1) Write An example implementing overriding and overloading (Polymorphism)?

Answer (a): Overriding (Polymorphism)

class vehicle{

String info(){

return "This is the base class vehicle";

}

}

class Car extends vehicle{

String info(){

return "This is Child class Car";

}

}

class Bus extends vehicle{

String info(){

return "This is Child class Bus";

}

}

public class Answer1a{

public static void main(String[] args){

Car mes=new Car();

System.out.println(mes.info());

Bus nes=new Bus();

System.out.println(nes.info());

}

}

Answer (b): Overloading (Polymorphism)

public class Answer1b{

public static void main(String[] args){

System.out.println(multiplication(3,4));

}

public static int multiplication(int a, int b){

return a\*b;

}

public static double multiplication(double a, double b){

return a\*b;

}

}

2) The Guessing Game:

The guessing game involves a 'game' object and three 'player' objects. The game generates a random number between

0 and 9, and the three player objects try to guess it.

Write a program for the game.

Classes:

GuessGame.class

Player.class

GameLauncher.class

The logic:

i. The GameLauncher class is where the application starts; it has the main method.

ii. In the main() method a GuessGame object is created, and its startGame() method is called.

iii. The GuessGame objects startGame() method is where the entire game plays out.

It creates three playerd, then thinkd of a random number(the target). It then asks each player to guess, checks the

result, and either prints out information about the winning players or asks them to guess again.

Answer:

class GuessGame{

Player p1=new Player();

Player p2=new Player();

Player p3=new Player();

public void startGame(){

System.out.println("I am thinking of a number between 0 and 9");

int target=(int)(Math.random()\*10);

System.out.println("Number to guess "+target);

boolean p1\_right=false;

boolean p2\_right=false;

boolean p3\_right=false;

while(true){

p1.guess();

p2.guess();

p3.guess();

System.out.println("Player 1 guessed: "+p1.number);

System.out.println("Player 2 guessed: "+p2.number);

System.out.println("Player 3 guessed: "+p3.number);

if(target==p1.number){

p1\_right=true;

}

if(target==p2.number){

p2\_right=true;

}

if(target==p3.number){

p3\_right=true;

}

if(p1\_right||p2\_right||p3\_right){

System.out.println("\nWe have winner!!");

System.out.println("Player 1 guessed it right?: "+p1\_right);

System.out.println("Player 2 guessed it right?: "+p2\_right);

System.out.println("Player 3 guessed it right?: "+p3\_right);

System.out.println("Game over.");

break;

}

else

System.out.println("Players have to guess again\n");

}

}

}

class Player{

int number;

void guess(){

number=(int)(Math.random()\*10);

}

}

public class GameLauncher{

public static void main(String[] args){

GuessGame round=new GuessGame();

round.startGame();

}

}

3) Write an inheritance hierarchy for classes Quadrilateral , Trapezoid , Parallelogram ,

Rectangle and Square . Use Quadrilateral as the superclass of the hierarchy. Create and use a Point

class to represent the points in each shape. Make the hierarchy as deep (i.e., as many levels) as possible. Specify the instance variables and methods for each class. The private instance variables of

Quadrilateral should be the x-y coordinate pairs for the four endpoints of the Quadrilateral.

Write a program that instantiates objects of your classes and outputs each object’s area (except Quadrilateral)

Answer:

**Point.java**

public class Point {

private double x;

private double y;

public Point(double x, double y) {

this.x = x;

this.y = y;

}

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;

}

@Override

public String toString() {

return "Point{" + "x=" + x + ", y=" + y +'}';

}

}

**Quadrilateral.java**

public class Quadrilateral {

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 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;

}

public double getArea() {

return 0;

}

@Override

public String toString() {

return "{" + "p1=" + p1 + ", p2=" + p2 + ", p3=" + p3 + ", p4=" + p4 + '}';

}

}

**Trapezoid.java**

public class Trapezoid extends Quadrilateral {

public Trapezoid(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);

}

@Override

public double getArea() {

double base1 = Math.sqrt(Math.pow(getP1().getX() - getP2().getX(), 2) + Math.pow(getP1().getY() - getP2().getY(), 2));

double base2 = Math.sqrt(Math.pow(getP3().getX() - getP4().getX(), 2) + Math.pow(getP3().getY() - getP4().getY(), 2));

double height = Math.sqrt(Math.pow(getP1().getX() - getP1().getX(), 2) + Math.pow(getP1().getY() - getP4().getY(), 2));

return (base1 + base2) \* height / 2;

}

public String Answer(){

return "\nCoordinates of Trapezoid are:\n" + super.toString()

+ "\nArea is :" + getArea();

}

}

**Parallelogram.java**

public class Parallelogram extends Trapezoid {

public Parallelogram(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);

}

@Override

public double getArea() {

double base = Math.sqrt(Math.pow(getP1().getX() - getP2().getX(), 2) + Math.pow(getP1().getY() - getP2().getY(), 2));

double height;

if(getP1().getY()==getP2().getY()){

height = Math.sqrt(Math.pow(getP4().getX() - getP4().getX(), 2) + Math.pow(getP1().getY() - getP4().getY(), 2));

}

else{

height = Math.sqrt(Math.pow(getP4().getX() - getP4().getX(), 2) + Math.pow(getP1().getY() - getP4().getY(), 2));

}

return base \* height;

}

public String Answer(){

return "\nCoordinates of Parallelogram are:\n" + super.toString()

+ "\nArea is :" + getArea();

}

}

**Rectangle.java**

public class Rectangle extends Parallelogram {

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);

}

@Override

public double getArea() {

double base = Math.sqrt(Math.pow(getP1().getX() - getP2().getX(), 2) + Math.pow(getP1().getY() - getP2().getY(), 2));

double height = Math.sqrt(Math.pow(getP2().getX() - getP3().getX(), 2) + Math.pow(getP2().getY() - getP3().getY(), 2));

return base \* height;

}

public String Answer(){

return "\nCoordinates of Rectangle are:\n" + super.toString()

+ "\nArea is :" + getArea();

}

}

**Square.java**

public class Square extends Rectangle {

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);

}

@Override

public double getArea() {

double base = Math.sqrt(Math.pow(getP1().getX() - getP2().getX(), 2) + Math.pow(getP1().getY() - getP2().getY(), 2));

return base \* base;

}

public String Answer(){

return "\nCoordinates of Square are:\n" + super.toString()

+ "\nArea is :" + getArea();

}

}

**Main.java**

public class Main {

public static void main(String[] args) {

Trapezoid trapezoid = new Trapezoid(5,5,10,5,8,10,7,10);

Parallelogram parallelogram = new Parallelogram(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.println(trapezoid);

// System.out.println(parallelogram);

// System.out.println(rectangle);

// System.out.println(square);

System.out.println(trapezoid.Answer());

System.out.println(parallelogram.Answer());

System.out.println(rectangle.Answer());

System.out.println(square.Answer());

}

}