МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

**Кафедра ''САПР''**

Отчет о выполненной лабораторной работе № 3

по дисциплине: “Объектно-ориентированное программирование”

Выполнили ст.гр. 20ВА1:

Пачков А.В.

Комаров М.В.

Принял:

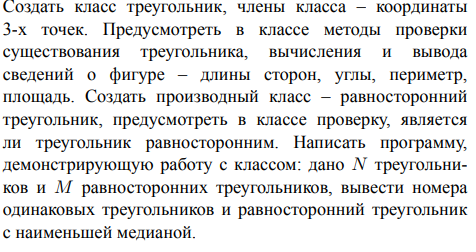
Подмарькова Е.М.

**Цель работы:**

Изучить возможности наследования классов на языке Java.

**Ход работы:**

Задание:



Код программы:

Класс **Triangle**

public class Triangle {  
 // Координаты вершин треугольника  
 protected double x1, y1;  
 protected double x2, y2;  
 protected double x3, y3;  
  
 // Конструктор  
 public Triangle(double x1, double y1, double x2, double y2, double x3, double y3) {  
 this.x1 = x1;  
 this.y1 = y1;  
 this.x2 = x2;  
 this.y2 = y2;  
 this.x3 = x3;  
 this.y3 = y3;  
 }  
  
 // Метод проверки существования треугольника  
 public boolean isValid() {  
 double a = distance(x1, y1, x2, y2);  
 double b = distance(x2, y2, x3, y3);  
 double c = distance(x3, y3, x1, y1);  
 return a + b > c && b + c > a && c + a > b;  
 }  
  
 // Метод вычисления расстояния между двумя точками  
 protected double distance(double x1, double y1, double x2, double y2) {  
 return Math.*sqrt*(Math.*pow*(x2 - x1, 2) + Math.*pow*(y2 - y1, 2));  
 }  
  
 // Метод вычисления периметра  
 public double perimeter() {  
 double a = distance(x1, y1, x2, y2);  
 double b = distance(x2, y2, x3, y3);  
 double c = distance(x3, y3, x1, y1);  
 return a + b + c;  
 }  
  
 // Метод вычисления площади  
 public double area() {  
 double a = distance(x1, y1, x2, y2);  
 double b = distance(x2, y2, x3, y3);  
 double c = distance(x3, y3, x1, y1);  
 double s = perimeter() / 2;  
 return Math.*sqrt*(s \* (s - a) \* (s - b) \* (s - c));  
 }  
  
 // Метод вычисления углов  
 public double[] angles() {  
 double a = distance(x1, y1, x2, y2);  
 double b = distance(x2, y2, x3, y3);  
 double c = distance(x3, y3, x1, y1);  
 double angleA = Math.*toDegrees*(Math.*acos*((b \* b + c \* c - a \* a) / (2 \* b \* c)));  
 double angleB = Math.*toDegrees*(Math.*acos*((a \* a + c \* c - b \* b) / (2 \* a \* c)));  
 double angleC = 180 - angleA - angleB;  
 return new double[]{angleA, angleB, angleC};  
 }  
  
 // Метод вывода информации о треугольнике  
 public void printInfo() {  
 System.*out*.println("Треугольник:");  
 System.*out*.println("Стороны: " + distance(x1, y1, x2, y2) + ", " + distance(x2, y2, x3, y3) + ", " + distance(x3, y3, x1, y1));  
 double[] angles = angles();  
 System.*out*.println("Углы: " + angles[0] + ", " + angles[1] + ", " + angles[2]);  
 System.*out*.println("Периметр: " + perimeter());  
 System.*out*.println("Площадь: " + area());  
 }  
}

Класс **EquilateralTriangle**

public class EquilateralTriangle extends Triangle {  
  
 // Конструктор  
 public EquilateralTriangle(double x1, double y1, double x2, double y2, double x3, double y3) {  
 super(x1, y1, x2, y2, x3, y3);  
 }  
  
 // Метод проверки равностороннего треугольника  
 public boolean isEquilateral() {  
 double a = distance(super.x1, super.y1, super.x2, super.y2);  
 double b = distance(super.x2, super.y2, super.x3, super.y3);  
 double c = distance(super.x3, super.y3, super.x1, super.y1);  
 return a == b && b == c;  
 }  
  
 @Override  
 public void printInfo() {  
 super.printInfo();  
 if (isEquilateral()) {  
 System.*out*.println("Этот треугольник равносторонний.");  
 } else {  
 System.*out*.println("Этот треугольник не является равносторонним.");  
 }  
 }  
}

Класс **Main**

import java.util.ArrayList;  
import java.util.List;  
  
public class Main {  
 public static void main(String[] args) {  
 int N = 3; // количество обычных треугольников  
 int M = 3; // количество равносторонних треугольников  
  
 List<Triangle> triangles = new ArrayList<>();  
 List<EquilateralTriangle> equilateralTriangles = new ArrayList<>();  
  
 // Создание N обычных треугольников с целочисленными сторонами  
 triangles.add(new Triangle(0, 0, 3, 0, 0, 4)); // Стороны: 3, 4, 5  
 triangles.add(new Triangle(0, 0, 4, 0, 0, 3)); // Стороны: 4, 3, 5  
 triangles.add(new Triangle(0, 0, 5, 0, 0, 12)); // Стороны: 5, 12, 13  
  
 // Создание M равносторонних треугольников с целочисленными сторонами  
 equilateralTriangles.add(new EquilateralTriangle(0, 0, 3, 0, 1.5, Math.*sqrt*(6.75))); // Сторона: 3  
 equilateralTriangles.add(new EquilateralTriangle(0, 0, 6, 0, 3, Math.*sqrt*(27))); // Сторона: 6  
 equilateralTriangles.add(new EquilateralTriangle(0, 0, 9, 0, 4.5, Math.*sqrt*(60.75))); // Сторона: 9  
  
