Лабораторная работа № 2

по дисциплине

“Объектно-ориентированное программирование”

Выполнил студент

группы БФИ1901

Бардюк Д.В.

Москва 2020

**Цель работы:** Java позволяет использовать объекты. В данной лабораторной работе необходимо использовать классы по одному на файл, чтобы описать, как эти объекты работают.

**Задание:**

1. Создайте новый класс Point3d для представления точек в трехмерном Евклидовом пространстве. Необходимо реализовать:

• создание нового объекта Point3d с тремя значениями с плавающей точкой (double);

• создание нового объекта Point3d со значениями (0.0, 0.0, 0.0) по умолчанию,

• возможность получения и изменения всех трех значений по отдельности; • метод для сравнения значений двух объектов Point3d. Нельзя предоставлять непосредственный доступ к внутренним элементам объекта класса Point3d .

2. Добавьте новый метод distanceTo, который в качестве параметра принимает другой объект Point3d, вычисляет расстояние между двумя точками с точность двух знаков после запятой и возвращает полученное значение.

3. Создайте другой класс под названием Lab1, который будет содержать статический метод main. Помните, что метод main должен быть общедоступным (public) с возвращаемым значением void, а в качестве аргумента должен принимать строку (String). Этот класс должен иметь следующую функциональность:

• Ввод координат трех точек, находящихся в трехмерном пространстве. Создание трех объектов типа Point3d на основании полученных данных. (Предполагается, что пользователь вводит корректные данные.)

• Создайте второй статический метод computeArea, который принимает три объекта типа Point3d и вычисляет площадь треугольника, образованного этими точками. (Вы можете использовать формулу Герона.) Верните получившееся значение площади в формате типа double.

• На основе полученных данных и с использованием реализованного алгоритма посчитайте площадь и выведите полученное значение пользователю. Перед вызовом метода computeArea проверьте на равенство значений всех трех объектов Point3d. Если одна из точек равна другой, то выведите соответствующее сообщение пользователю и не вычисляйте площадь.

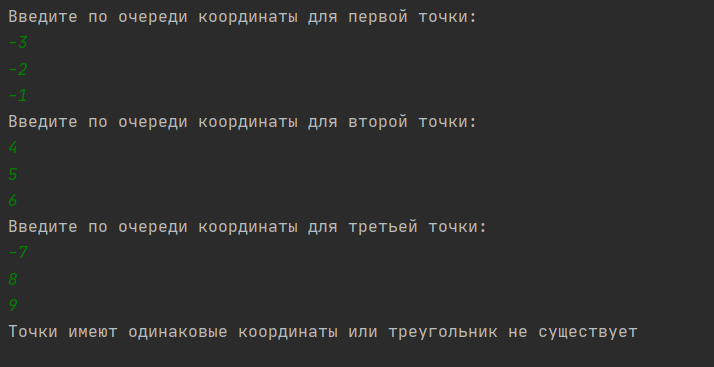
4. Скомпилируйте оба исходных файла вместе: javac Point3d.java Lab1.java и затем запустите программу Lab1, тестируя ее с несколькими образцами треугольников.

**Результат выполнения**

public class Point2d {  
  
 private double xCoord; // Координата X  
 private double yCoord; // Координата Y  
  
 public Point2d(double x, double y) // Конструктор инициализации  
 {  
 xCoord = x;  
 yCoord = y;  
 }  
  
 public Point2d()  
 {  
 this(0,0);  
 }  
  
 public double getX() // Возвращение координаты X  
 {  
 return xCoord;  
 }  
  
 public double getY() // Возвращение координаты Y  
 {  
 return yCoord;  
 }  
  
 public void setX(double val) // Установка значения координаты X  
 {  
 xCoord = val;  
 }  
  
 public void setY(double val) // Установка значения координаты Y  
 {  
 yCoord = val;  
 }  
  
