Министерство образования и молодежной политики Свердловской области



ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

Отчёт по программе «**Практическая 7\_2**»

Выполнил: Карамов Д.Э.

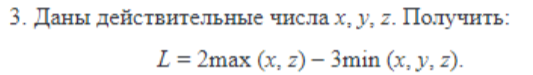
Группа: ПР-24

Преподаватель: Мирошниченко Г.В

2025

**1 Задание**

**Задание:**

****

**Входные данные**

X – double

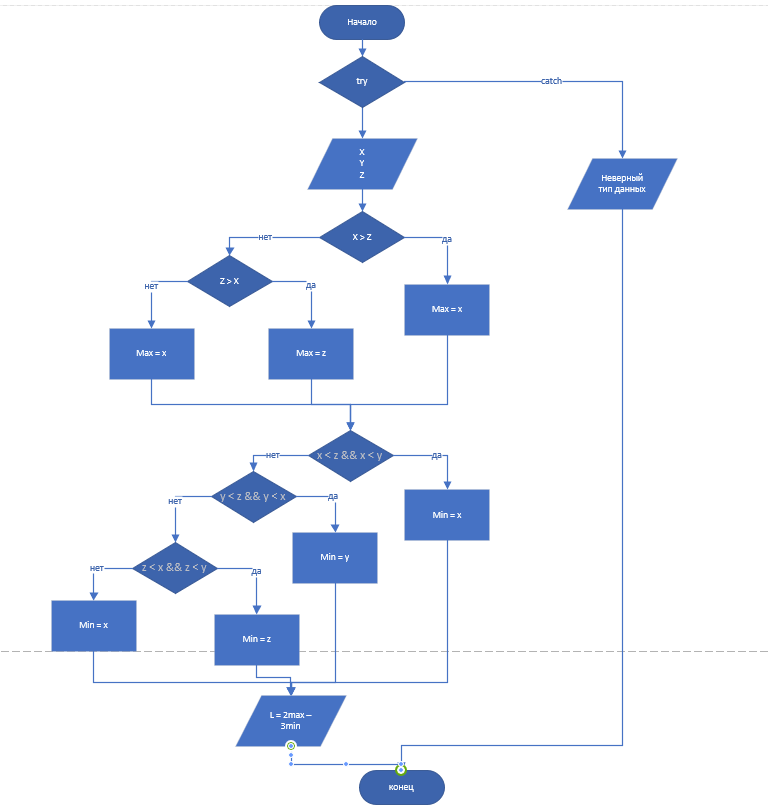
Y – double

Z – double

**Выходные данные**

L – double

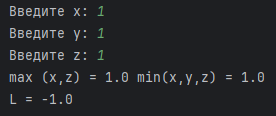
**Блок-схема**

****

**Листинг программы**

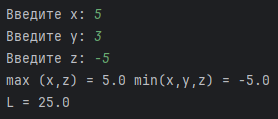
fun main()  
{  
 try  
 {  
 *print*("Введите x: ")  
 val x = *readln*().*toDouble*()  
 *print*("Введите y: ")  
 val y = *readln*().*toDouble*()  
 *print*("Введите z: ")  
 val z = *readln*().*toDouble*()  
 var max:Double; var min:Double  
 when  
 {  
 x > z -> max = x  
 z > x -> max = z  
 else -> max = x  
 }  
 when  
 {  
 x < z && x < y -> min = x  
 y < z && y < x -> min = y  
 z < x && z < y -> min = z  
 else -> min = x  
 }  
 *println*("max (x,z) = " + max + " min(x,y,z) = " + min)  
 *println*("L = " + (2 \* max - 3 \* min))  
 }  
 catch (e:Exception)  
 {  
 *println*("Неверный тип данных")  
 }  
}

