

**UNIVERSIDAD AUTÓNOMA “TOMÁS FRÍAS”
FACULTAD DE INGENIERIA
CARRERA DE INGENIERIA EN SISTEMAS**



DOCENTE:

**NOMBRES Y APELLIDOS: VALENTIN
ALEXANDER CARPIO GUZMAN**

Gestión 2024

Potosí- Bolivia


PROGRAMACIÓN 2 TAREA NRO 1

Ejercicio 1

Escribe un programa que solicite las coordenadas de dos puntos en el espacio tridimensional (x1, y1, z1) y (x2, y2, z2) y calcule la distancia entre ambos puntos utilizando la fórmula:

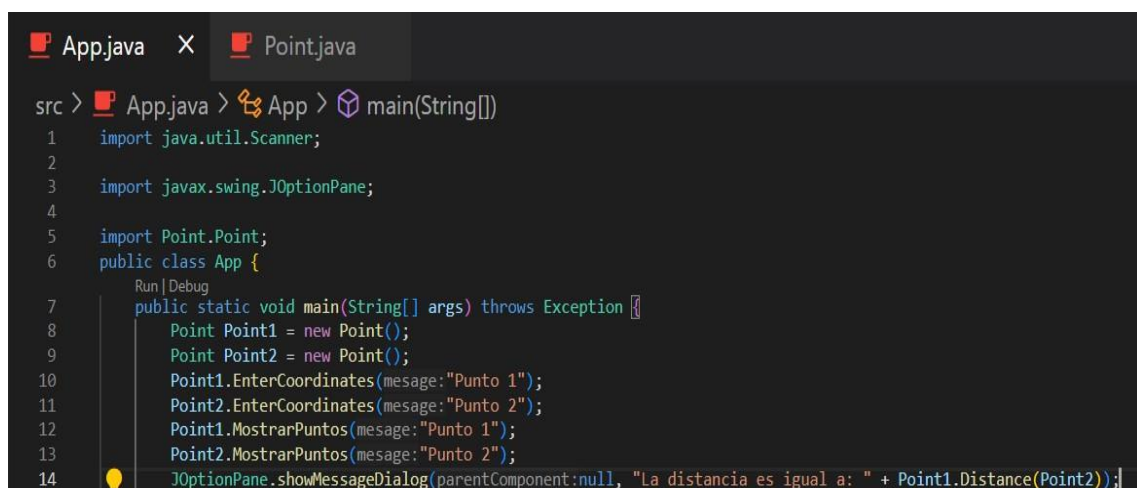
Solución

creamos un paquete con la clase punto y los métodos para llenado de los puntos y el calculo de su distancia



```
src > Point > Point.java > Point
1 package Point;
2
3 import javax.swing.JOptionPane;
4
5 public class Point {
6     float CoordinateX;
7     float CoordinateY;
8     float CoordinateZ;
9     public void EnterCoordinates(String message) {
10         CoordinateX = Float.parseFloat(JOptionPane.showInputDialog(message + "\n Escribir coordenada x"));
11         CoordinateY = Float.parseFloat(JOptionPane.showInputDialog(message + "\n Escribir coordenada y"));
12         CoordinateZ = Float.parseFloat(JOptionPane.showInputDialog(message + "\n Escribir coordenada z"));
13     }
14     public void MostrarPuntos(String message){
15         JOptionPane.showMessageDialog(parentComponent:null, message + "\n(" + CoordinateX + "," + CoordinateY + "," + CoordinateZ + ")");
16     }
17     public double Distance(Point point){
18         return Math.sqrt(Math.pow((point.CoordinateX - CoordinateX), b:2) + Math.pow((point.CoordinateY - CoordinateY), b:2) + Math.pow((point.CoordinateZ - CoordinateZ), b:2));
19     }
20 }
```

Importamos el paquete y llamamos a los métodos



```
src > App.java > App > main(String[])
1 import java.util.Scanner;
2
3 import javax.swing.JOptionPane;
4
5 import Point.Point;
6 public class App {
7     Run | Debug
8     public static void main(String[] args) throws Exception {
9         Point Point1 = new Point();
10        Point Point2 = new Point();
11        Point1.EnterCoordinates(message:"Punto 1");
12        Point2.EnterCoordinates(message:"Punto 2");
13        Point1.MostrarPuntos(message:"Punto 1");
14        Point2.MostrarPuntos(message:"Punto 2");
15        JOptionPane.showMessageDialog(parentComponent:null, "La distancia es igual a: " + Point1.Distance(Point2));
16    }
```

Iniciamos


1. Introducción de puntos

Entrada ✕

 Punto 1
Escribir coordenada x


5

Entrada ✕

 Punto 1
Escribir coordenada y


8

Entrada ✕

 Punto 1
Escribir coordenada z


3

Entrada ✕

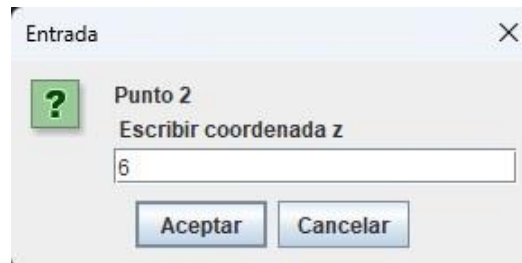
 Punto 2
Escribir coordenada x

3

Entrada ✕

 Punto 2
Escribir coordenada y

11



2. Mostrar los puntos



3. Calculo de la distancia de los 2 puntos



Ejercicio 2

Simula el lanzamiento de dos dados y calcula la probabilidad de que la suma de los valores de ambos dados sea mayor o igual a un número dado por el usuario. Utiliza la función `Math.random` para generar los números aleatorios.

