Министерство образования и молодежной политики Свердловской области



ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

Отчёт по программе «**Практическое задание 7. Задание 3**»

Выполнил: Зиятдинова Алина Ленаровна

Группа: ПР-22

Преподаватель: Мирошниченко Г.В

2023

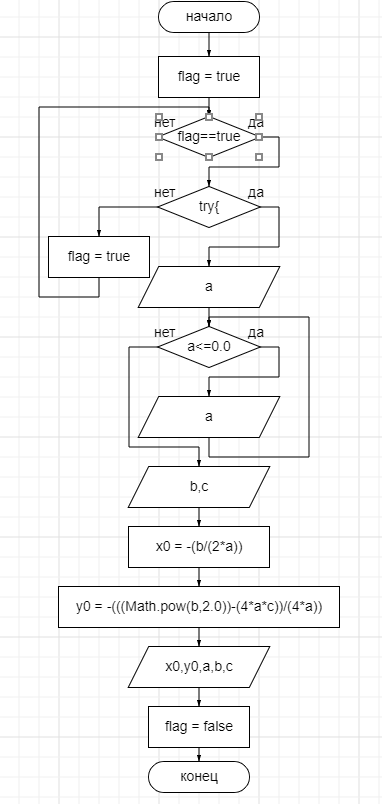
**Задание 3****(1)**

**Входные данные**

a,b,c – коэффициенты, вещественный тип

**Выходные данные**

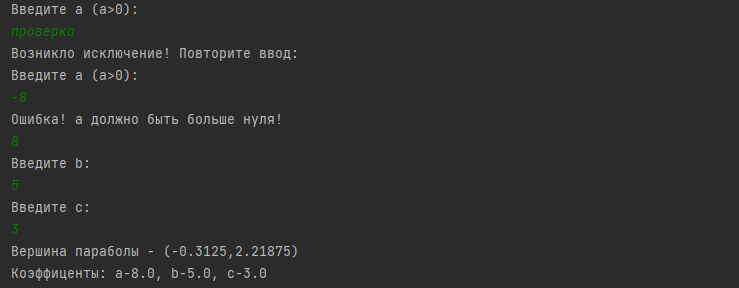
**x0 , y0 –** координаты вершины параболы, вещественный тип

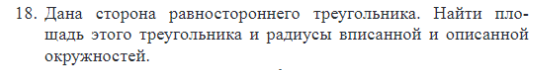
**Блок-схема**

**Листинг программы**

import kotlin.math.\*  
fun main()  
{  
 var flag = true  
 while (flag == true)  
 {  
 try  
 {  
 *println*("Введите а (a>0):")  
 var a = *readLine*()!!.*toDouble*()  
 while (a<=0.0)  
 {  
 *println*("Ошибка! а должно быть больше нуля!")  
 a = *readLine*()!!.*toDouble*()  
 }  
 *println*("Введите b:")  
 var b = *readLine*()!!.*toDouble*()  
 *println*("Введите c:")  
 var c = *readLine*()!!.*toDouble*()  
 val x0 = -(b/(2\*a))  
 val y0 = -(((Math.pow(b,2.0))-(4\*a\*c))/(4\*a))  
 *println*("Вершина параболы - ($x0,$y0)")  
 *println*("Коэффиценты: a-$a, b-$b, c-$c")  
 flag = false  
 }  
 catch (e:Exception)  
 {  
 *println*("Возникло исключение! Повторите ввод:")  
 flag = true  
 }  
 }  
}

**Тестовые ситуации**

 **Вывод**



**Входные данные**

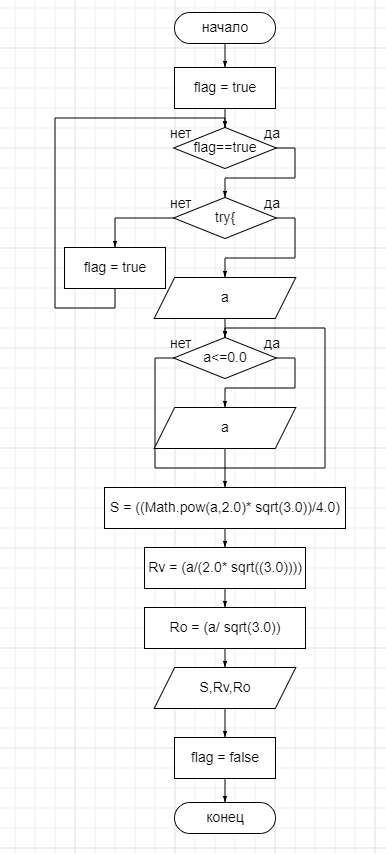
a – сторона равностороннего треугольника, вещественный тип

