Министерство образования и молодежной политики Свердловской области



ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

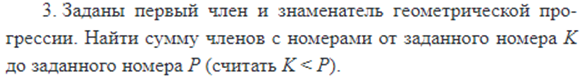
Отчёт по программе «**Практическое задание №7»**

Выполнил: Зиятдинова Алина Ленаровна

Группа: ПР-21

Преподаватель: Мирошниченко Г.В

2023

**Задание 1**

**Входные данные**

num1 = первый член геометрической прогрессии, вещественный тип.

znam = знаменатель геометрической прогрессии, вещественный тип.

a = номер числа – начала геометрической прогрессии, вещественный тип.

b = номер числа – конца геометрической прогрессии, вещественный тип.

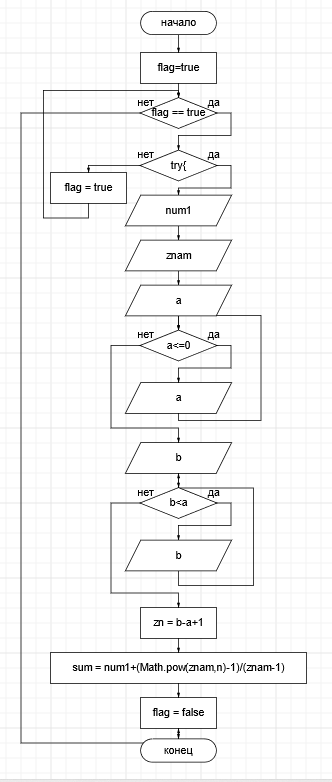
**Выходные данные**

sum = сумма чисел геометрической прогрессии в диапазоне от a до b, вещественный тип.

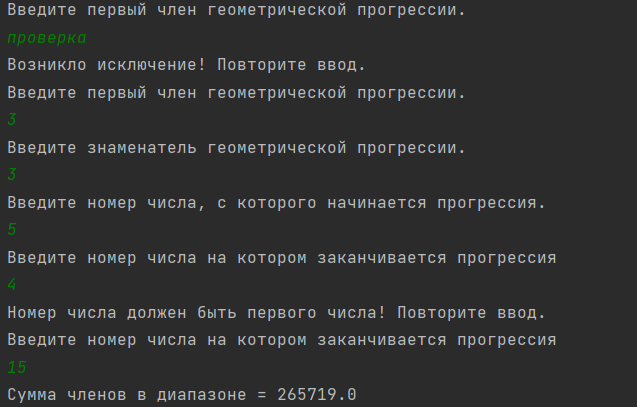
**Листинг программы (если есть)**

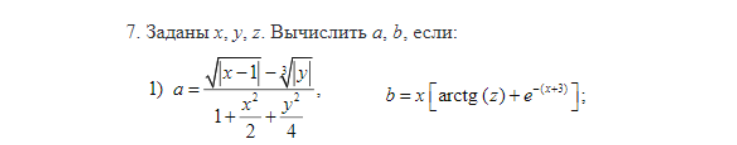
import kotlin.math.pow  
fun main() {  
 var flag = true  
 var num1: Double  
 var znam: Double  
 var a: Double  
 var b: Double  
 while (flag == true) {  
 try {  
 *println*("Введите первый член геометрической прогрессии.")  
 num1 = *readLine*()!!.*toDouble*()  
 *println*("Введите знаменатель геометрической прогрессии.")  
 znam = *readLine*()!!.*toDouble*()  
 *println*("Введите номер числа, с которого начинается прогрессия.")  
 a = *readLine*()!!.*toDouble*()  
 while (a<=0)  
 {  
 *println* ("Номер числа должен быть больше нуля! Повторите ввод.")  
 *println*("Введите номер числа, с которого начинается прогрессия.")  
 a = *readLine*()!!.*toDouble*()  
 }  
 *println*("Введите номер числа на котором заканчивается прогрессия")  
 b = *readLine*()!!.*toDouble*()  
 while (b<a)  
 {  
 *println* ("Номер числа должен быть первого числа! Повторите ввод.")  
 *println*("Введите номер числа на котором заканчивается прогрессия")  
 b = *readLine*()!!.*toDouble*()  
 }  
 var zn : Double = b-a+1  
 var sum:Double = num1 \* (Math.pow(znam,zn)-1)/(znam-1)  
 *println*("Сумма членов в диапазоне = $sum")  
 flag = false  
 }  
 catch (e:Exception) {  
 *println*("Возникло исключение! Повторите ввод.")  
 flag = true  
 }

