import java.util.Arrays;  
import java.util.Scanner;  
  
public class TriangleArea {  
 private boolean legality;  
  
 private void judgeTriangleCalculateType(int type) {  
 Scanner in = new Scanner(System.*in*);  
 double result = 0;  
 switch (type) {  
 case 1 -> {  
 System.*out*.println("请分别输入 角角边 的值 每行一个 下一行输入是否是弧度");  
 result = this.AAS(in.nextDouble(), in.nextDouble(), in.nextDouble(), in.nextBoolean());  
 }  
 case 2 -> {  
 System.*out*.println("请分别输入 角边角 的值 每行一个 下一行输入是否是弧度");  
 result = this.ASA(in.nextDouble(), in.nextDouble(), in.nextDouble(), in.nextBoolean());  
 }  
 case 3 -> {  
 System.*out*.println("请分别输入 边角边 的值 每行一个 下一行输入是否是弧度");  
 result = this.SAS(in.nextDouble(), in.nextDouble(), in.nextDouble(), in.nextBoolean());  
 }  
 case 4 -> {  
 System.*out*.println("请分别输入 底高 的值 每行一个");  
 result = this.LH(in.nextDouble(), in.nextDouble());  
 }  
 case 5 -> {  
 System.*out*.println("请逐行输入 三个边 的值");  
 result = this.HeronFormula(in.nextDouble(), in.nextDouble(), in.nextDouble());  
 }  
 case 6 -> {  
 System.*out*.println("请输入第一个点的 x y值");  
 double[] point\_2a = new double[2];  
 for (double coordinate : point\_2a) {  
 coordinate = in.nextDouble();  
 }  
 System.*out*.println("请输入第二个点的 x y值");  
 double[] point\_2b = new double[2];  
 for (double coordinate : point\_2b) {  
 coordinate = in.nextDouble();  
 }  
 System.*out*.println("请输入第三个点的 x y值");  
 double[] point\_2c = new double[2];  
 for (double coordinate : point\_2c) {  
 coordinate = in.nextDouble();  
 }  
 result = this.TwoDimensionCoordinate(point\_2a, point\_2b, point\_2c);  
 }  
 case 7 -> {  
 System.*out*.println("请输入第一个点的 x y z 值");  
 double[] point\_3a = new double[3];  
 for (double coordinate : point\_3a) {  
 coordinate = in.nextDouble();  
 }  
 System.*out*.println("请输入第二个点的 x y z 值");  
 double[] point\_3b = new double[3];  
 for (double coordinate : point\_3b) {  
 coordinate = in.nextDouble();  
 }  
 System.*out*.println("请输入第三个点的 x y z 值");  
 double[] point\_3c = new double[3];  
 for (double coordinate : point\_3c) {  
 coordinate = in.nextDouble();  
 }  
 result = this.ThreeDimensionCoordinate(point\_3a, point\_3b, point\_3c);  
 }  
 }  
  
 System.*out*.println("三角形的面积为: " + result);  
 }  
  
 */\*\*  
 \* 底x高 类型  
 \*  
 \** ***@param*** *length 底边长  
 \** ***@param*** *height 高  
 \** ***@return*** *面积  
 \*/* private double LH(double length, double height) {  
 if (length <= 0 || height <= 0) {  
 System.*out*.println("三角形不合法");  
 System.*exit*(0);  
 }  
 return 0.5 \* length \* height;  
 }  
  
 */\*\*  
 \* 边角边  
 \*  
 \** ***@param*** *side\_a 边A  
 \** ***@param*** *angle 夹角  
 \** ***@param*** *side\_b 边B  
 \** ***@param*** *radian 是否为弧度  
 \** ***@return*** *面积  
 \*/* private double SAS(double side\_a, double angle, double side\_b, boolean radian) {  
 if (!radian) {  
 angle = Math.*toRadians*(angle);  
 }  
 return 0.5 \* side\_a \* side\_b \* Math.*sin*(angle);  
 }  
  
 */\*\*  
 \* 角角边  
 \*  
 \** ***@param*** *angle\_a 角A  
 \** ***@param*** *angle\_b 角B  
 \** ***@param*** *side 邻边  
 \** ***@param*** *radian 是否为弧度  
 \** ***@return*** *面积  
 \*/* private double AAS(double angle\_a, double angle\_b, double side, boolean radian) {  
 validateTriangle(new double[]{angle\_a, angle\_b}, new double[]{side});  
 if (!radian) {  
 angle\_a = Math.*toRadians*(angle\_a);  
 angle\_b = Math.*toRadians*(angle\_b);  
 }  
 return (side \* side \* Math.*sin*(angle\_a) \* Math.*sin*(angle\_a + angle\_b)) / (2 \* Math.*sin*(angle\_b));  
 }  
  
 */\*\*  
 \* 角边角  
 \*  
 \** ***@param*** *angle\_a 角A  
 \** ***@param*** *side 邻边  
 \** ***@param*** *angle\_b 角B  
 \** ***@param*** *radian 是否为弧度  
 \** ***@return*** *面积  
 \*/* private double ASA(double angle\_a, double side, double angle\_b, boolean radian) {  
 validateTriangle(new double[]{angle\_a, angle\_b}, new double[]{side});  
 if (!radian) {  
 angle\_a = Math.*toRadians*(angle\_a);  
 angle\_b = Math.*toRadians*(angle\_b);  
 }  
 return (side \* side \* Math.*sin*(angle\_b) \* Math.*sin*(angle\_a)) / (2.0 \* Math.*sin*(angle\_a + angle\_b));  
 }  
  