Solución

Creemos un método para simular el lanzamiento de un dado

```
public static double Dado(){  
    return (int)(Math.random()*(6 - 1) + 1);  
}
```

Creamos un método que calcule la probabilidad de un número N para el lanzamiento de dos dados

```
public static double ProbabilityOF2DiceRoll(int Number) {
    int FavorableCase = 0;
    for (int i = 1; i <= 6; i++) {
        for (int j = 1; j <= 6; j++) {
            if (Number == (i + j)) {
                FavorableCase += 1;
            }
        }
    }
    return FavorableCase * Math.pow(36, -1);
}
```

(el método .equals() no me funciona para colocar como condicional para este caso, por eso uso el "==")

Creamos un método que sume las probabilidades iguales y mayores que el número que introduciremos

```
public static double SumOfProbability(int Number){
    double TotalProbability = 0;
    for (int i = Number; i <= 12; i++) {
        TotalProbability = TotalProbability + (ProbabilityOF2DiceRoll(i));
    }
    return TotalProbability;
}
```

Iniciamos un scanner para introducir el valor de N por consola

```
try(Scanner Variables = new Scanner(System.in)){
```

Creamos la variable N donde se almacenará el número que introducimos por consola y llamamos al método dado 2 veces para simular los lanzamientos de los dados

```
System.out.println("simulacion del tiro de 2 dados");
System.out.println(Dado());
System.out.println(Dado());
int Number;
System.out.println("introduzca el numero tope: ");
Number = Variables.nextInt();
System.out.println("la probabilidad de que la suma de los 2 lanzamientos salga tu numero es: " + SumOfProbability(Number));
```


Iniciamos

```
simulacion del tiro de 2 dados
5.0
2.0
introduzca el numero tope:
7
la probabilidad de que la suma de los 2 lanzamientos salga tu numero es: 0.5833333333333334
```

Ejercicio 3

Escribe un programa que resuelva una ecuación cuadrática de la forma $ax^2+bx+c=0$ utilizando la fórmula general:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Solución

Creamos un metodo que resuelva la ecuación cuadrada, usando la formula general

```
public static void SolveSquareFunction(int cuadratico ,int lineal ,int independiente){
    double solve1 = ((-lineal + Math.sqrt(Math.pow(lineal, 2) - 4*cadratico*independiente))/(2*cadratico));
    double solve2 = ((-lineal - Math.sqrt(Math.pow(lineal, 2) - 4*cadratico*independiente))/(2*cadratico));
    System.out.println("solve1: " + solve1);
    System.out.println("solve2: " + solve2);
}
```

Usando el scanner pedimos por consola los términos de la ecuación cuadrática y llamamos al método introduciendo los términos

```
try(Scanner Variables = new Scanner(System.in)){
    int QuadraticTerm;
    System.out.println(x:"digite el numero cuadratico: ");
    QuadraticTerm = Variables.nextInt();
    int LinealTerm;
    System.out.println(x:"digite el numero lineal: ");
    LinealTerm = Variables.nextInt();
    int IndependentTerm;
    System.out.println(x:"digite el numero independiente: ");
    IndependentTerm = Variables.nextInt();
    SolveSquareFunction(QuadraticTerm, LinealTerm, IndependentTerm);
}
```

Iniciamos

```
digite el numero cuadratico:
5
digite el numero lineal:
12
digite el numero independiente:
5
solve1: -0.53667504192892
solve2: -1.86332495807108
```

Ejercicio 4

Escribe un programa que calcule una aproximación del valor de PI utilizando la serie infinita de

$$\pi = 4 \left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \dots \right)$$

Permite que el usuario ingrese el número de términos a utilizar en la aproximación. Utiliza Math.pow.

Solución

Creemos un método que nos desarrolle la serie que usaremos para calcular la aproximación de pi

```
public static double SeriePi(int Termin){
    double Summation = 0;
    for (int i = 1; i < Termin; i++) {
        Summation = Summation + (Math.pow(-1, (i + 1))) * (1 * Math.pow((2 * i) - 1, -1));
    }
    return Summation;
}
```

Pedimos por consola el número de términos que tendrá la serie y llamamos al método para su cálculo de la aproximación de pi

```
int Termin;
System.out.println(x:"introduzca el numero de terminos para la serie pi: ");
Termin = Variables.nextInt();
System.out.println("la aproximacion de pi es: " + SeriePi(Termin)*4);
```