**Блок схема**

****

**Тестовые ситуации**

****

****

**Входные данные**

x-число, вещественный тип

y-число, вещественный тип

z-число, вещественный тип

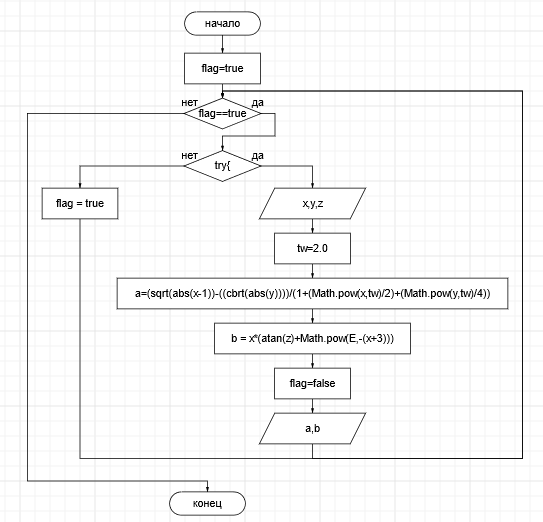
**Выходные данные**

a,b-результаты вычислений, вещественные типы

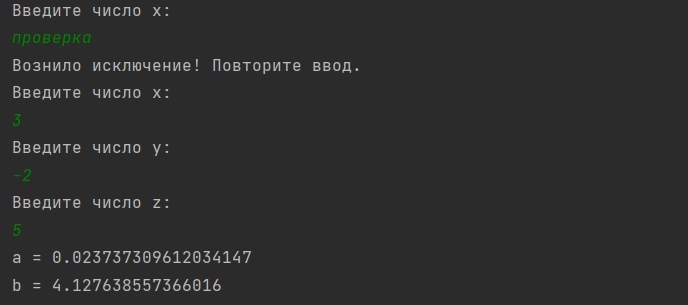
**Листинг-программы**

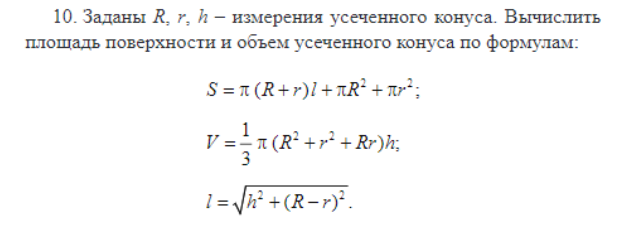
import kotlin.math.\*  
fun main()  
{  
 var flag = true  
 var x: Double  
 var y: Double  
 var z: Double  
 var a:Double  
 var b: Double  
 val E: Double = 2.7182818284  
 while (flag == true)  
 {  
 try  
 {  
 *println* ("Введите число x:")  
 x = *readln*()!!.*toDouble*()  
 *println* ("Введите число y:")  
 y = *readln*()!!.*toDouble*()  
 *println* ("Введите число z:")  
 z = *readln*()!!.*toDouble*()  
 var tw = 2.0  
 a = (*sqrt*(*abs*(x-1))-(*cbrt*(*abs*(y))))/(1+(Math.pow(x,tw)/2)+(Math.pow(y,tw)/4))  
 b = x\*(*atan*(z)+Math.pow(E,-(x+3)))  
 *println*("a = $a")  
 *println* ("b = $b")  
 flag = false  
 }  
 catch (e:Exception)  
 {  
 *println*("Вознило исключение! Повторите ввод.")  
 flag = true  
 }  
 }  
}

