Design a class **Point** with two data members **x-cord** and **y-cord**. This class should have an arguments **constructor**, **setters**, **getters** and a **display** function.

Now create another class ― **Line**, which contains two Points ― **startPoint** and ― **endPoint**. It should have a function that finds the **length** of the line.

Hint: formula is: sqrt((x2-x1)2 + (y2-y1)2)

Create two line **objects** in **runner** and display the length of each line.

Design a class Point with two data members x-cord and y-cord. This class should have an arguments constructor, setters, getters and a display function.

// Point class

class Point {

private double xCord;

private double yCord;

// Constructor with arguments

public Point(double x, double y) {

this.xCord = x;

this.yCord = y;

}

// Setter for x coordinate

public void setXCord(double x) {

this.xCord = x;

}

// Setter for y coordinate

public void setYCord(double y) {

this.yCord = y;

}

// Getter for x coordinate

public double getXCord() {

return this.xCord;

}

// Getter for y coordinate

public double getYCord() {

return this.yCord;

}

// Display function

public void display() {

System.out.println("Point (" + this.xCord + ", " + this.yCord + ")");

}

}

Now create another class ―Line‖, which contains two Points ―startPoint‖ and ―endPoint‖. It should have a function that finds the length of the line. Hint: formula is: sqrt((x2-x1)2 + (y2-y1)2) Create two line objects in runner and display the length of each line.

// Line class

class Line {

private Point startPoint;

private Point endPoint;

// Constructor with two Points

public Line(Point start, Point end) {

this.startPoint = start;

this.endPoint = end;

}

// Method to calculate the length of the line

public double calculateLength() {

double xDiff = endPoint.getXCord() - startPoint.getXCord();

double yDiff = endPoint.getYCord() - startPoint.getYCord();

return Math.sqrt(Math.pow(xDiff, 2) + Math.pow(yDiff, 2));

}

// Display function

public void displayLength() {

System.out.println("Length of the line: " + calculateLength());

}

}

// Runner class to test the functionality

public class Runner {

public static void main(String[] args) {

// Creating two points for the first line

Point p1 = new Point(1, 2);

Point p2 = new Point(4, 6);

// Creating first line

Line line1 = new Line(p1, p2);

// Displaying the length of the first line

line1.displayLength();

// Creating two points for the second line

Point p3 = new Point(3, 4);

Point p4 = new Point(7, 8);

// Creating second line

Line line2 = new Line(p3, p4);

// Displaying the length of the second line

line2.displayLength();

}

}

**Example 02**

Create a class named **Pizza** that stores information about a single pizza. It should contain the following:

Private instance variables to store the **size** of the pizza (either small, medium, or large), the number of cheese **toppings**, the number of pepperoni toppings, and the number of ham toppings.

**Constructor**(s) that set all of the instance variables.

Public methods to **get** and **set** the instance variables.

A public method named **calcCost**( ) that returns a double that is the cost of the pizza. Pizza cost is determined by:

Small: $10 + $2 per topping Medium: $12 + $2 per topping Large: $14 + $2 per topping

public method named **getDescription**( ) that returns a String containing the pizza size, quantity of each topping.

// Pizza class

class Pizza {

// Private instance variables

private String size; // small, medium, or large

private int cheeseToppings;

private int pepperoniToppings;

private int hamToppings;

// Constructor to set all the instance variables

public Pizza(String size, int cheeseToppings, int pepperoniToppings, int hamToppings) {

this.size = size.toLowerCase();

// Convert size to lowercase to avoid case mismatch

this.cheeseToppings = cheeseToppings;

this.pepperoniToppings = pepperoniToppings;

this.hamToppings = hamToppings;

}

// Getters

public String getSize() {

return size;

}

public int getCheeseToppings() {

return cheeseToppings;

}

public int getPepperoniToppings() {

return pepperoniToppings;

}

public int getHamToppings() {

return hamToppings;

}

// Setters

public void setSize(String size) {

this.size = size;

}

public void setCheeseToppings(int cheeseToppings) {

this.cheeseToppings = cheeseToppings;

}

public void setPepperoniToppings(int pepperoniToppings) {

this.pepperoniToppings = pepperoniToppings;

}

public void setHamToppings(int hamToppings) {

this.hamToppings = hamToppings;

}

Write test code to create several pizzas and output their descriptions. For example, a large pizza with one cheese, one pepperoni and two ham toppings should cost a total of $22.

// Method to calculate the cost of the pizza

public double calcCost() {

double baseCost;

switch (size) {

case "small":

baseCost = 10;

break;

case "medium":

baseCost = 12;

break;

case "large":

baseCost = 14;

break;

default:

baseCost = 0; // default to 0 if invalid size

}

// Adding $2 per topping

int totalToppings = cheeseToppings + pepperoniToppings + hamToppings;

return baseCost + (totalToppings \* 2);

}

// Method to get the description of the pizza

public String getDescription() {

return "Size: " + size + ", Cheese Toppings: " + cheeseToppings +

", Pepperoni Toppings: " + pepperoniToppings +

", Ham Toppings: " + hamToppings +

", Cost: $" + calcCost();

}

}

Now Create a **PizzaOrder** class that allows up to three pizzas to be saved in an order. Each pizza saved should be a Pizza object. Create a method **calcTotal**() that returns the cost of order.

// PizzaOrder class

class **PizzaOrder** {

// Array to hold up to three Pizza objects

private Pizza[] pizzas;

private int pizzaCount;

// Constructor to initialize the array

public PizzaOrder() {

pizzas = new Pizza[3];

// Maximum of 3 pizzas

pizzaCount = 0;

}

// Method to add a pizza to the order

public void addPizza(Pizza pizza) {

if (pizzaCount < 3) {

pizzas[pizzaCount] = pizza;

pizzaCount++;

} else {

System.out.println("Cannot add more than 3 pizzas to the order.");

}

}

// Method to calculate the total cost of the order

public double calcTotal() {

double totalCost = 0;

for (int i = 0; i < pizzaCount; i++) {

totalCost += pizzas[i].calcCost();

}

return totalCost;

}

}

**In the runner order two pizzas and return the total cost.**

// Runner class to test the functionality

public class **Runner** {

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

// Creating first pizza (Large, 1 cheese, 1 pepperoni, 2 ham)

Pizza pizza1 = new Pizza("large", 1, 1, 2);

System.out.println("Pizza 1: " + pizza1.getDescription());

// Creating second pizza (Medium, 2 cheese, 0 pepperoni, 1 ham)

Pizza pizza2 = new Pizza("medium", 2, 0, 1);

System.out.println("Pizza 2: " + pizza2.getDescription());

// Creating a PizzaOrder and adding the two pizzas

**PizzaOrder** order = new PizzaOrder();

order.addPizza(pizza1);

order.addPizza(pizza2);

// Display the total cost of the order

System.out.println("Total Order Cost: $" + order.calcTotal());

}

}