Министерство образования и молодежной политики Свердловской области



ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

Отчёт по программе «Практическое занятие 7 (3 задание)»

Выполнил: Форрат Лия Константиновна

Группа: ПР-21

Преподаватель: Мирошниченко Г.В

2023

**Задание:**

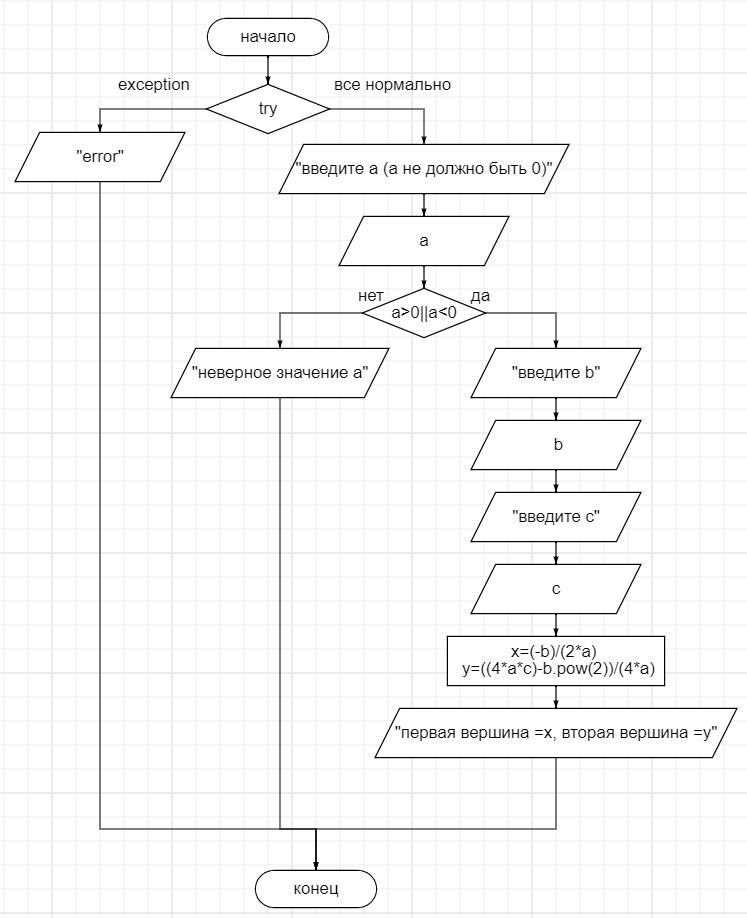
****

**Входные и выходные данные:**

**Double a,b,c -значения a,b,c**

**Выводятся значения вершин x,y- double**

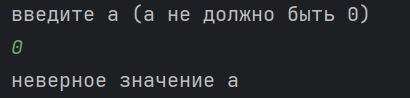
**Блок-схема**

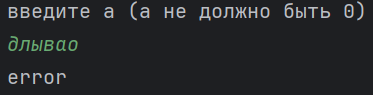
****

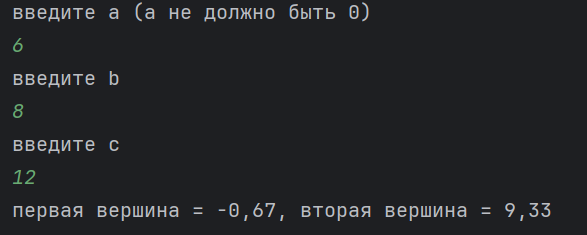
**Листинг программы (если есть)**

import kotlin.math.pow  
fun main()  
{  
 try{  
 val x: Double  
 val y: Double  
 *println*("введите а (a не должно быть 0)")  
 val a=*readLine*()!!.*toDouble*()  
 if (a>0||a<0) {  
 *println*("введите b")  
 val b=*readLine*()!!.*toDouble*()  
 *println*("введите c")  
 val c=*readLine*()!!.*toDouble*()  
 x= (-b)/(2\*a)  
 y=((4\*a\*c)-b.*pow*(2))/(4\*a)  
 *println*("первая вершина = ${String.*format*("%.2f",x)}, вторая вершина = ${String.*format*("%.2f",y)}")  
 }  
 else {  
 *println*("неверное значение а")  
 }  
 }  
 catch (e: Exception)  
 {  
 *println*("error")  
 }  
}