**Блок-схема**

****

**Тестовые ситуации**

****

****

**Входные данные**

R – радиус 1, вещественный тип

r - радиус 2, вещественный тип

h – высота, вещественный тип

**Выходные данные**

S-площадь поверхности конуса, вещ.тип

l-образующая конуса, вещ.тип

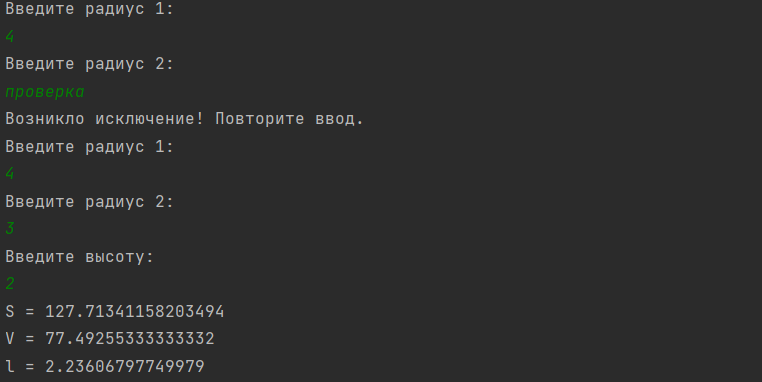
V-объем усеченного конуса, вещ.тип

**Листинг программы**

import kotlin.math.\*  
 fun main() {  
 var flag = true  
 var R: Double  
 var r:Double  
 var h:Double  
 var PI:Double = 3.14159  
 while (flag == true) {  
 try {  
 *println*("Введите радиус 1:")  
 R = *readln*()!!.*toDouble*()  
 *println*("Введите радиус 2:")  
 r = *readln*()!!.*toDouble*()  
 *println*("Введите высоту:")  
 h = *readln*()!!.*toDouble*()  
 var V = (1.0/3.0\*PI\*((Math.pow(R,2.0))+(Math.pow(r,2.0))+(R\*r))\*h)  
 var l = *sqrt*((Math.pow(h,2.0))+Math.pow(R-r,2.0))  
 var S = ((PI\*(R+r)\*l)+(PI\*Math.pow(R,2.0))+(PI\*Math.pow(r,2.0)))  
 *println* ("S = $S")  
 *println*("V = $V")  
 *println* ("l = $l")

flag = false  
 }  
 catch (e:Exception)  
 {  
 *println*("Возникло исключение! Повторите ввод.")  
 flag = true  
 }

**Тестовые ситуации**

****

**Задание 2**

****

**Входные данные**

a-действительное число 1, вещ.тип

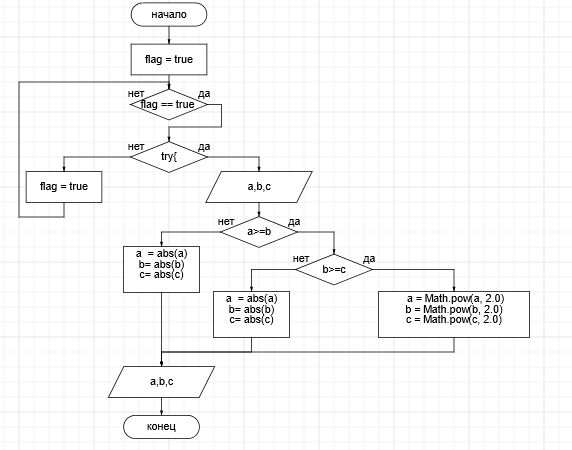
b-действительное число 2, вещ.тип

c-действительное число 3, вещ.тип

**Выходные данные**

a,b,c – действительные числа после изменения, в зависимости от условия задания, вещ.типы.

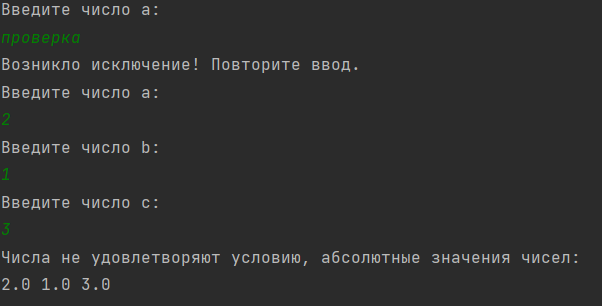
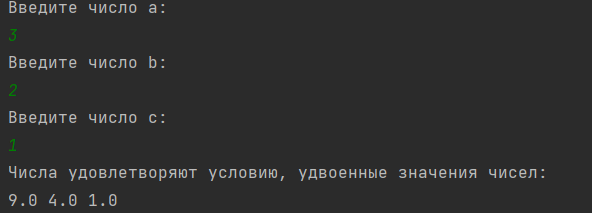
**Блок схема**