 // Вывод информации о треугольниках  
 System.*out*.println("Обычные треугольники:");  
 for (int i = 0; i < triangles.size(); i++) {  
 System.*out*.println("Треугольник " + (i + 1) + ":");  
 if (triangles.get(i).isValid()) {  
 triangles.get(i).printInfo();  
 } else {  
 System.*out*.println("Этот треугольник не существует.");  
 }  
 System.*out*.println();  
 }  
  
 System.*out*.println("Равносторонние треугольники:");  
 for (int i = 0; i < equilateralTriangles.size(); i++) {  
 System.*out*.println("Равносторонний треугольник " + (i + 1) + ":");  
 if (equilateralTriangles.get(i).isValid()) {  
 equilateralTriangles.get(i).printInfo();  
 } else {  
 System.*out*.println("Этот треугольник не существует.");  
 }  
 System.*out*.println();  
 }  
  
 // Поиск и вывод номеров одинаковых треугольников  
 System.*out*.println("Одинаковые треугольники:");  
 for (int i = 0; i < triangles.size(); i++) {  
 for (int j = i + 1; j < triangles.size(); j++) {  
 if (*areTrianglesEqual*(triangles.get(i), triangles.get(j))) {  
 System.*out*.println("Треугольники " + (i + 1) + " и " + (j + 1) + " одинаковые.");  
 }  
 }  
 }  
  
 // Поиск равностороннего треугольника с наименьшей медианой  
 EquilateralTriangle minMedianTriangle = null;  
 double minMedian = Double.*MAX\_VALUE*;  
 for (EquilateralTriangle triangle : equilateralTriangles) {  
 double median = *calculateMedian*(triangle);  
 if (median < minMedian) {  
 minMedian = median;  
 minMedianTriangle = triangle;  
 }  
 }  
  
 System.*out*.println("Равносторонний треугольник с наименьшей медианой:");  
 if (minMedianTriangle != null) {  
 minMedianTriangle.printInfo();  
 System.*out*.println("Наименьшая медиана: " + minMedian);  
 } else {  
 System.*out*.println("Нет действительных равносторонних треугольников.");  
 }  
 }  
  
 // Метод для проверки равенства двух треугольников по длине сторон  
 private static boolean areTrianglesEqual(Triangle t1, Triangle t2) {  
 double[] sides1 = *getSortedSides*(t1);  
 double[] sides2 = *getSortedSides*(t2);  
 for (int i = 0; i < 3; i++) {  
 // Фикс сравнения числа с плавающей точкой  
 if (Math.*abs*(sides1[i] - sides2[i]) > 1e-6) {  
 return false;  
 }  
 }  
 return true;  
 }  
  
 // Метод для получения отсортированных длин сторон треугольника  
 private static double[] getSortedSides(Triangle t) {  
 double a = t.distance(t.x1, t.y1, t.x2, t.y2);  
 double b = t.distance(t.x2, t.y2, t.x3, t.y3);  
 double c = t.distance(t.x3, t.y3, t.x1, t.y1);  
 double[] sides = {a, b, c};  
 java.util.Arrays.*sort*(sides);  
 return sides;  
 }  
  
 // Метод для вычисления медианы равностороннего треугольника  
 private static double calculateMedian(EquilateralTriangle t) {  
 double side = t.distance(t.x1, t.y1, t.x2, t.y2);  
 return Math.*sqrt*(3) / 2 \* side;  
 }  
}

**Результат работы:**

Обычные треугольники:

Треугольник 1:

Треугольник:

Стороны: 3.0, 5.0, 4.0

Углы: 36.86989764584401, 90.0, 53.13010235415598

Периметр: 12.0

Площадь: 6.0

Треугольник 2:

Треугольник:

Стороны: 4.0, 5.0, 3.0

Углы: 53.13010235415599, 90.0, 36.86989764584402

Периметр: 12.0

Площадь: 6.0

Треугольник 3:

Треугольник:

Стороны: 5.0, 13.0, 12.0

Углы: 22.61986494804042, 90.0, 67.38013505195957

Периметр: 30.0

Площадь: 30.0

Равносторонние треугольники:

Равносторонний треугольник 1:

Треугольник:

Стороны: 3.0, 3.0, 3.0

Углы: 60.00000000000001, 60.00000000000001, 59.99999999999999

Периметр: 9.0

Площадь: 3.897114317029974

Этот треугольник равносторонний.

Равносторонний треугольник 2:

Треугольник:

Стороны: 6.0, 6.0, 6.0

Углы: 60.00000000000001, 60.00000000000001, 59.99999999999999

Периметр: 18.0

Площадь: 15.588457268119896

Этот треугольник равносторонний.

Равносторонний треугольник 3:

Треугольник:

Стороны: 9.0, 9.0, 9.0

Углы: 60.00000000000001, 60.00000000000001, 59.99999999999999

Периметр: 27.0

Площадь: 35.074028853269766

Этот треугольник равносторонний.

Одинаковые треугольники:

Треугольники 1 и 2 одинаковые.

Равносторонний треугольник с наименьшей медианой:

Треугольник:

Стороны: 3.0, 3.0, 3.0

Углы: 60.00000000000001, 60.00000000000001, 59.99999999999999

Периметр: 9.0

Площадь: 3.897114317029974

Этот треугольник равносторонний.

Наименьшая медиана: 2.598076211353316

**Вывод:**

Мы изучили возможности наследования классов на языке Java.