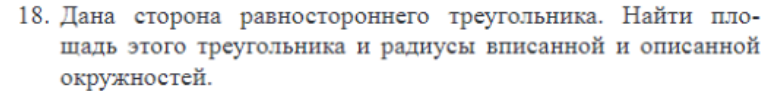
**Тестовые ситуации**

****

****

****

**Задание:**

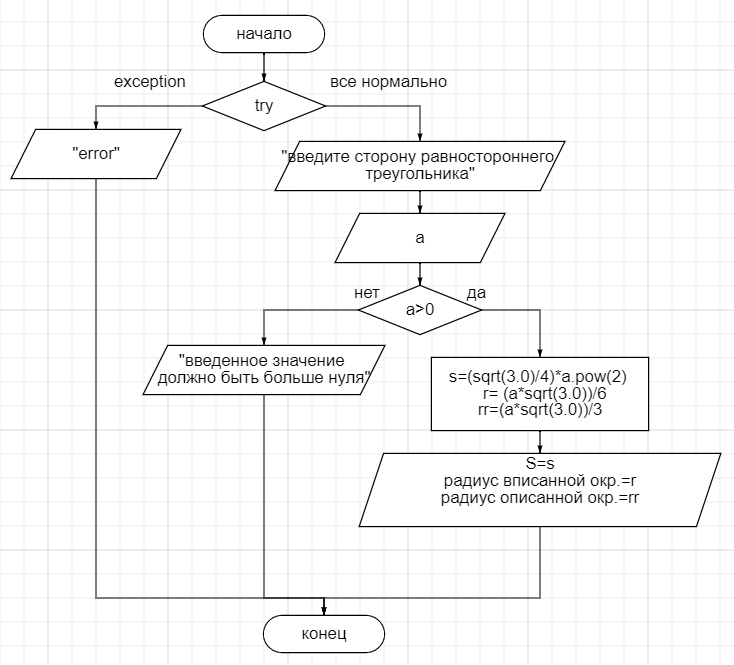
****

**Входные и выходные данные:**

**Double a-сторона равностороннего треугольника**

**Выводятся: s-double площадь треугольника, r-double радус вписанной оркужности, rr-double радус описанной оркужности**

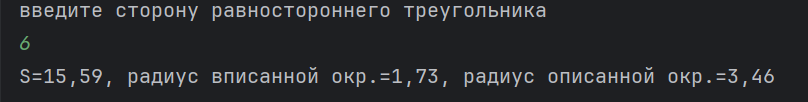
**Блок-схема**

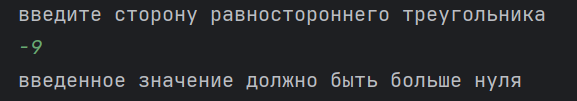
****

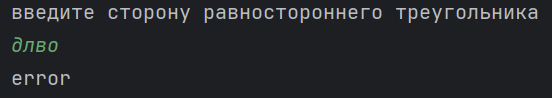
**Листинг программы (если есть)**

import kotlin.math.\*  
fun main()  
{  
 try{  
 val s: Double  
 val r: Double  
 val rr: Double  
 println("введите сторону равностороннего треугольника")  
 val a=readLine()!!.toDouble()  
 if (a>0) {  
 s=(sqrt(3.0)/4)\*a.pow(2)  
 r= (a\*sqrt(3.0))/6  
 rr=(a\*sqrt(3.0))/3  
 println("S=${String.format("%.2f",s)}, радиус вписанной окр.=${String.format("%.2f",r)}, радиус описанной окр.=${String.format("%.2f",rr)}")  
 }  
 else {  
 println("введенное значение должно быть больше нуля")  
 }  
 }  
 catch (e: Exception)  
 {  
 println("error")  
  
