OOP Practice Problems

Shashank K

SE21UCSE198

CSE 3

Sheet - 1

Q1. Write a program.

```
class Shape{
    public void draw(){
        System.out.println("Shape drawn");
    }
    public void erase(){
        System.out.println("Shape is no more");
    }
}

class Triangle extends Shape{
    // @Override
    public void draw(){
        System.out.println("Triangle drawn");
    }
    // @Override
    public void erase(){
        System.out.println("Triangle is no more");
    }
}
class Circle extends Shape{
    // @Override
    public void draw(){
        System.out.println("Circle drawn");
    }
```

```
    // @Override
    public void erase(){
        System.out.println("Circle is no more");
    }
}
class Square extends Shape{
    // @Override
    public void draw(){
        System.out.println("Square drawn");
    }
    // @Override
    public void erase(){
        System.out.println("Square is no more");
    }
}

class shape_main{
    public static void main(String[] args){
        Shape shape = new Shape();
        shape.draw();
        shape.erase();

        Circle circle = new Circle();
        circle.draw();
        circle.erase();

        Triangle triangle = new Triangle();
        triangle.draw();
        triangle.erase();

        Square square = new Square();
        square.draw();
        square.erase();
    }
}
```

Q2 : Automatic type conversions to overriding.

```
class Base {
    void display() {
        System.out.println("Base class");
    }
}

class Derived extends Base {
    @Override
    void display() {
        System.out.println("Derived class");
    }

    void display(int num) {
        System.out.println("Derived class param: " +
num);
    }
}

public class main_override {
    public static void main(String[] args) {
        Base baseObj = new Derived();
        baseObj.display();
        Derived derivedObj = new Derived();
        derivedObj.display();
        derivedObj.display(42);
    }
}
```

Q3: Boxes question.

```
class Box {
    protected double length;
    protected double breadth;
    protected double height;
    public Box(double length, double breadth, double
height) {
        this.length = length;
        this.breadth = breadth;
        this.height = height;
    }
    public void set(double length, double breadth,
double height) {
        this.length = length;
        this.breadth = breadth;
        this.height = height;
    }
    public double area() {
        return 2 * (length * breadth + breadth * height
+ length * height);
    }
}

class Box3D extends Box {
    public Box3D(double length, double breadth, double
height) {
        super(length, breadth, height);
    }

    public double volume() {
        return length * breadth * height;
    }
}

public class box_main {
```

```
    public static void main(String[] args) {
        Box3D box3D = new Box3D(5.0, 3.0, 2.0);

        double area = box3D.area();
        double volume = box3D.volume();

        System.out.println("Area of the 3D Box: " +
area);
        System.out.println("Volume of the 3D Box: " +
volume);
    }
}
```

Q3. Uppercase convert.

```
import java.util.Scanner;

public class upper_main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Char num : ");
        int numCharacters = scanner.nextInt();
        scanner.nextLine();

        if (numCharacters <= 0) {
                        System.out.println("You broke
the machine :(");
                        scanner.close();
                        return;
        }

        System.out.print("Enter the chars: ");
        String input = scanner.nextLine();

        if (input.length() != numCharacters) {
            System.out.println("You broke the machine
:(");
            scanner.close();
            return;
        }

        String convertedInput = input.substring(0,
numCharacters).toUpperCase();

        System.out.println("Uppercase Conversion Result:
" + convertedInput);

        scanner.close();}}
```

Q5: Read and Write.

```
import java.io.*;

public class read_write_main {
    public static void main(String[] args) {
        try {
            BufferedReader reader = new
BufferedReader(new InputStreamReader(System.in));

            System.out.print("Enter text (type 'exit' to
finish): ");


            FileWriter fileWriter = new
FileWriter("output.txt");
            BufferedWriter writer = new
BufferedWriter(fileWriter);

            String input;
            while (true) {
                input = reader.readLine();
                if (input.equalsIgnoreCase("exit")) {
                    break;
                }
                writer.write(input);
                writer.newLine();
            }

            writer.close();
            System.out.println("Saved to 'output.txt'");
        } catch (IOException e) {
            e.printStackTrace();
        }}}
```

Q6: Area for rectangle

```
class Shape{
    public double getArea(){
       return 0.0;
    }
}

class Rectangle extends Shape{
    double length,breadth;
    public Rectangle(double length,double breadth){
        this.length = length;
        this.breadth = breadth;
    }

    @Override
    public double getArea(){
        return this.length*this.breadth;
    }
}

public class area_main {
    public static void main(String[] args){
    Rectangle rectangle = new Rectangle(10,20);
    System.out.println("Area is : " +
rectangle.getArea());}}
```

Q7: Read from user and throw exception if any duplicates.

