JAVA

Assingment-1

S. Vineela
19BQ1A05J4

1. What is data abstraction? Differentiate data and procedural abstractions. Write inheritance hierarcy for the super class Quadrilateral, Parallelogram, Square and Rectangle. Calculate area of square, rectangle and parallelogram.

ns: Data Abstraction: Abstraction refers to the act of representing essential features without including the background details or explanations. Classes use the concept of abstraction. Abstraction can be achieved with either abstract classes or interfaces. To access the abstract class, it must be inherited from another class.

Difference between Data and Procedural Abstraction

Data Abstraction	Procedural Abstraction
* Data abstraction is a programming methodology in which details of the programing codes are hidden away from the user, and only the essential things are displayed to the user.	* A Procedural abstraction is sequence of operations that has a well-defined effect and can be represented as a single task.
* Abstraction is achieved in either Abstract classes or interface in Java.	* Methods are very good for implementing procedural abstractions.

* It helps the user to avoid writing the low level code, code duplication and increase reusability.

* It normally characterized in a programming language as "function or procedure";

* It makes it easier to read your code, especially if methods have names corresponding to their succinct descriptions.

* It just focusing on operation we focus on data first and then the operations that manipulate the data.

Inheritance Hirachy for the super class Quadrilateral, Parallelogram, square and Rectangle. Programme to calculate area of square, Rectangle and Parallelogram.

```
class Point
{
    private double x;
    private double y;
    public Point(double x, double y)
    {
        this.x = x;
        this.y = y;
    }
    public Point()
    {                    // implicit super constructor.
    }
    public double getX()
    {
        return x;
    }
```

```
public void setX(double x)
{
    this.x = x;
}
public double getY()
{
    return y;
}
public void setY(double y)
{
    this.y=y;
}
}
public class Quadrilateral extends Point
{
    private Point p1;
    private Point p2;
    private Point p3;
    private Point p4;
    public Quadrilateral(double x1, double y1, double x2,
double y2, double x3, double y3, double x4, double y4);
    {
        this.p1 = new Point(x1,y1);
        this.p2 = new Point(x2,y2);
        this.p3 = new Point(x3,y3);
        this.p4 = new Point(x4,y4);
    }
    public Quadrilateral(double x, double y)
    {
        super(x,y);
    }
```

```
public Point getP1()
{
    return p1;
}
public void setP1(Point p1)
{
    this.p1 = p1;
}
public Point getP2()
{
    return p2;
}
public void setP2(Point p2)
{
    this.p2 = p2;
}
public point getP3()
{
    return p3;
}
public void setP3(Point p3)
{
    this.p3 = p3;
}
public point getP4()
{
    return p4;
}
public void setP4(Point p4)
{
    this.p4 = p4;
}
}
```

// Parallalogram

```
import java.text.DecimalFormat;
public class Parallalogram extends Quadrilateral
{
    private double width;
    private double heigth;
    public Parallalogram (double x1, double y1, double x2,
double y2, double x3, double y3, double x4, double y4)
    {
        super( x1, y1, x2, y2, x3, y3, x4, y4 );
        width = Math.sqrt(Math.pow((getP2().getX() - getP1().
getX()),2)+((getP2().getY() - getP1().getY()),2)) ;
        heigth = Math.sqrt(Math.pow((getP4().getX() - getP1.
getX()), 2) +((getP4().getY() - getP1().getY()),2));

    }
    public double area()
    {
        return width* height;

    }
    public String toString() {
    {
        DecimalFormat df = new DecimalFormat (".0");
        return "\n Area is " + df.format(area());
                        of parallalogram
    }

}
```

// Square

```
import java.text.DecimalFormat;
public class Square extends Quadrilateral
{
    private double side;
    public Square (double x1, double y1, double x2, double y2,
double x3, double y3, double x4, double y4);
    {
      super (x1, y1, x2, y2, x3, y3, x4, y4);
      side = Math.sqrt (Math.pow ((getP2().getX() - getP1().
getX()), 2) + ((getP2().getY() - getP1().getY()), 2));

    }
    public double area()
    {
        reture side x side;

    }
    public String toString()
    {
        DecimalFormat df = new DecimalFormat("0.0");
        return "\nArea is :" + df.format(area());
                          of square
    }
}
```