****

**Листинг программы**

import kotlin.math.\*  
fun main() {  
 var flag = true  
 while (flag == true) {  
 try {  
 *println*("Введите число a:")  
 var a = *readln*()!!.*toDouble*()  
 *println*("Введите число b:")  
 var b = *readln*()!!.*toDouble*()  
 *println*("Введите число c:")  
 var c = *readln*()!!.*toDouble*()  
 if (a>=b)  
 {  
 if (b>=c) {  
 a = Math.pow(a, 2.0)  
 b = Math.pow(b, 2.0)  
 c = Math.pow(c, 2.0)  
 *println*("Числа удовлетворяют условию, удвоенные значения чисел:")  
 *println* ("$a $b $c")  
 }  
 else  
 {  
 *println*("Числа не удовлетворяют условию, абсолютные значения чисел:")  
 a = *abs*(a)  
 b= *abs*(b)  
 c= *abs*(c)  
 *println* ("$a $b $c")  
 }  
 }  
 else  
 {  
 *println*("Числа не удовлетворяют условию, абсолютные значения чисел:")  
 a = *abs*(a)  
 b= *abs*(b)  
 c= *abs*(c)  
 *println* ("$a $b $c")  
 }  
 flag = false  
 }  
 catch (e:Exception)  
 {  
 *println*("Возникло исключение! Повторите ввод.")  
 flag = true  
 }  
 }  
}

**Тестовые ситуации**

****

****

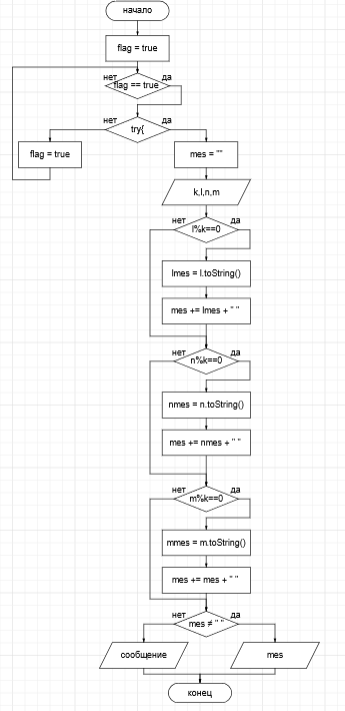
**Входные данные**

**k,l,n,m – целые числа, целочисленный тип**

**Выходные данные**

**mes – сообщение с числами, для которых k является делителем, строковый тип**

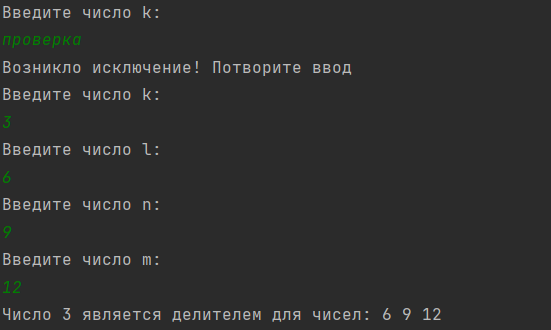
**Блок схема**

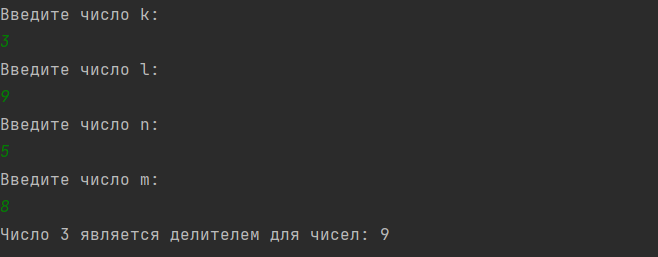
****

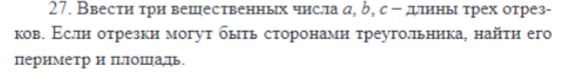
**Листинг программы**

import kotlin.math.\*  
fun main()  
{  
 var flag = true  
 var mes: String  
 mes = ""  
 while (flag == true) {  
 try {  
 *println*("Введите число k:")  
 var k = *readln*()!!.*toInt*()  
 *println*("Введите число l:")  
 var l = *readln*()!!.*toInt*()  
 *println*("Введите число n:")  
 var n = *readln*()!!.*toInt*()  
 *println*("Введите число m:")  
 var m = *readln*()!!.*toInt*()  
 if (l%k==0)  
 {  
 var lmes = l.toString()  
 mes += lmes + " "  
 }  
 if (n%k==0)  
 {  
 var nmes = n.toString()  
 mes += nmes + " "  
 }  
 if (m%k==0)  
 {  
 var mmes = m.toString()  
 mes += mmes +" "  
 }  
 if (mes != " ") {  
 *println*("Число $k является делителем для чисел: $mes")  
 }  
 else  
 {  
 *print*("Число $k не является делителем.")  
 }  
 flag = false  
 }  
 catch (e:Exception)  
 {  
 *println*("Возникло исключение! Потворите ввод")  
 flag = true  
 }  
 }  
}