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class duplicate_main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter a list of integers
separated by spaces: ");
        String input = scanner.nextLine();

        List<Integer> numbers = new ArrayList<>();

        String[] numberStrings = input.split(" ");
        for (String numberString : numberStrings) {
            try {
                int number =
Integer.parseInt(numberString);
                numbers.add(number);
            } catch (NumberFormatException e) {
                System.out.println("Invalid input");
                return;
            }
        }

        if (hasDuplicates(numbers)) {
            System.out.println("Duplicates detected.");
            throw new
DuplicateNumberException("Duplicate numbers found in the
list.");
        } else {
            System.out.println("List is free of
duplicates");
```

```
        }
    }

    public static boolean hasDuplicates(List<Integer>
numbers) {
        List<Integer> uniqueNumbers = new ArrayList<>();
        for (int number : numbers) {
            if (uniqueNumbers.contains(number)) {
                return true;
            }
            uniqueNumbers.add(number);
        }
        return false;
    }
}

class DuplicateNumberException extends RuntimeException
{
    public DuplicateNumberException(String message) {
        super(message);
    }
}
```

Q8: Banking code.

```
class Box {
    protected double length;
    protected double breadth;
    protected double height;
    public Box(double length, double breadth, double
height) {
        this.length = length;
        this.breadth = breadth;
        this.height = height;
    }
    public void set(double length, double breadth,
double height) {
        this.length = length;
        this.breadth = breadth;
        this.height = height;
    }
    public double area() {
        return 2 * (length * breadth + breadth * height
+ length * height);
    }
}

class Box3D extends Box {
    public Box3D(double length, double breadth, double
height) {
        super(length, breadth, height);
    }

    public double volume() {
        return length * breadth * height;
    }
}

public class box_main {
```

```
    public static void main(String[] args) {
        Box3D box3D = new Box3D(5.0, 3.0, 2.0);

        double area = box3D.area();
        double volume = box3D.volume();

        System.out.println("Area of the 3D Box: " +
area);
        System.out.println("Volume of the 3D Box: " +
volume);
    }
}
```

Q9: Shape with perimeter and area.

```
class Shape{
    public double area(){
        return 0;
    }
    public double perimeter(){
        return 0;
    }
}

class Triangle extends Shape{
    double height;
    double base;
    double a,b,c;

    public Triangle(double height,double base,double
a,double b,double c){
        this.height = height;
        this.base = base;
        this.a = a;
        this.b = b;
        this.c = c;
    }
    @Override
    public double area(){
        return 0.5*this.base*this.height;
    }
    public double perimeter(){
        return this.a+this.b+this.c;
    }
}
class Circle extends Shape{
    double radius;
    public Circle(double radius){
        this.radius = radius;
```

```
    }
    @Override
    public double area(){
        return 3.141*this.radius*this.radius;
    }
    @Override
    public double perimeter(){
        return 2*3.141*this.radius;
    }
}
class Rectangle extends Shape{
    double length;
    double breadth;
    public Rectangle(double length,double breadth){
        this.length = length;
        this.breadth = breadth;
    }
    @Override
    public double area(){
        return this.length*this.breadth;
    }
    @Override
    public double perimeter(){
        return this.length + this.breadth;
    }
}

class area_perimeter{
    public static void main(String[] args){
        Shape shape = new Shape();
        System.out.println("Shape area "+shape.area());
        System.out.println("Shape perimaeter
"+shape.perimeter());

        Circle circle = new Circle(12.7);
```

```
        System.out.println("Circle area
"+circle.area());
        System.out.println("Circle perimeter
"+circle.perimeter());

        Triangle triangle = new
Triangle(4.8,6.3,7.1,8.2,9.5);
        System.out.println("Triangle area
"+triangle.area());
        System.out.println("Triangle perimeter
"+triangle.perimeter());

        Rectangle rectangle = new Rectangle(13.5,12.5);
        System.out.println("Area rect
"+rectangle.area());
        System.out.println("Perimeter rect
"+rectangle.perimeter());
    }
}
```

Q10: compare files lexicographically