// Rectangle

```
import java.text.DecimalFormat;
public class Rectangle extends Quadrilateral
{
    private double width;
```

```
    private double heigth;
    public Rectangle (double x1, double y1, double x2,
double y2, double x3, double y3, double x4, double y4);
    {
        super (x1, y1, x2, y2, x3, y3, x4, y4);
        width = Math.sqrt((Math.pow((getP2().getX - getP1()
getX()),2) + Math.pow((getP2().getY - getP1().getY()),2));
        heigth = Math.sqrt(Math.pow((getP4().getX-getP1().
getX()),2) + Math.pow((getP4().getY() - getP1().getY()),2));
    }
    public double area()
    {
        return width * height;
    }
    public String toString()
    {
        DecimalFormat df = new DecimalFormat(".0");
        return "\nArea is : " + df.format(area());
                    of rectangle
    }
}
// Main
Class QuadrilateralTest
{
    public static void main (String[] args)
    {
        Parallalogram parallelogram = new Parallalogram
(5.0, 5.0, 11.0, 5.0, 12.0, 20.0, 6.0, 20.0);
```

```
        Rectangle rectangle = new Rectangle (17.0, 14.0,
30.0, 14.0, 30.0, 28.0, 17.0, 28.0);
        Square square = new Square (4.0, 0.0, 8.0, 0.0, 8.0,
4.0, 4.0, 4.0);
        System.out.printf ("%s %s %s \n", parallelogram,
rectangle, square);
    }
}
```

Output :

Area is : 90.2 of parallelogram
Area is : 182.0 of rectangle
Area of square is : 16.0

Area of parallalogram is : 90.2

Area of rectangle is : 182.0

Area of square is : 16.0

2. What is the importance of constructor? Write a java Program to perform constructor overloading. Describe the usuage of static member and nesting members with suitable example programs in java.

Ans:- Constructor :- Constructor in java is a special type of method that is used to initialize the object. Java constructor is invoked at the time of object creation. It constructs the values i.e provides data for the object that is why it is known as constructor.

Importance of Constructor :-

* A constructor eliminates placing the default values.
* A constructor eliminates calling the normal method implicity.

Constructor overloading :- A constructor can also be overloaded. Overloaded constructors are differentiated on the basis of their type of parameters or number of parameters. Constructor overloading is not much differen than method overloading. In case of method overloading you have multi methods with same name but different signatures, where as in Constructor Overloading you have multi constructor with different signature but only difference is that constructor doesn't have "return" type in Java. Constructor overloading is done to construct object in different ways.

Program to perform Constructor Overloading:-

```
Class Student
{
    int Roll;
    String Name;
    double Marks;
    Student (int R, String N, double M) || Constructor 1
    {
        Roll = R;
        Name = N;
        double =1
        Marks = M;
    }
    Student (String N, double M, int R) || Constructor 2
    {
        Roll = R;
        Name = N;
        Marks = M;
    }
    void Display ()
    {
        System.out.print ("\n\t" + Roll + "\t" + Name
                          + "\t" + Marks);
    }
}
Class Constructor Overloading Demo
{
    public static void main (String[] args)
    {
        Student S1 = new Student (1, "Ravi", 78.53);
        Student S2 = new Student ("Sumit", 89.42, 2);
        System.out.print ("\n\t Roll \t Name \t Marks \n");
```

```
        S1.Display();
        S2.Display();
    }
}
```

Output:

Roll	Name	Marks
1	kumar	78.53
2	Sumit	89.42

Static members: In Java, static members are those which belongs to the class and you can access these members without instantiating the class. To create a static member precede its declaration with the keyword "static". When a member is declared static, it can be accessed before any objects of its class are created, and without reference to any object.