Iniciamos

```
introduzca el numero de terminos para la serie pi:  
1000  
la aproximacion de pi es: 3.142593654340044
```

Ejercicio 5

Dado un punto en coordenadas polares (r , θ), conviértelo a coordenadas cartesianas (x , y) utilizando las siguientes fórmulas:

$$x = r \cdot \cos(\theta)$$
$$y = r \cdot \sin(\theta)$$

Solución

Creemos 2 metodos para las transformadas de las coordenadas "x" y "y"

```
public static double TransformLinealX(float Ratio, float Angle){  
    return (Math.abs(Ratio)) * Math.cos((Math.abs(Angle) * Math.PI));  
}  
public static double TransformLinealY(float Ratio, float Angle){  
    return (Math.abs(Ratio)) * Math.sin((Math.abs(Angle) * Math.PI));  
}
```

Introducimos por consola el valor del radio y del Angulo y llamamos a los métodos para transformar a coordenadas cartesianas

```
float Ratio;  
System.out.println(x:"introduzca la coordenada radio: ");  
Ratio = Variables.nextFloat();  
float Angle;  
System.out.println(x:"introduzca el angulo en pi radianes: ");  
Angle = Variables.nextFloat();  
System.out.println(TransformLinealX(Ratio, Angle));  
System.out.println(TransformLinealY(Ratio, Angle));
```

Iniciamos


```
introduzca la coordenada radio:
5
introduzca el angulo en pi radianes:
1
-5.0
6.123233995736766E-16
```

Ejercicio 7

Escribe un programa que calcule el monto final después de aplicar interés compuesto, utilizando la fórmula:

$$A = P \left(1 + \frac{r}{n}\right)^{nt}$$

Donde:

- A es el monto final,
- P es el monto principal,
- r es la tasa de interés anual,
- n es el número de veces que se aplica el interés por año,
- t es el tiempo en años.

Utiliza Math.pow.

Solución

Creemos un metodo que calcule el monto final del interés compuesto

```
public static double CompountInterest(float PrincMount, float Rate, int Applicable, int Time){
    return PrincMount * (Math.pow((1 - (Rate/Applicable)), (Applicable * Time)));
}
```

Introducimos por consola todos los valores que nos pide el método y llamamos al método

```

System.out.println(x:"introduzca el monto principal: ");
PrincipalMount = Variables.nextFloat();
float InterestRate;
System.out.println(x:"introduzca la tasa del interes que desea aplicar (en decimal 1% = 0.01): ");
InterestRate = Variables.nextFloat();
int ApplicableOfInterest;
System.out.println(x:"introduzca el numero de veces que aplicara el interes: ");
ApplicableOfInterest = Variables.nextInt();
int Time;
System.out.println(x:"introduzca el tiempo en years: ");
Time = Variables.nextInt();
System.out.println(CompountInterest(PrincipalMount, InterestRate, ApplicableOfInterest, Time));

```

Iniciamos

```

introduzca el monto principal:
1000
introduzca la tasa del interes que desea aplicar (en decimal 1% = 0.01):
0,15
introduzca el numero de veces que aplicara el interes:
5
introduzca el tiempo en years:
10
218.0656969404458

```

Ejercicio 9

Escribe un programa que calcule el MCD de dos números enteros utilizando el algoritmo de Euclides. Utiliza Math.abs para asegurar que siempre se trabaje con valores positivos.

Solución

Creemos un método que nos de un numero aleatorio entre un rango que nosotros pediremos por consola

```

public static double Random(int Maximum, int Minimum){
    return (int)(Math.random()*(Maximum - Minimum) + Minimum);
}

```

Introducimos el rango por consola y llamamos al metodo

```
int Maximum;
System.out.println(x:"introduzca la cota maxima: ");
Maximum = Variables.nextInt();
int Minimum;
System.out.println(x:"introduzca la cota minima: ");
Minimum = Variables.nextInt();
System.out.println(Random(Maximum, Minimum));
```

Iniciamos

```
introduzca la cota maxima:
11
introduzca la cota minima:
5
9.0
```

(si introducimos el mínimo primero y después el máximo el numero aleatorio generado seguirá dentro del rango)

Ejercicio 10

Escribe un programa que calcule el área y el volumen de una esfera, dado su radio. Utiliza las siguientes fórmulas:

$$\text{Área: } A = 4\pi r^2$$

$$\text{Volumen: } V = \frac{4}{3}\pi r^3$$

Utiliza Math.PI y Math.pow.

Solución

Creemos 2 métodos para el cálculo del área y el volumen de una esfera

```
public static double Area(float radio){
    return 4 * Math.PI * Math.pow(radio, b:2);
}
public static double Volume(float radio){
    return (4 * Math.PI * Math.pow(radio, b:3))/3;
}
```

Introducimos el radio por consola y llamamos a los metodos

```
float ratio;  
System.out.println(x:"coloque el radio de la esfera: ");  
ratio = Variables.nextFloat();  
System.out.println(Area(ratio));  
System.out.println(Volume(ratio));
```

Iniciamos

```
coloque el radio de la esfera:  
10  
1256.6370614359173  
4188.790204786391
```