**Тестовые ситуации**



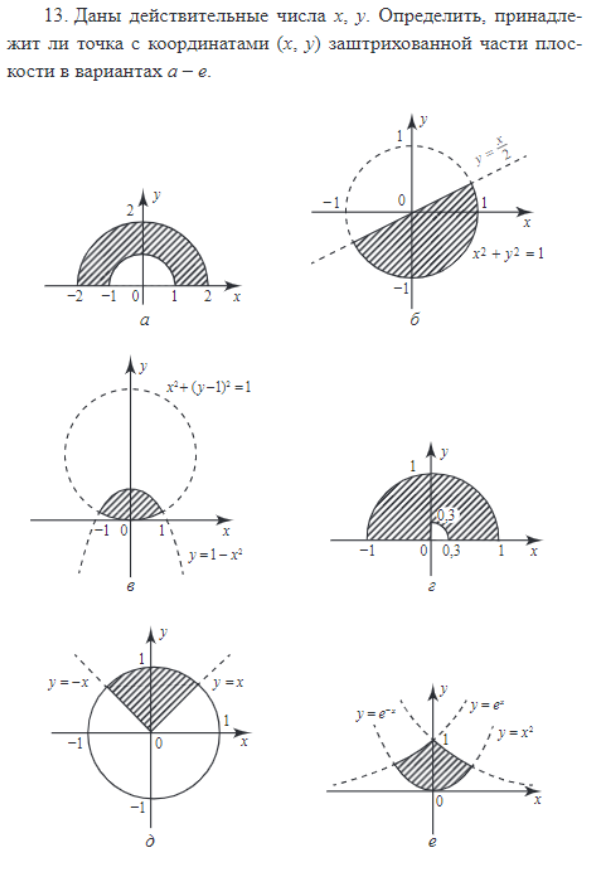


**Результат:**

****

**2 Задание**

**Задание:**

****

**Входные данные**

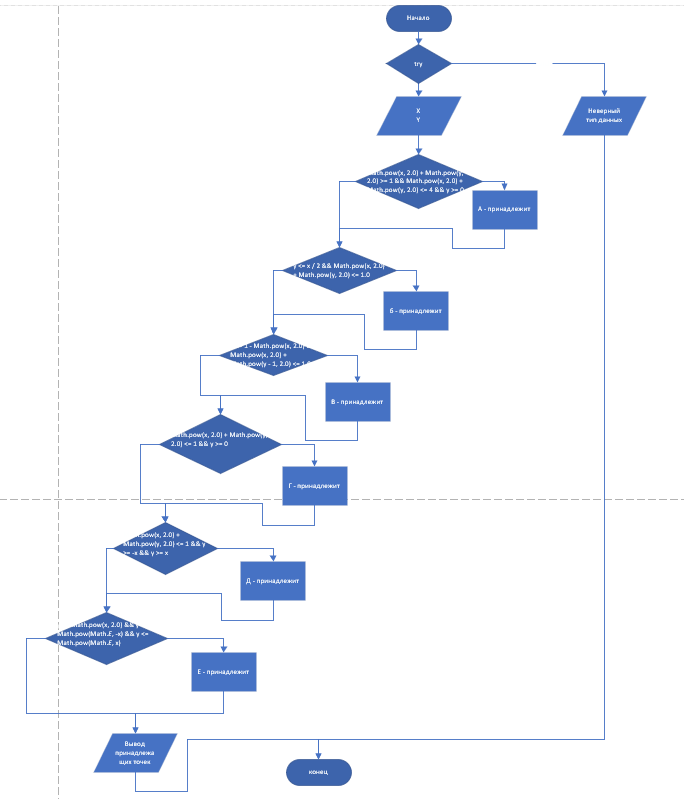
X – double

Y - double

**Выходные данные**

“принадлежит || не принадлежит”

**Блок-схема**



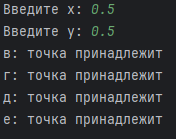
**Листинг программы**

fun main()  
{  
 try  
 {  
 *print*("Введите x: ")  
 val x = *readln*().*toDouble*()  
 *print*("Введите y: ")  
 val y = *readln*().*toDouble*()  
 when  
 {  
 Math.pow(x, 2.0) + Math.pow(y, 2.0) >= 1 && Math.pow(x, 2.0) + Math.pow(y, 2.0) <= 4 && y >= 0 ->  
 {  
 *println*("a: точка принадлежит")  
 }  
 }  
 when  
 {  
 y <= x / 2 && Math.pow(x, 2.0) + Math.pow(y, 2.0) <= 1.0 ->  
 {  
 *println*("б: точка принадлежит")  
 }  
 }  
 when  
 {  
 y <= 1 - Math.pow(x, 2.0) && Math.pow(x, 2.0) + Math.pow(y - 1, 2.0) <= 1.0 ->  
 {  
 *println*("в: точка принадлежит")  
 }  
 }  
 when  
 {  
 Math.pow(x, 2.0) + Math.pow(y, 2.0) <= 1 && y >= 0 ->  
 {  
 if (Math.pow(x, 2.0) + Math.pow(y, 2.0) >= Math.pow(0.3, 2.0) && x >= 0)  
 {  
 *println*("г: точка принадлежит")  
 }  
 }  
 }  
 when  
 {  
 Math.pow(x, 2.0) + Math.pow(y, 2.0) <= 1 && y >= -x && y >= x ->  
 {  
 *println*("д: точка принадлежит")  
 }  
 }  
 when  
 {  
 y >= Math.pow(x, 2.0) && y <= Math.pow(Math.*E*, -x) && y <= Math.pow(Math.*E*, x) ->  
 {  
 *println*("е: точка принадлежит")  
 }  
 }  
 }  
 catch (e:Exception)  
 {  
 *println*("Неверный формат ввода")  
 }  
}