**Тестовые ситуации**

****

****



**Входные данные**

a – длина отрезка 1, вещественный тип

b – длина отрезка 2, вещественный тип

c – длина отрезка 3, вещественный тип

**Выходные данные**

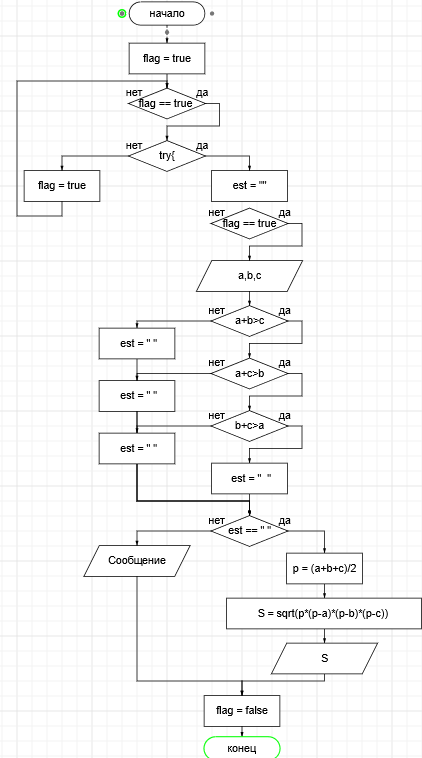
P – периметр треугольника

S – площадь треугольника

ИЛИ

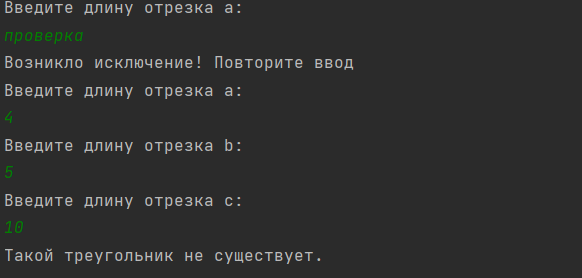
Сообщение о невозможности решения

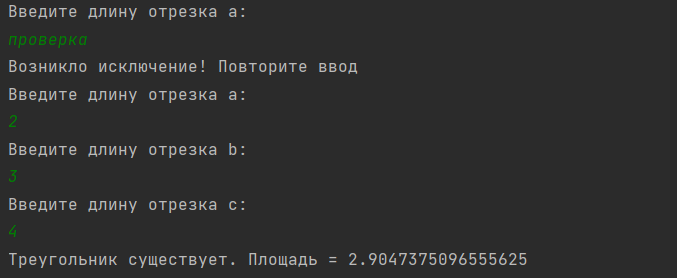
**Блок схема**

**  
Листинг программы**

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

**Тестовые ситуации**

****

****

****

**Входные данные**

k-число 1, вещ.тип

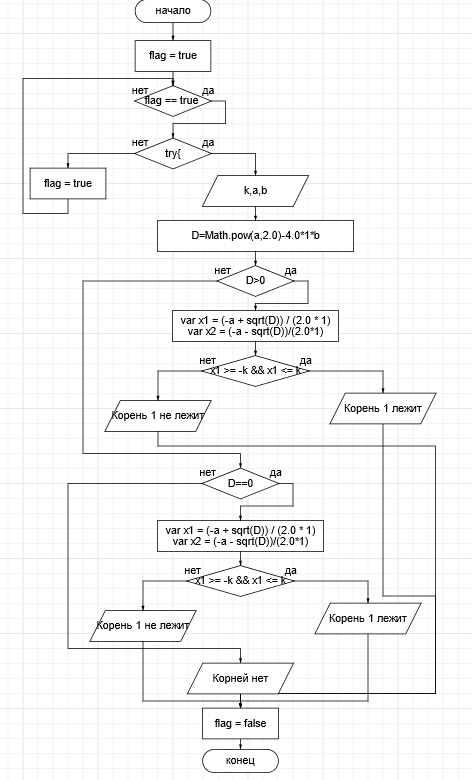
a-число 2, вещ.тип

b-число 3, вещ.тип

**Выходные данные**

Сообщение попадают корни или нет.