**Выходные данные**

**S –** площадь треугольника, вещ.тип

**Rv-** радиус вписанной окружности, вещ тип

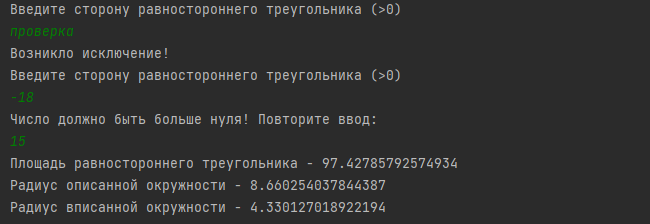
**Ro**-радиус описанной окружности, вещ.тип

**Блок-схема**

**Листинг программы**

import kotlin.math.\*  
fun main() {  
 var flag = true  
 while (flag == true) {  
 try {  
 *println*("Введите сторону равностороннего треугольника (>0)")  
 var a = *readLine*()!!.*toDouble*()  
 while (a<=0.0)  
 {  
 *println*("Число должно быть больше нуля! Повторите ввод:")  
 a = *readLine*()!!.*toDouble*()  
 }  
 var S = ((Math.pow(a,2.0)\* *sqrt*(3.0))/4.0)  
 var Rv = (a/(2.0\* *sqrt*((3.0))))  
 var Ro = (a/ *sqrt*(3.0))  
 *println*("Площадь равностороннего треугольника - $S")  
 *println*("Радиус описанной окружности - $Ro")  
 *println*("Радиус вписанной окружности - $Rv")  
 flag=false  
 } catch (e: Exception) {  
 *println*("Возникло исключение!")  
 flag = true  
 }  
 }  
}

**Тестовые ситуации**



**Задание 3(2)**

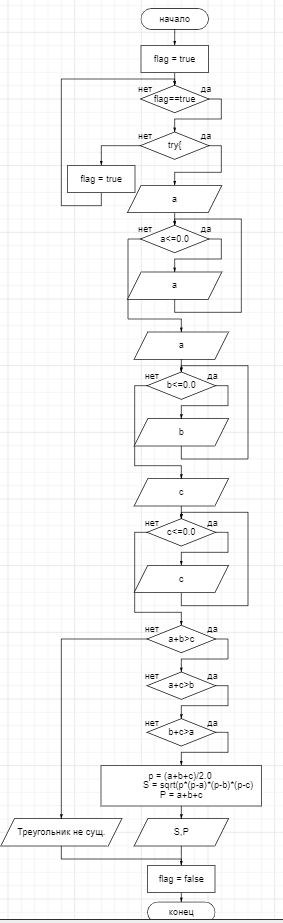
**Входные данные**

**a,b,c –** стороны треугольника, вещ. типы. (>0)

**Выходные данные**

**S-**площадь треугольника, вещ.тип

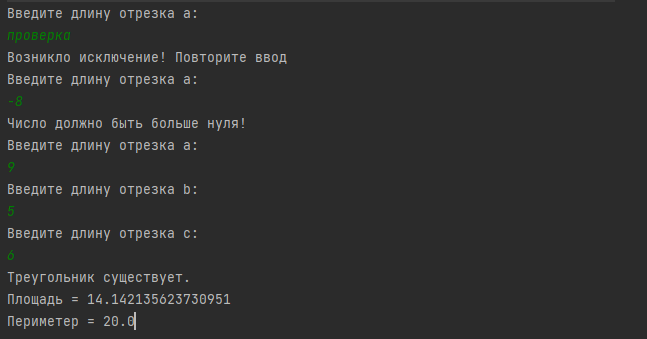
**P-**периметр треугольника, вещ.тип

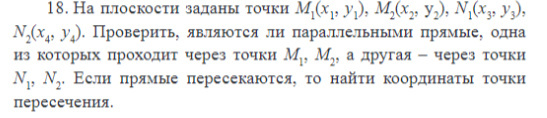
**Блок схема**

**Листинг программы**

import kotlin.math.\*  
fun main()  
{  
 var flag = true  
 var est = ""  
 while (flag == true) {  
 try {  
 *println*("Введите длину отрезка a:")  
 var a = *readLine*()!!.*toDouble*()  
 while (a<=0.0)  
 {  
 *println*("Число должно быть больше нуля!")  
 *println*("Введите длину отрезка a:")  
 a = *readLine*()!!.*toDouble*()  
 }  
 *println*("Введите длину отрезка b:")  
 var b = *readLine*()!!.*toDouble*()  
 while (b<=0.0)  
 {  
 *println*("Число должно быть больше нуля!")  
 *println*("Введите длину отрезка a:")  
 b = *readLine*()!!.*toDouble*()  
 }  
 *println*("Введите длину отрезка c:")  
 var c = *readLine*()!!.*toDouble*()  
 while (c<=0.0)  
 {  
 *println*("Число должно быть больше нуля!")  
 *println*("Введите длину отрезка a:")  
 c = *readLine*()!!.*toDouble*()  
 }  
 if (a+b>c)  
 {  
 if (a+c>b)  
 {  
 if (b+c>a)  
 {  
 var p = (a+b+c)/2.0  
 var S = *sqrt*(p\*(p-a)\*(p-b)\*(p-c))  
 var P = a+b+c  
  