**Тестовые ситуации**

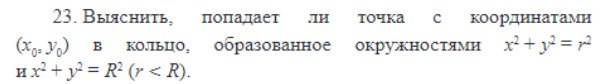


**Результат:**

****

**3 Задание**

**Задание:**

****

**Входные данные**

X – double

Y – double

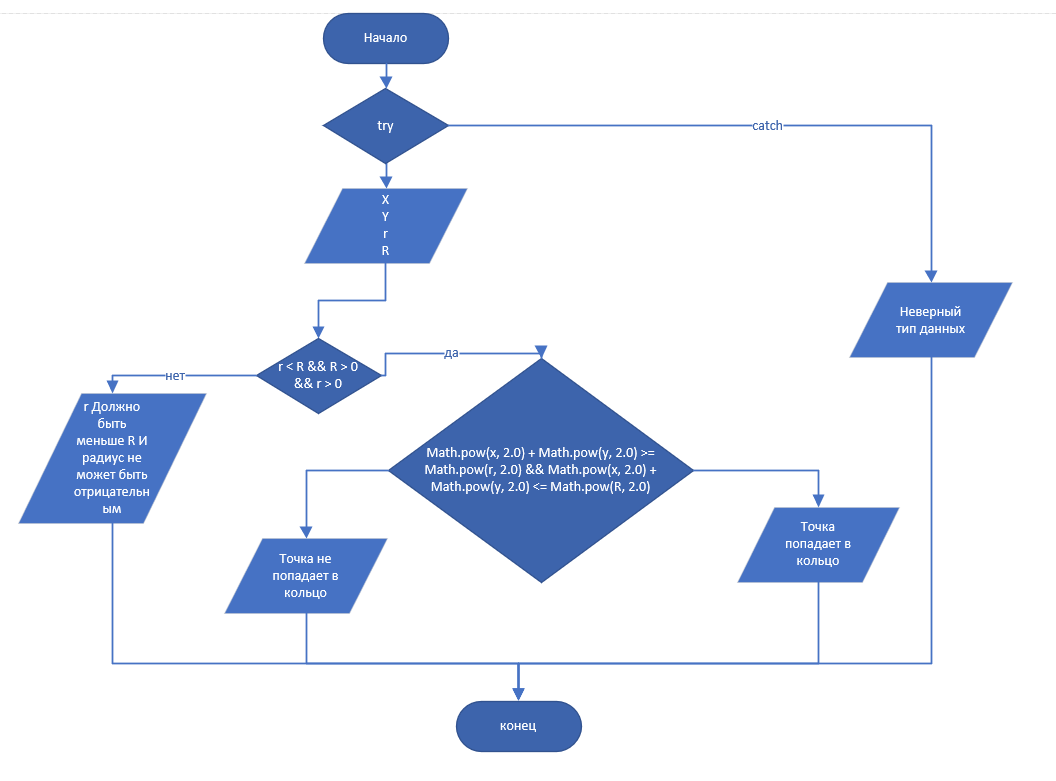
r – double

R - double

**Выходные данные**

“Точка попадает в кольцо || Точка не попадает в кольцо”