 public static boolean compare(Point2d a, Point2d b) // Сравнение значений точек  
 {  
 if ((a.getX() == b.getX()) && (a.getY() == b.getY()))  
 return true;  
 else  
 return false;  
 }  
 }

public class Point3d extends Point2d {  
  
 static double sqr(double val){  
 return val\*val;  
 }  
 private double zCoord;  
 public Point3d(double x, double y, double z){  
 zCoord = z;  
 }  
  
 public double getZ(){  
 return zCoord;  
 }  
  
 public void setZ(double val){  
 zCoord = val;  
 }  
  
 public Point3d(){  
 this(0.0,0.0,0.0);  
 }  
  
 public static boolean compare(Point3d first, Point3d second){  
 return first.getX()== second.getX() && first.getZ() == second.getY() && first.getZ() == second.getZ();  
 }  
  
 public static double distanceTo(Point3d first, Point3d second){  
 double Round = Math.*sqrt*(*sqr*(second.getX() - first.getX())+ *sqr*(second.getY()- first.getY())+*sqr*(second.getZ()- first.getZ()));  
 Round\*=100;  
 return Round = ((int) Round)/100;  
  
 }  
  
}

import java.util.Scanner;  
  
public class Lab1 extends Point3d {  
 public static void main(String[] args){  
 Scanner in = new Scanner(System.*in*);  
 System.*out*.print("Введите по очереди координаты для первой точки: " + "\n");  
 double a = in.nextInt();  
 double b = in.nextInt();  
 double c = in.nextInt();  
 System.*out*.print("Введите по очереди координаты для второй точки: " + "\n");  
 double a1 = in.nextInt();  
 double b1 = in.nextInt();  
 double c1 = in.nextInt();  
 System.*out*.print("Введите по очереди координаты для третьей точки: " + "\n");  
 double a2 = in.nextInt();  
 double b2 = in.nextInt();  
 double c2 = in.nextInt();  
 Point3d Point = new Point3d(a,b,c);  
 Point3d Point1 = new Point3d(a1,b1,c1);  
 Point3d Point2 = new Point3d(a2,b2,c2);  
  
 Point.setX(a);  
 Point1.setX(a1);  
 Point2.setZ(a2);  
  
 Point.setY(b);  
 Point1.setY(b1);  
 Point2.setY(b2);  
  
 Point.setZ(c);  
 Point1.setZ(c1);  
 Point2.setZ(c2);  
  
 in.close(); //Останавливаем работу Scanner  
  
 if(*compare*(Point, Point1) || *compare*(Point1, Point2) || *compare*(Point, Point2) || *existance*(Point, Point1, Point2) ) //Сравниваем координаты точек  
 System.*out*.println("Точки имеют одинаковые координаты или треугольник не существует");  
 else {  
 System.*out*.println("Площадь треугольника равна = 1" + Lab1.*Area*(Point, Point1, Point2));  
 }  
 }  
 public static double Area(Point3d first, Point3d second, Point3d third){  
 double lenght1 = *distanceTo*(first, second), lenght2 = *distanceTo*(first, third), lenght3 = *distanceTo*(second, third);  
 double per = (lenght1 + lenght2 + lenght3) / 2;  
 return Math.*sqrt*(per \* (per - lenght1) \* (per - lenght2) \* (per - lenght3));  
 }  
  
 public static boolean existance (Point3d first, Point3d second, Point3d third){  
 if (*distanceTo*(first, second)>*distanceTo*(first, third)+*distanceTo*(second, third)) return false;  
 if (*distanceTo*(third, second)>*distanceTo*(first, third)+*distanceTo*(first, second)) return false;  
 if (*distanceTo*(first, third)>*distanceTo*(first, second)+*distanceTo*(second, third)) return false;  
 else return true;  
 }  
}

**Результат выполнения программы  
**

Ссылка на гит-репозиторий:

https://github.com/NillBard/Java/tree/master/%D0%9E%D0%9E%D0%BF