 }  
}

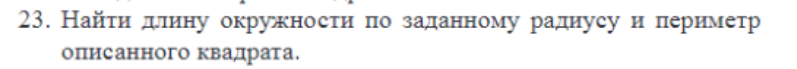
**Тестовые ситуации**







**Задание:**

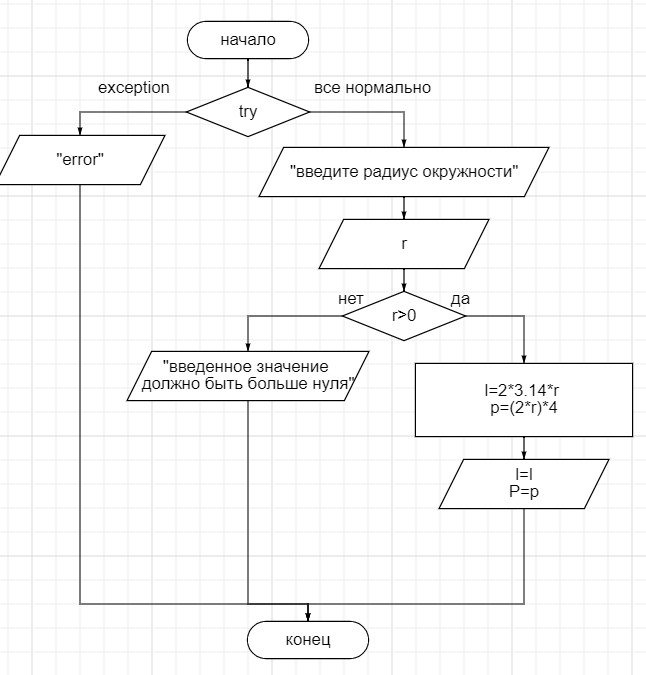
****

**Входные и выходные данные:**

**Double r-радиус окружности**

**Выводятся: l-длина окружности, p-периметр описанного квадрата**

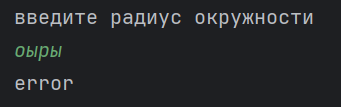
**Блок-схема**

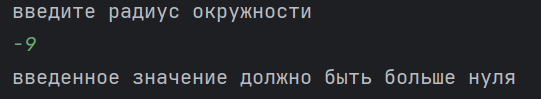
****

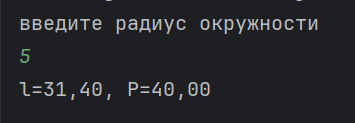
**Листинг программы (если есть)**

fun main()  
{  
 try{  
 val l: Double  
 val p: Double  
 *println*("введите радиус окружности")  
 val r=*readLine*()!!.*toDouble*()  
 if (r>0) {  
 l=2\*3.14\*r  
 p=(2\*r)\*4  
 *println*("l=${String.*format*("%.2f",l)}, P=${String.*format*("%.2f",p)}")  
 }  
 else {  
 *println*("введенное значение должно быть больше нуля")  
 }  
 }  
 catch (e: Exception)  
 {  
 *println*("error")  
 }  
}