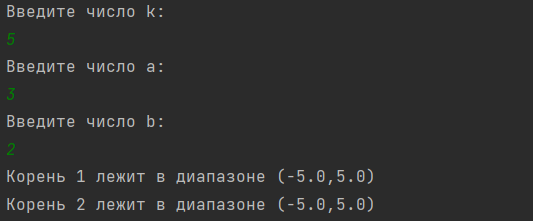
**Блок схема**

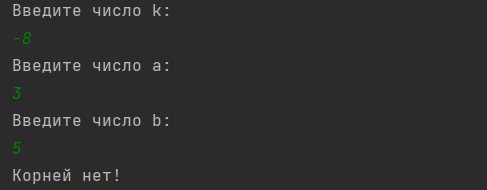
****

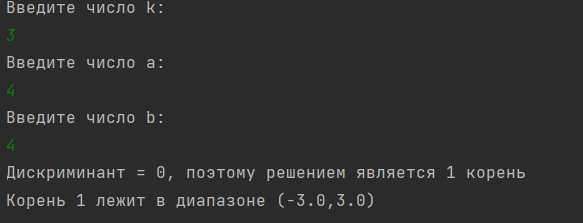
**Листинг программы**

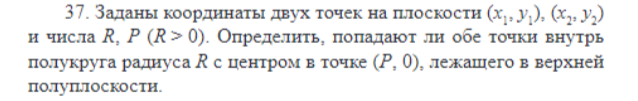
import kotlin.math.\*  
fun main() {  
 var flag = true  
 while (flag == true) {  
 try {  
 *println*("Введите число k:")  
 var k = *readln*()!!.*toDouble*()  
 *println*("Введите число a:")  
 var a = *readln*()!!.*toDouble*()  
 *println*("Введите число b:")  
 var b = *readln*()!!.*toDouble*()  
 var D = Math.pow(a,2.0)-4.0\*1\*b  
 if (D>0) {  
 var x1 = (-a + *sqrt*(D)) / (2.0 \* 1)  
 var x2 = (-a - *sqrt*(D))/(2.0\*1)  
 if (x1 >= -k && x1 <= k)  
 {  
 *println*("Корень 1 лежит в диапазоне (-$k,$k)")  
 }  
 else  
 *println*("Корень 1 не лежит в диапазоне (-$k,$k)")  
 if (x2 >= -k && x2 <= k)  
 {  
 *println*("Корень 2 лежит в диапазоне (-$k,$k)")  
 }  
 else  
 *println*("Корень 2 не лежит в диапазоне (-$k,$k)")  
 }  
 else  
 {  
 if (D==0.0)  
 {  
 *println*("Дискриминант = 0, поэтому решением является 1 корень")  
 var x1 = (-b/(2.0\*a))  
 if (x1 >= -k && x1 <= k)  
 {  
 *println*("Корень 1 лежит в диапазоне (-$k,$k)")  
 }  
 else  
 *println*("Корень 1 не лежит в диапазоне (-$k,$k)")  
 }  
 else {  
 *println*("Корней нет!")  
 }  
 }  
 flag = false  
 }  
 catch (e:Exception)  
 {  
 *println*("Возникло исключение! Повторите ввод")  
 flag = true  
 }  
 }  
}