 *println* ("Треугольник существует. \nПлощадь = $S\nПериметер = $P")  
 }  
 }  
 }  
 else  
 *println* ("Такой треугольник не существует.")  
 flag = false  
 }  
 catch (e:Exception)  
 {  
 *println*("Возникло исключение! Повторите ввод")  
 flag = true  
 }  
 }  
}

**Тестовые ситуации**





**Входные данные**

**x1,y1 –** координаты M1 точки, вещ.типы

**x2,y2 –** координаты M2 точки, вещ.типы

**x3,y3 –** координаты M3 точки, вещ.типы

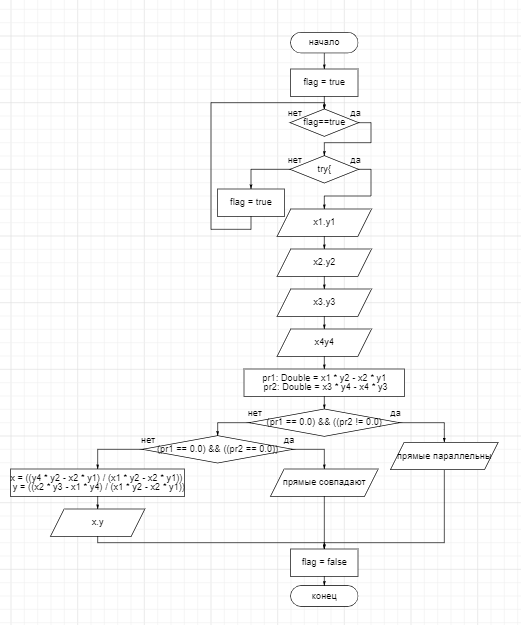
**x4,y4 –** координаты M4 точки, вещ.типы

**Выходные данные**

Сообщение о параллельности или совпадении прямых

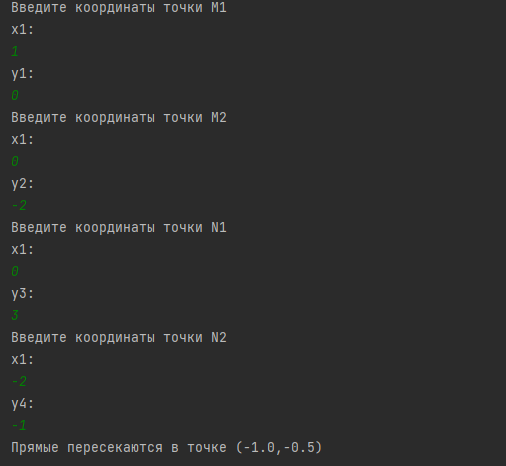
ИЛИ

x,y – точка пересечения прямых, вещ.типы

**Блок схема**

**Листинг программы**

import kotlin.math.\*  
fun main() {  
 var flag = true  
 while (flag == true) {  
 try {  
 *println*("Введите координаты точки М1")  
 *println*("x1:")  
 var x1 = *readLine*()!!.*toDouble*()  
 *println*("y1:")  
 var y1 = *readLine*()!!.*toDouble*()  
 *println*("Введите координаты точки М2")  
 *println*("x1:")  
 var x2 = *readLine*()!!.*toDouble*()  
 *println*("y2:")  
 var y2 = *readLine*()!!.*toDouble*()  
 *println*("Введите координаты точки N1")  
 *println*("x1:")  
 var x3 = *readLine*()!!.*toDouble*()  
 *println*("y3:")  
 var y3 = *readLine*()!!.*toDouble*()  
 *println*("Введите координаты точки N2")  
 *println*("x1:")  
 var x4 = *readLine*()!!.*toDouble*()  
 *println*("y4:")  
 var y4 = *readLine*()!!.*toDouble*()  
 var pr1: Double = x1 \* y2 - x2 \* y1  
 var pr2: Double = x3 \* y4 - x4 \* y3  
 if ((pr1 == 0.0) && ((pr2 != 0.0))) {  
 *println*("Прямые паралелльны")  
 } else if ((pr1 == 0.0) && ((pr2 == 0.0))) {  
 *println*("Прямые совпадают")  
 } else {  
 var x = ((y4 \* y2 - x2 \* y1) / (x1 \* y2 - x2 \* y1))  
 var y = ((x2 \* y3 - x1 \* y4) / (x1 \* y2 - x2 \* y1))  
 *println*("Прямые пересекаются в точке ($x,$y)")  
 }  
 flag = false  
 } catch (e: Exception) {  
 *println*("Возникло исключение! Повторите ввод")  
 flag = true  
 }  
 }  
}

**Тестовые ситуации**