```
import java.io.*;

public class compare_main {
    public static void main(String[] args) {
        String file1Path = "file1.txt";
        String file2Path = "file2.txt";

        try {
            boolean areEqual = compareFiles(file1Path,
file2Path);

            if (areEqual) {
                System.out.println("The two files are
equal lexicographically.");
            } else {
                System.out.println("The two files are
not equal lexicographically.");
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static boolean compareFiles(String filePath1,
String filePath2) throws IOException {
        try (BufferedReader reader1 = new
BufferedReader(new FileReader(filePath1));
             BufferedReader reader2 = new
BufferedReader(new FileReader(filePath2))) {

            String line1, line2;

            while ((line1 = reader1.readLine()) != null
&& (line2 = reader2.readLine()) != null) {
```

```
                if (!line1.equals(line2)) {
                    return false;
                }
            }
            return reader1.readLine() == null &&
reader2.readLine() == null;
        }
    }
}
```

Question Sheet 2

Q1: Output of program

A: Output will be 10. ‘a’ is of type A. So when trying to access ‘i’, the ‘i’ in A will be accessed.

Q2: Error in program

A: Multiple inheritance is not accepted in java.

Q3: Output of program

A: The ouput will be 1 2 3. When an instance of C is created, class B is initiated and because of class B, class A is initiated.

Q4: Output of program

A: Output : Class A constructor Class B constructor Class C constructor.

When an instance of C is created, constructor of B is executed and because B is executed it will execute constructor of A.

Q5: Reason for compilation error.

A: Because the data type of ‘z’ is not defined. There is also no semicolon after ‘new Y()’. Another error is declaration of class C ‘Class C’. Another error is class X constructor has no parameters but class Y has a constructor with parameters. This will throw an error.

Q6: Error.

A: Class Y has constructor with no arguments but contructor of X expects an integer as arguement.

Q7: Error.

A: We are using super even though the class doesn’t extend anything.

Q8: Error.

A: No error.

Q9: You know that compiler will implicitly keep super() calling statement as a first statement in every constructor. What happens if we write this() as a first statement in our constructor?

A: Will call constructor of the same class. If there is constructor overloading it will call the constructor according to the arguments given.

Q10: Can a class extend itself?

A: No a class cannot extend itself.

Q11: Does java support multiple inheritance?

A: Yes java supports multiple inheritance as in multiple classes can inherit one class.

Q12: Output of the program.

A: output : 200

Q13: Is the code correct?

A: Yes it is correct. We can define methods in abstract classes but not variables.

Q14: Error?

A: A method can be abstract but it should provide an implementation.

Q15: Is the code correct?

A: SECOND FIRST

Q16: Output?

A: FIRST THIRD SECOND FIRST THIRD THIRD

Q17: What is wrong?

A: We cannot declare instance variables inside interfaces.

Q18: Will it compile or nah?

A: It will compile. X is an interface with method ‘methodX’ and we are defining ‘methodX’ in class Y. So there are no errors.

Q19: Will compile or nah?

A: It will not compile. In interfaces constants are considered ‘final’ automatically so we cannot change it using class B which implements interface A.

Q20: In a class, one method has two overloaded forms. One form is defined as static and another form is defined as non-static. Is that method properly overloaded?

A: Yes a method can be overloaded with two forms even if one is static it will not throw an error.

Q21: Is the method overloaded or dupluicated?

A: It is an example of method duplication. Overloading is based on number of parameters or type of parameters only.

Q22: Discuss trace of execution of program.

A: 1. An instance of class Y is created.

2. An argument is passed to the method.

3 Inside the method overridden and it is converted to double.

4. Prints “THREE”

Q23: Discuss trace of execution of program.

A: 1. Prints 3 to the console.

2. Instance of A is created.

2. Intializations is executed and prints 1

3. public A is executed and prints 2.

Q24: Error?

A: Trying to access variable before it was initialized.

Q25: Trace of the program?

A: 1. Prints 1.

2. Code enter try block.

3. 2 is printed.

3. Attemps to parse “ABC” as an interger which throws an exception.

4. Program catches the exception.

5. catch block is executed and prints 4.

6. Finally is executed and prints out 5.

7. After all the blocs are over it will print 6.