 */\*\*  
 \* 三边类型 即 海伦公式  
 \*  
 \** ***@param*** *side\_a 边A  
 \** ***@param*** *side\_b 边B  
 \** ***@param*** *side\_c 边C  
 \** ***@return*** *面积  
 \*/* private double HeronFormula(double side\_a, double side\_b, double side\_c) {  
 validateTriangle(new double[]{1}, new double[]{side\_a, side\_b, side\_c});  
 legality = side\_a + side\_b > side\_c;  
  
 double s = (side\_a + side\_b + side\_c) / 2;  
 return Math.*sqrt*(s \* (s - side\_a) \* (s - side\_b) \* (s - side\_c));  
 // 一般 在有小角度时 Heron 公式的值不稳定 可以用替代公式  
 // double cosC = (Math.pow(side\_b,2) + Math.pow(side\_c, 2) - Math.pow(side\_a, 2) ) / (2 \* side\_b \* side\_c);  
 }  
  
 */\*\*  
 \* 三维坐标计算类型  
 \*  
 \** ***@param*** *point\_a 点A  
 \** ***@param*** *point\_b 点B  
 \** ***@param*** *point\_c 点C  
 \** ***@return*** *面积  
 \*/* private double ThreeDimensionCoordinate(double[] point\_a, double[] point\_b, double[] point\_c) {  
 this.validateDimensionTriangle(3, point\_a, point\_b, point\_c);  
 if (!this.legality) {  
 System.*out*.println("三角形不合法");  
 System.*exit*(0);  
 }  
 double x1 = point\_a[0];  
 double y1 = point\_a[1];  
 double z1 = point\_a[2];  
  
 double x2 = point\_b[0];  
 double y2 = point\_b[1];  
 double z2 = point\_b[2];  
  
 double x3 = point\_c[0];  
 double y3 = point\_c[1];  
 double z3 = point\_c[2];  
  
 return HeronFormula(Math.*sqrt*(Math.*pow*(x2 - x1, 2) + Math.*pow*(y2 - y1, 2) + Math.*pow*(z2 - z1, 2)), Math.*sqrt*(Math.*pow*(x3 - x1, 2) + Math.*pow*(y3 - y1, 2) + Math.*pow*(z3 - z1, 2)), Math.*sqrt*(Math.*pow*(x3 - x2, 2) + Math.*pow*(y3 - y2, 2) + Math.*pow*(z3 - z2, 2)));  
 }  
  
 */\*\*  
 \* 二维坐标类型  
 \*  
 \** ***@param*** *point\_a 点A  
 \** ***@param*** *point\_b 点B  
 \** ***@param*** *point\_c 点C  
 \** ***@return*** *面积  
 \*/* private double TwoDimensionCoordinate(double[] point\_a, double[] point\_b, double[] point\_c) {  
 this.validateDimensionTriangle(2, point\_a, point\_b, point\_c);  
 if (!this.legality) {  
 System.*out*.println("三角形不合法");  
 System.*exit*(0);  
 }  
  
 double x1 = point\_a[0];  
 double y1 = point\_a[1];  
 double x2 = point\_b[0];  
 double y2 = point\_b[1];  
 double x3 = point\_c[0];  
 double y3 = point\_c[1];  
  
 return x1 \* y2 - x1 \* y3 + x2 \* y3 - x2 \* y1 + x3 \* y1 - x2 \* y2;  
 }  
  
 */\*\*  
 \* 校验三角形合法性  
 \*  
 \** ***@param*** *angles 角集合  
 \** ***@param*** *sides 边集合  
 \*/* private void validateTriangle(double[] angles, double[] sides) {  
 for (double angle : angles) {  
 if (angle <= 0) {  
 this.legality = false;  
 System.*exit*(0);  
 }  
 }  
 for (double side : sides) {  
 if (side <= 0) {  
 this.legality = false;  
 System.*exit*(0);  
 }  
 }  
 this.legality = true;  
 }  
  
 */\*\*  
 \* 校验维度型三角形合法性  
 \*  
 \** ***@param*** *dimension 维度 2 / 3  
 \** ***@param*** *point\_a 点A  
 \** ***@param*** *point\_b 点B  
 \** ***@param*** *point\_c 点C  
 \*/* private void validateDimensionTriangle(int dimension, double[] point\_a, double[] point\_b, double[] point\_c) {  
 // *todo : 判断 三点 在一条直线上* if (point\_a.length == dimension || point\_b.length == dimension || point\_c.length == dimension) {  
 this.legality = true;  
 } else if (Arrays.*equals*(point\_a, point\_b) || Arrays.*equals*(point\_c, point\_b) || Arrays.*equals*(point\_c, point\_a)) {  
 this.legality = false;  
 System.*exit*(0);  
 }  
  
 }  
  
 public static void main(String[] args) {  
 TriangleArea triangleCalc = new TriangleArea();  
 Scanner in = new Scanner(System.*in*);  
 while (true) {  
 System.*out*.println("请选择: 已知的三角形数据类型: ");  
 System.*out*.println("\t0 - 结束输入 \n\t1 - 角角边 AAS \n\t2 - 角边角 ASA \n\t3 - 边角边 SAS \n\t4 - 底高 LH \n\t5 - 三边 SSS \n\t6 - 二维坐标 \n\t7 - 三维坐标");  
 int type = in.nextInt();  
 if (type == 0) {  
 System.*exit*(0);  
 }  
 triangleCalc.judgeTriangleCalculateType(type);  
 }  
 }  
  
}