scanner ;
class Student {
    String name ;
    int id ;
    double cgpa ;
    // Task A: Write a suitable constructor here
}
public class StudentArray {
    public static void main (String[] args ) {
        Scanner scn = new Scanner(System.in) ;

        // Task B: Declare a jagged 2D array of Student with 3
           rows

        // Task C: Allocate each row with the fixed sizes :

        // Task D: Nested loops -> take input -> create objects
           -> store in array
        scn.close();
    }
}
```

(a) Complete the **Student** class by writing a suitable constructor so that the following code runs correctly: **Student st = new Student("Anwar",501,3.65);**

(b) Declare a jagged 2D array of Student objects with 3 rows, following the structure given above.

(c) Allocate each row with the following fixed sizes:

- Row 0 $\rightarrow$ 2 students
- Row 1 $\rightarrow$ 1 student
- Row 2 $\rightarrow$ 3 students

(d) Using nested loops, take input for each student (name, id, cgpa) in the order **name (String), id (int), cgpa (double)**. Create and store **Student** objects for every position in the jagged array.

2. Write the output of the following program: (6)

B.java

```java
class A {
    int a = 3;
    static int cnt = 5;
    A() {
        this(6);
        a = a + cnt;
        System.out.println("A(): no-arg");
    }
    A(int a) {
        this.a = a;
        cnt++;
        System.out.println("A(int)");
    }
    {
        a = 9;
        System.out.println("A: instance block");
    }
    static {
        cnt = 10;
        System.out.println("A: static block");
    }
    void display() {
        System.out.println("A - a=" + a + ", cnt=" + cnt);
    }

}

public class B {
    static int cnt;

    static  {
        System.out.println("B: static block");
    }

    B() {
        cnt++;
        System.out.println("B() - no arg");
    }

    void show() {
        System.out.println("B - cnt=" + cnt);
    }
    public static void main(String[] args) {
        System.out.println("---main---");
        B b1 = new B();
        A a1 = new A();
        a1.display();
        b1.show();
        A a2 = new A(5);
        a2.display();
    }
}
```

3. Observe the following code. Make necessary changes so that the code runs without any errors and generate the given output.**You are not allowed to change any access modifiers.** (6)

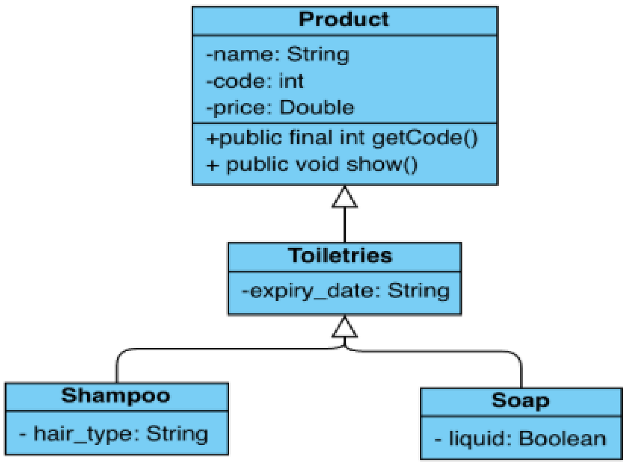
PhoneInfo.java

```java
class Phone{
    final String company;
    final int launch_year;
    private String chip;
    double price;
    private String imei_no;

    public Phone(String company, int launch_year, double price) {
        this.company = company;
        this.launch_year = launch_year;
        this.price = price;
    }
    public void setCompany(int company){
        this.company = company;
    }
}
public class PhoneInfo{
    public static void main(String[] args) {
        Phone phone = new Phone("Samsung",2024,35000); /// cannot change
        phone.setCompany("Apple");
        phone.setChip = "Helio";
        phone.price = 45000;
        phone.imei_no = 19203192;
        System.out.println(phone.company+" "+phone.launch_year+" "+phone
            .chip+" "+phone.price+" "+phone.getImei_no);
    }
}
```

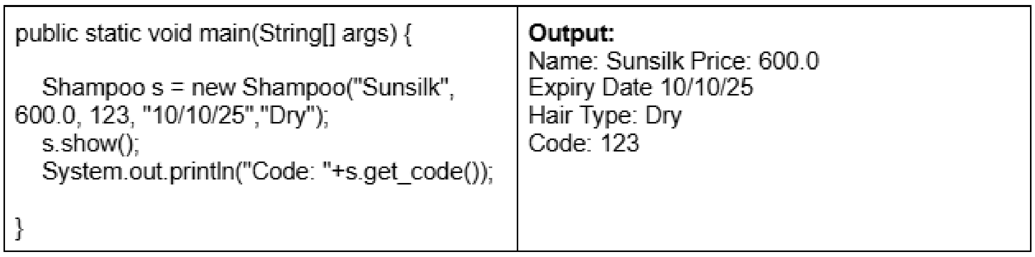
Expected Output: Samsung 2024 Qualcom snapdragon 45000.0 54654684

4. Observe the following diagram representing the relationship between multiple classes: (7)



Write a Java program to implement the given class hierarchy. Your implementations should include constructors and override necessary methods so that the following output can be attained by running the main() function. [**You do not need to write a main function in any of the classes**]

Expected Output



```
public static void main(String[] args) {

   Shampoo s = new Shampoo("Sunsilk",
600.0, 123, "10/10/25","Dry");
   s.show();
   System.out.println("Code: "+s.get_code());

}
```

```
Output:
Name: Sunsilk Price: 600.0
Expiry Date 10/10/25
Hair Type: Dry
Code: 123
```

5. Consider the following Java Program: (6)

B.java

```java
class Demon{
    public String name;
    public int rank;
    public String category = "
       Demon";
    public Demon(String name){
        this.name = name;
    }
    public void summon(){
        System.out.println("Muzan
            summoned " + category
            + " " + rank);
    }
}
class LowerMoon extends Demon{
    public LowerMoon(String name,
        int rank){
        super(name);
        this.rank = rank;
        this.category = "Lower";
    }
}
class UpperMoon extends LowerMoon{
    public UpperMoon(String name, int
       rank){
        super(name, rank);
        this.category = "Upper";
    }
    public void summon(){
        super.summon();
        System.out.println("twanggg");
    }
// add your codes here
}
class Main {
    public static void main(String[]
       args) {
        Demon[] demon = new Demon[4];
        demon[0] = new Demon("Slasher"
           );
        demon[1] = new LowerMoon("Rui"
           , 5);
        demon[2] = new UpperMoon("
           Akaza", 3);
        demon[3] = new UpperMoon("
           Kokushibo", 1);
        for (Demon d : demon)
            d.summon();
        demon[2].summon("Gyomei");
        demon[2].summon(demon[3], "
           Gyomei");
    }
}
```

(a) Find the exact output produced by the **for-each loop**.

(b) Modify only the `UpperMoon` class so that the last two method calls execute successfully and produce the following outputs:

- Muzan summoned Akaza to beat Gyomei
- Akaza was replaced by Kokushibo to beat Gyomei