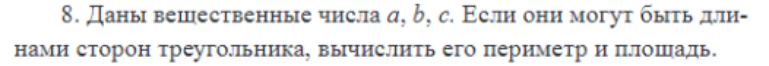
**Тестовые ситуации**

****

****

****

**Задание:**

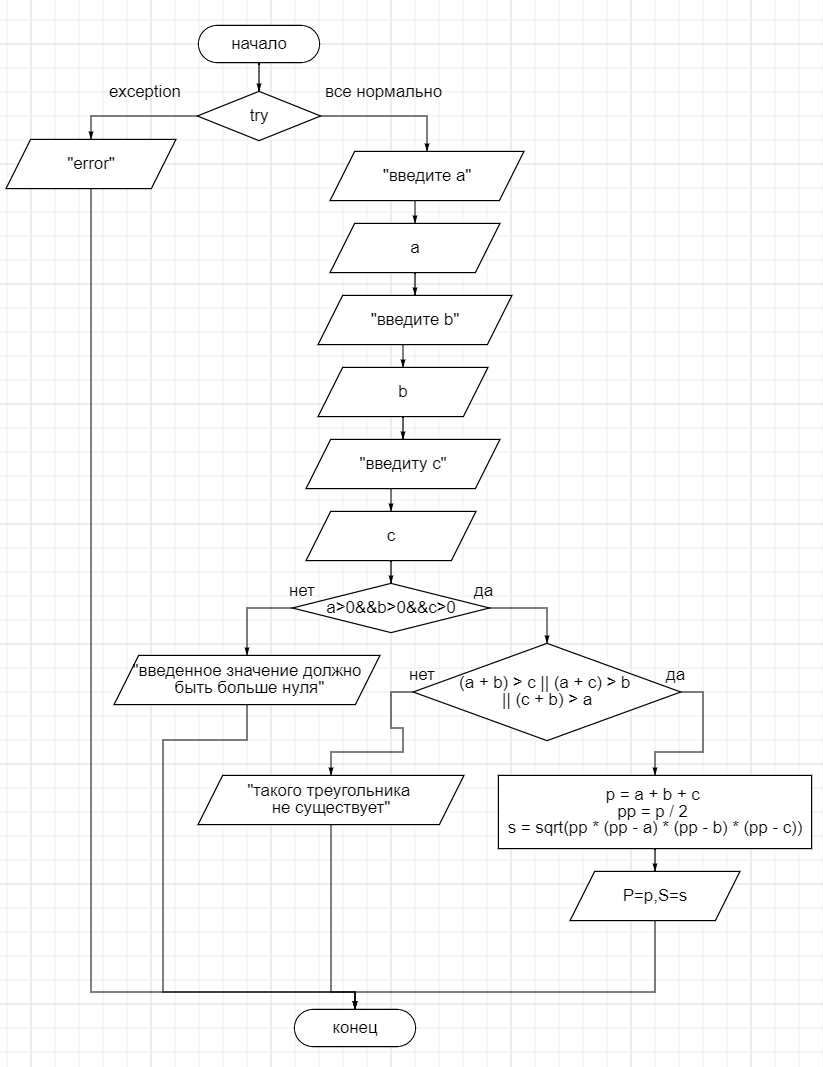
****

**Входные и выходные данные:**

**Double a,b,c-стороны треугольника**

**Выводится либо сообщение-string либо double-p,s-периметр и площадь треугольника**

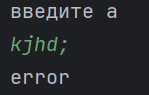
**Блок-схема**

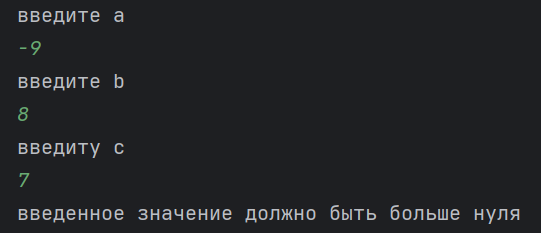
****

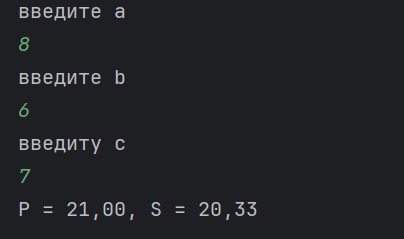
**Листинг программы (если есть)**

import kotlin.math.\*  
fun main()  
{  
 try{  
 val p: Double  
 val s: Double  
 *println*("введите а")  
 val a=*readLine*()!!.*toDouble*()  
 *println*("введите b")  
 val b=*readLine*()!!.*toDouble*()  
 *println*("введиту c")  
 val c=*readLine*()!!.*toDouble*()  
 if (a>0&&b>0&&c>0) {  
 if ((a + b) > c || (a + c) > b || (c + b) > a) {  
 p = a + b + c  
 val pp = p / 2  
 s = *sqrt*(pp \* (pp - a) \* (pp - b) \* (pp - c))  
 *println*("P = ${String.*format*("%.2f", p)}, S = ${String.*format*("%.2f", s)}")  
 } else {  
 *println*("такого треугольника не существует")  
 }  
 }  
 else {*println*("введенное значение должно быть больше нуля")}  
 }  
 catch (e: Exception)  
 {  
 *println*("error")  
 }  
}