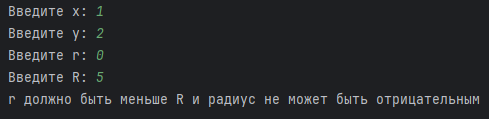
**Блок-схема**



**Листинг программы**

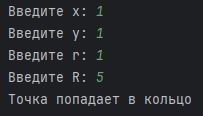
fun main()  
{  
 try  
 {  
 *print*("Введите x: ")  
 val x = *readln*().*toDouble*()  
 *print*("Введите y: ")  
 val y = *readln*().*toDouble*()  
 *print*("Введите r: ")  
 val r = *readln*().*toDouble*()  
 *print*("Введите R: ")  
 val R = *readln*().*toDouble*()  
 when  
 {  
 r < R && R > 0 && r > 0 ->  
 {  
 when  
 {  
 Math.pow(x, 2.0) + Math.pow(y, 2.0) >= Math.pow(r, 2.0) && Math.pow(x, 2.0) + Math.pow(y, 2.0) <= Math.pow(R, 2.0) -> *println*("Точка попадает в кольцо")  
 else -> *println*("Точка не попадает в кольцо")  
 }  
 }  
 else -> *println*("r должно быть меньше R и радиус не может быть отрицательным ")  
 }  
 }  
 catch (e:Exception)  
 {  
 *println*("Неверный формат ввода")  
 }  
}

**Тестовые ситуации**



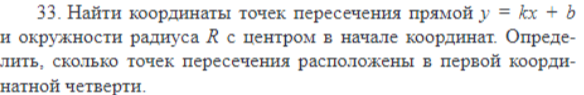


**Результат:**

****

**4 Задание**

**Задание:**

****

**Входные данные**

R – double

K – double

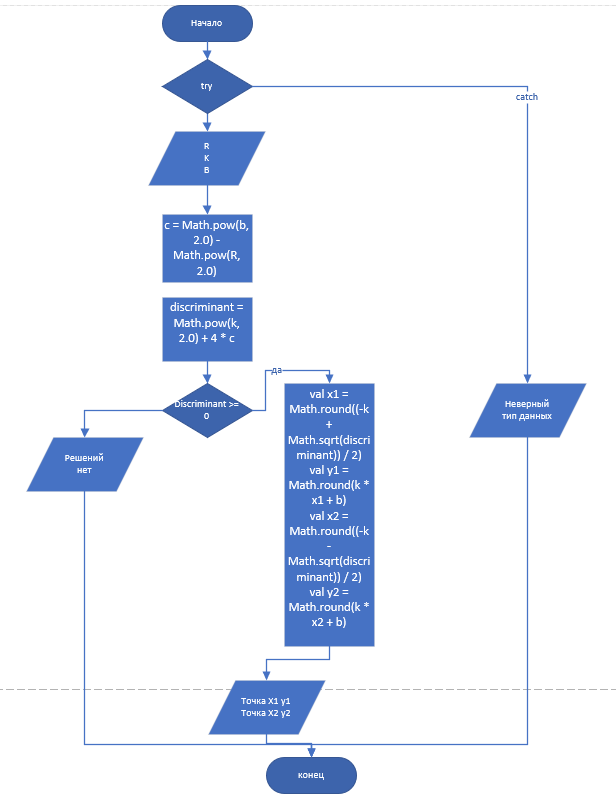
B - double

**Выходные данные**

X1,Y1 - double

X2,Y2 – double – Координаты точек

**Блок-схема**

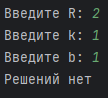


**Листинг программы**

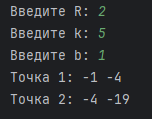
fun main()  
{  
 try  
 {  
 *print*("Введите R: ")  
 val R = *readln*().*toDouble*()  
 *print*("Введите k: ")  
 val k = *readln*().*toDouble*()  
 *print*("Введите b: ")  
 val b = *readln*().*toDouble*()  
 val c = Math.pow(b, 2.0) - Math.pow(R, 2.0)  
 val discriminant = Math.pow(k, 2.0) + 4 \* c  
 when  
 {  
 discriminant >= 0 -> {  
 val x1 = Math.round((-k + Math.sqrt(discriminant)) / 2)  
 val y1 = Math.round(k \* x1 + b)  
 val x2 = Math.round((-k - Math.sqrt(discriminant)) / 2)  
 val y2 = Math.round(k \* x2 + b)  
 *println*("Точка 1: " + x1 + " " + y1)  
 *println*("Точка 2: " + x2 + " " + y2)  
 }  
 else -> *println*("Решений нет")  
 }  
 }  
 catch (e:Exception)  
 {  
 *println*("Неверный тип данных")  
 }  
}