Program for static member

```
class Test
{
    static void display()
    {
        System.out.println("Hello Java Project");
    }
    public static void main(String[] args)
    {
        display();
    }
}
```

Output :

Hello Java Project

Nested members :- The Java programming language allows you to define a class within another class. Such a class is clalled a Nested class.

* It is a way of logically grouping classes that are only used in one place.
* It increases encapsulation.
* It can lead to more readable and maintainable code.

Program for Nested member :-

```
class Outer_Test
{
    int num;
    private class Inner_Test
    {
        public void print()
        {
            System.out.println("This is an Our inner
                                class");
        }
    }
    void display_Inner()
    {
        Inner_Test inner = new Inner_Test();
        inner.print();
    }
}
public class My_class
{
    public static void main(String args[])
    {
```

```
        Outer_Test outer = new Outer_Test();
        outer.display_Inner();
    }
}
```

Output:

This is an Our inner class.

3. Define a class named BookFair with the following description;

Instance variables/Data members;

String Bname - stores the name of the book

double price - stores the price of the book.

Member Methods;

(i) BookFair() - Default constructor to initalize data members

(ii) void Input() - To input and store the name and the price of the book.

(iii) void calculate() - To calculate the price after discount. Discount is calculated based on the following criteria.

Price	Discount
Less than or equal to RS 1000	2% of price
More than RS 1000 and less than or equal to RS 3000	10% of price
More than RS 3000	15% of price

Write a main method to creat an object of the class and call the above member methods.

Ans:

```
public class BookFair
{
    static void voidInput (String Bname, double price)
    {
        System.out.println ("Rice of the Book " + Bname +
                            " before discount is " + price);

        double amount;
        amount = voidCalculate (price);
        System.out.println (" Rice of the Book " + Bname +
                            " before after discount is " + amount);
    }
    static double voidCalculate (double m)
    {
        double discountPrice;
        if (m<=1000)
        {
            discountPrice = (m*2)/100;
            return discountPrice;
        }
        else if (m>1000 && m<=3000)
        {
            discountPrice = (m*10)/100;
            return discountPrice;
        }
        else
        {
            discountPrice = (m*15)/100;
            return discountPrice;
        }
    }
```

```
    public static void main (String [] args)
    {
        BookFair bf = new BookFair();
        bf.voidInput ("Maths", 5000.5);
    }
}
```

Output :

Price of the Book Maths before discount is 5000.5

Price of the Book Maths after discount is 750.075

4. Special words are those words which starts and ends with the same letter.

Examples:

EXISTENCE, COMIC, WINDOW

Palindrome words are those words which read the same from left to right and vice-versa.

Examples :

MALAYALAM, MADAM, LEVEL, ROTATOR, CIVIC

All palindromes are special words, but all speciel words are not palindromes.

Write a program to accept a word check and print whether the word is a palindrome or only special word.

Ans

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

```
System.out.println ("Enter the word");
String a = new String ("MALAYALAM")
String a = new
String a = new String ("MALAYALAM");
int n = a.length();
int PS = 0;
int P = 0;
int S = 0;
for (int i = 0; i <= (n/2); i++)
{
    for (int j = n-1; j >= (n/2); j--)
    {
        if (a.charAt(i) == a.charAt(j) &&
            a.charAt(0) == a.charAt(n-1))
        {
            PS = 1;
        }
        else if (a.charAt(i) == a.charAt(j))
        {
            P = 1;
        }
        else if (a.charAt(0) == a.charAt(n-1))
        {
            S = 1;
        }
    }
}
if (PS == 1)
    System.out.println (a + " is both pallindrom
                and special");
else if (P == 1)
```

```
            System.out.println(a+" only pallindrom");
        else if(s==1)
            System.out.println(a+" is only special");
    }
}
```

Output :-

```
Enter the word
MALAYALAM is both pallindrom and special
```