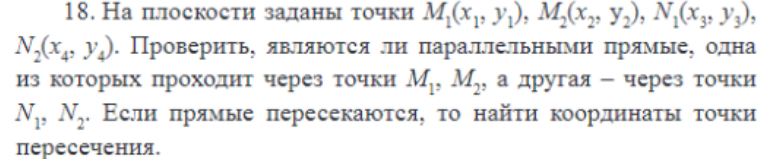
**Тестовые ситуации**

****

****

****

**Задание:**

****

**Входные и выходные данные:**

**Double x1,y1-координаты M1**

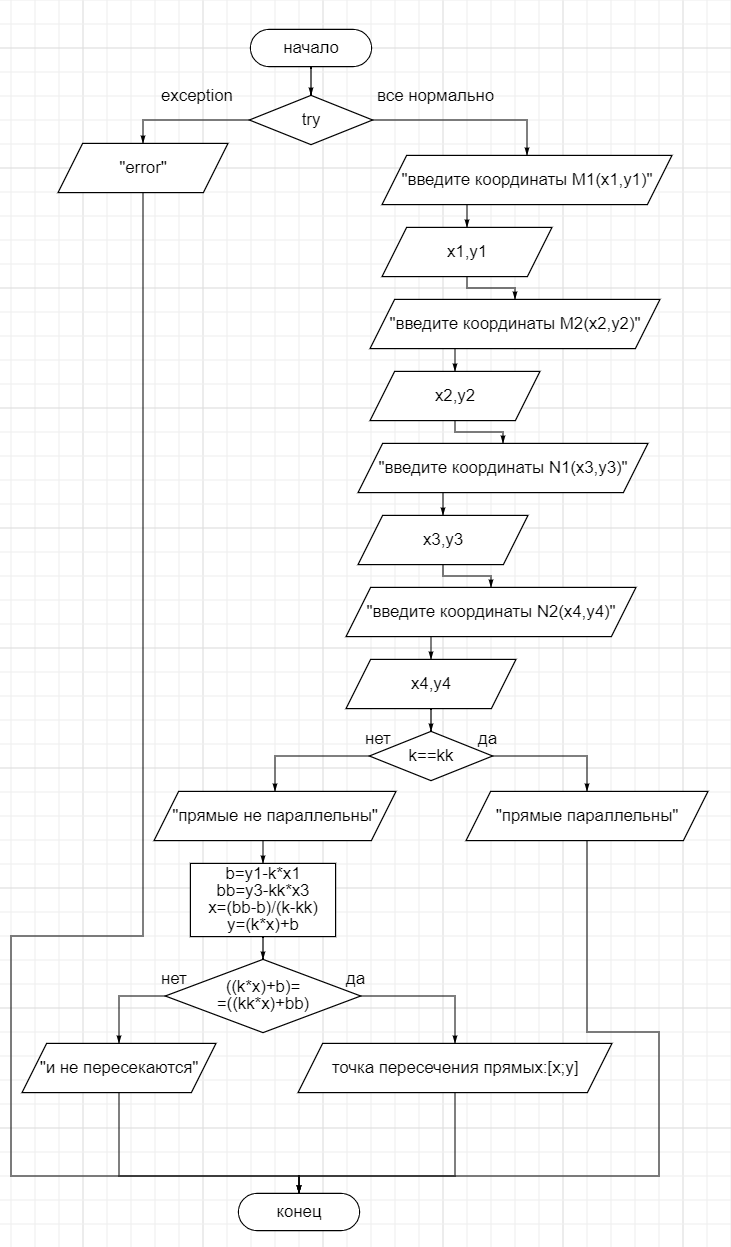
**Double x2,y2-координаты M2**

**Double x3,y3-координаты N1**

**Double x4,y4-координаты N2**

**Выводятся сообщение тип string или точки пересечения x,y-double**

**Блок-схема**

****

**Листинг программы (если есть)**

fun main()  
{  
 try{  
 *println*("введите координаты M1(x1,y1)")  
 val x1=*readLine*()!!.*toDouble*()  
 val y1=*readLine*()!!.*toDouble*()  
 *println*("введите координаты M2(x2,y2)")  
 val x2=*readLine*()!!.*toDouble*()  
 val y2=*readLine*()!!.*toDouble*()  
 *println*("введите координаты N1(x3,y3)")  
 val x3=*readLine*()!!.*toDouble*()  
 val y3=*readLine*()!!.*toDouble*()  
 *println*("введите координаты N2(x4,y4)")  
 val x4=*readLine*()!!.*toDouble*()  
 val y4=*readLine*()!!.*toDouble*()  
 val k=(y2-y1)/(x2-x1)  
 val kk=(y4-y3)/(x4-x3)  
 if (k==kk) {  
 *println*("прямые параллельны")  
 }  
 else {  
 *println*("прямые не параллельны")  
 val b=y1-k\*x1  
 val bb=y3-kk\*x3  
 val x=(bb-b)/(k-kk)  
 val y=(k\*x)+b  
 if (((k\*x)+b)==((kk\*x)+bb))  
 {  
 *println* ("точка пересечения прямых:[${String.*format*("%.2f",x)};${String.*format*("%.2f",y)}]")  
 }  
 else {*println*("и не пересекаются")}  
 }  
 }  
 catch (e: Exception)  
 {  
 *println*("error")  
 }  
}

**Тестовые ситуации**