**Тестовые ситуации**



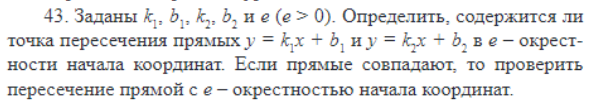


**Результат:**

****

**5 Задание**

**Задание:**

****

**Входные данные**

**K1 – double**

**B1 – double**

**K2 – double**

**B2 – double**

**E - double**

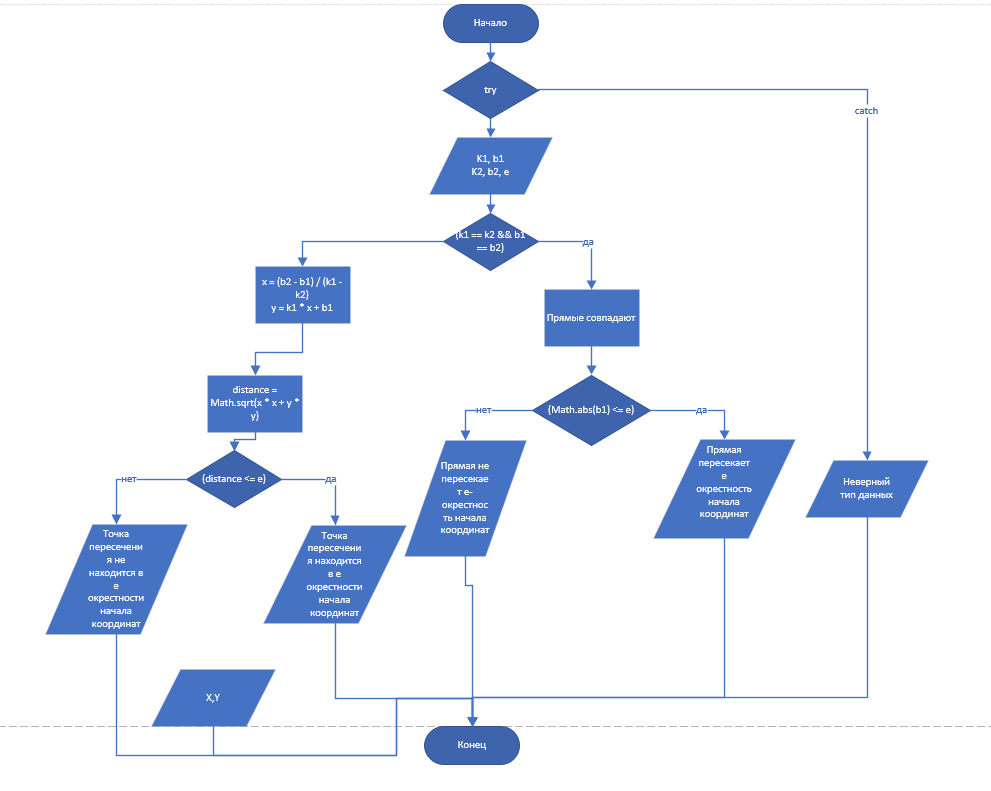
**Выходные данные**

**“Совпадают || не совпадают” + “Пересекаются || не пересекаются”**

**X – double**

**Y – double Координаты точки если пересекаются**

**Блок-схема**

****

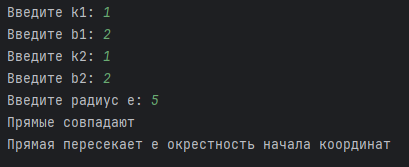
**Листинг программы**

fun main() {  
 try {  
 *println*("Введите k1: ")  
 val k1 = *readln*().*toDouble*()  
 *println*("Введите b1: ")  
 val b1 = *readln*().*toDouble*()  
 *println*("Введите k2: ")  
 val k2 = *readln*().*toDouble*()  
 *println*("Введите b2: ")  
 val b2 = *readln*().*toDouble*()  
 *println*("Введите радиус e: ")  
 val e = *readln*().*toDouble*()  
 when {  
 (k1 == k2 && b1 == b2) -> {  
 *println*("Прямые совпадают")  
 when {  
 (Math.abs(b1) <= e) -> *println*("Прямая пересекает e окрестность начала координат")  
 else -> *println*("Прямая не пересекает e окрестность начала координат")  
 }  
 }  
 else -> {  
 val x = (b2 - b1) / (k1 - k2)  
 val y = k1 \* x + b1  
 val distance = Math.sqrt(x \* x + y \* y)  
 when {  
 (distance <= e) -> *println*("Точка пересечения находится в e окрестности начала координат" + x + " " + y)  
 else -> *println*("Точка пересечения не находится в e окрестности начала координат" + x + " " + y)  
 }  
 }  
 }  
 }  
 catch (e:Exception)  
 {  
 *println*("Неверный тип данных")  
 }  
}

**Тестовые ситуации**



**Результат:**

****