**Тестовые ситуации**

****

****

****

****

**Входные данные**

R- радиус , вещ.тип

P – координата радиуса, вещ.тип

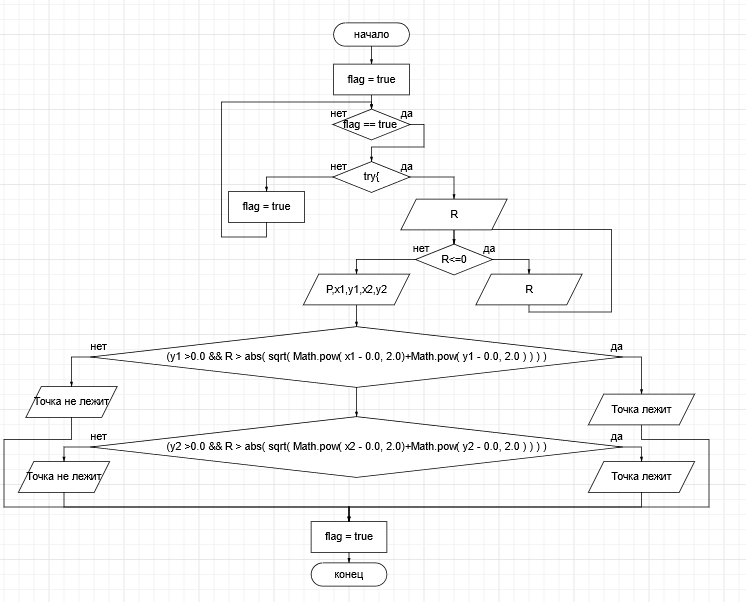
x1,y1 – координаты 1 точки, вещ.типы

x2,y2 – координаты 2 точки, вещ.типы

**Выходные данные**

Сообщения о том, лежит ли точка в полуплоскости

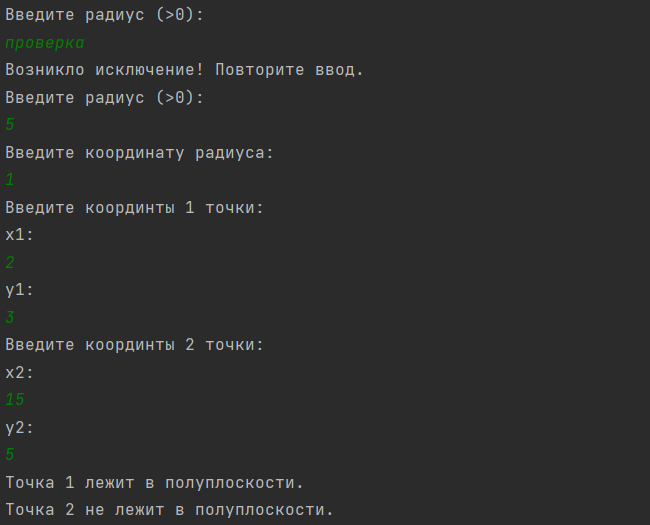
**Блок схема**

****

**Листинг программы**

import kotlin.math.\*  
fun main() {  
 var flag = true  
 while (flag == true) {  
 try {  
 *println*("Введите радиус (>0):")  
 var R = *readln*()!!.*toDouble*()  
 while (R<0.0)  
 {  
 *println*("Радиус должен быть больше нуля!")  
 *println*("Введите радиус (>0):")  
 var R = *readln*()!!.*toDouble*()  
 }  
 *println*("Введите координату радиуса:")  
 var P = *readln*()!!.*toDouble*()  
 *println*("Введите координты 1 точки:")  
 *println*("x1:")  
 var x1 = *readln*()!!.*toDouble*()  
 *println*("y1:")  
 var y1 = *readln*()!!.*toDouble*()  
 *println*("Введите координты 2 точки:")  
 *println*("x2:")  
 var x2 = *readln*()!!.*toDouble*()  
 *println*("y2:")  
 var y2 = *readln*()!!.*toDouble*()  
 //проверка 1 точки  
 if (y1 >0.0 && R > *abs*( *sqrt*( Math.pow( x1 - 0.0, 2.0)+Math.pow( y1 - 0.0, 2.0 ) ) ) )  
 {  
 *println*("Точка 1 лежит в полуплоскости.")  
 }  
 else  
 *println*("Точка 1 не лежит в полуплоскости.")  
 //проверка 2 точки  
 if (y2 >0.0 && R > *abs*( *sqrt*( Math.pow( x2 - 0.0, 2.0)+Math.pow( y2 - 0.0, 2.0 ) ) ) )  
 {  
 *println*("Точка 2 лежит в полуплоскости.")  
 }  
 else  
 *println*("Точка 2 не лежит в полуплоскости.")  
 flag = false  
 }  
 catch (e:Exception)  
 {  
 *println*("Возникло исключение! Повторите ввод.")  
 flag = true  
 }  
 }  
}

**Тестовые ситуации**

****