Министерство образования и молодежной политики Свердловской области

ГАПОУ СО «Екатеринбургский колледж транспортного строительства»

# Отчет по программе «Практическое занятие 12 (корутины и потоки)»

Выполнил: Меньшиков Артур Андреевич

Группа: ПР-22

Преподаватель: Мирошниченко Г. В.

2023

**Задание:**

Задание: 1. Реализовать через корутины вызов функций нажатий и

параллельного получения данных из n функций (n вводит пользователь).

Затем, нужно их объединять и отображать.

Задание: 2. Создайте программу для загрузки участников для всех репозиториев

данной организации, с сортировкой пользователей по количеству их

репрозиториев.

**Входные и выходные данные**

Задание 1:

var sv = int/целочисленный тип

var k = int/целочисленный тип

var b = int/целочисленный тип

Задание 2:

var sv = int/целочисленный тип

var k = int/целочисленный тип

var kol = int/целочисленный тип

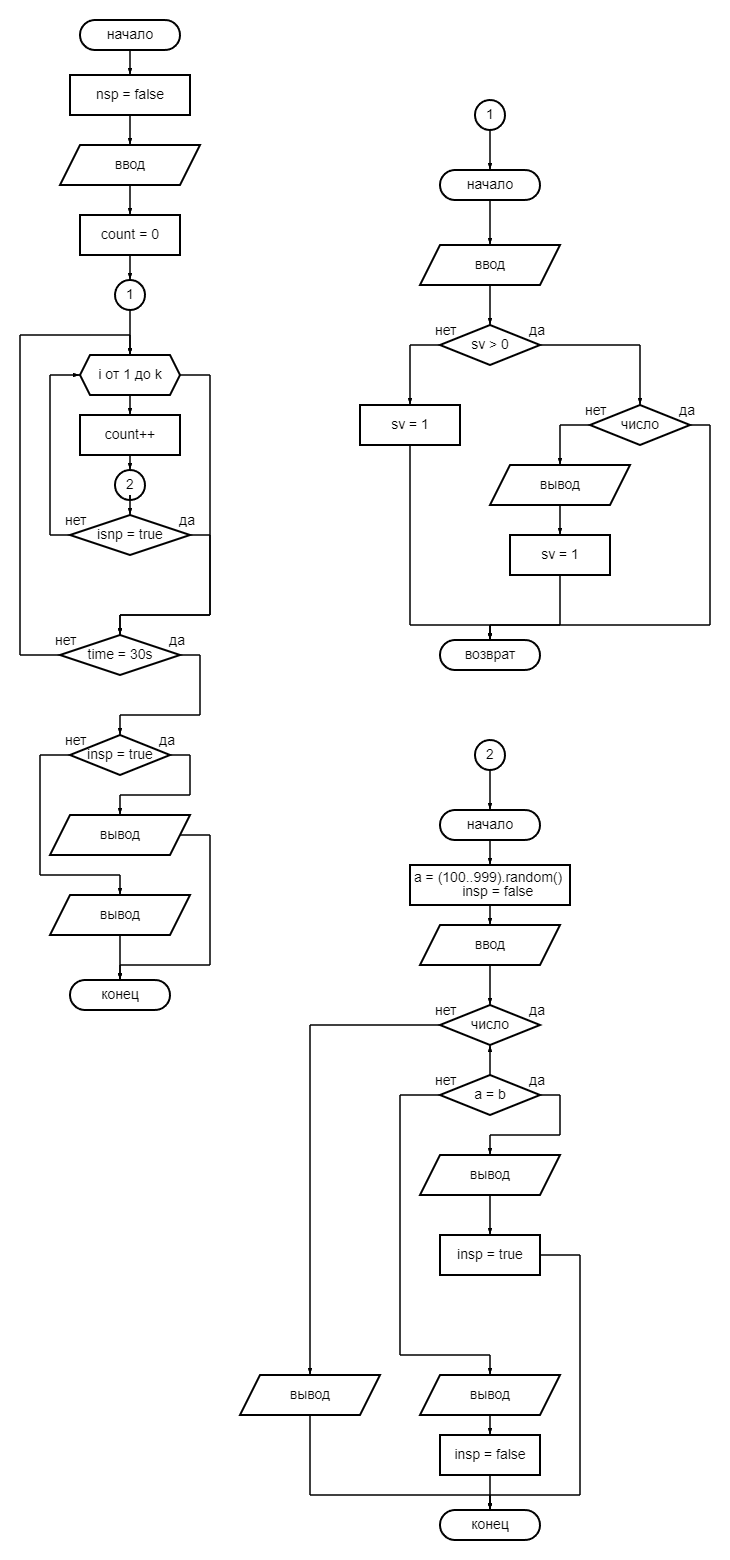
var rep = int/целочисленный тип

var pass = string/строковый тип

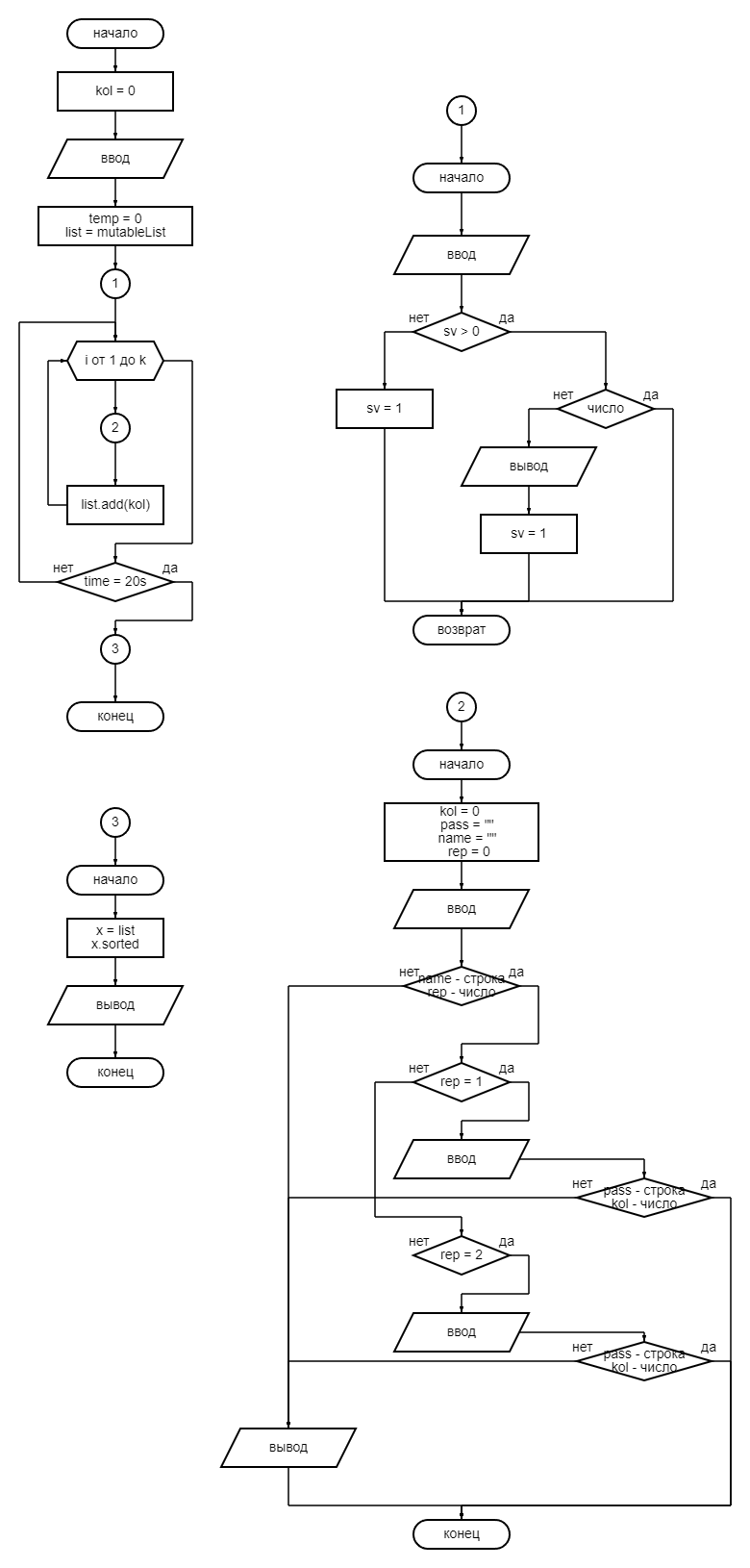
var name = string/строковый тип

**Блок-схема**

Задание 1:



Задание 2:



**Листинг программы (если есть)**

import kotlinx.coroutines.\*  
  
suspend fun main(){  
 val a: `class` = `class`()  
 var count = 0  
 var k = a.input()  
 GlobalScope.*launch*{  
 for(i in 1..k){  
 count++  
 *println*("$i раз")  
 a.findPass()  
 if(a.insp == true){  
 break  
 }  
 }  
 }  
 *runBlocking* {  
 delay(30000)  
 if(a.insp == true){  
 *println*("вы отгадали пароль за $count попытку")  
 }  
 else *println*("вы не отгадали пароль")  
 }  
  
}

import kotlin.random.Random  
  
class `class` {  
 var a = (100..999).*random*()  
 var insp = false  
  
 fun input():Int {  
 try {  
 *println*("введите сколько потворений будет")  
 var sv = *readln*()!!.*toInt*()  
 if(sv>0)  
 return sv  
 else *println*("число должно быть больше 0")  
 return 1  
 }catch (e:Exception) {*println*("некоректный ввод")}  
 return 1  
 }  
 fun findPass(){  
 try {  
 *println*("Угадайте пароль \*\*\* ($a)");  
 var b = *readln*()!!.*toInt*()  
 if (a == b) {  
 *println*("вы угадали пароль")  
 insp = true  
 } else {  
 *println*("поробуйте снова")  
 insp = false  
 }  
 }catch (e:Exception) { *println*("некоректный ввод") }  
 }  
}

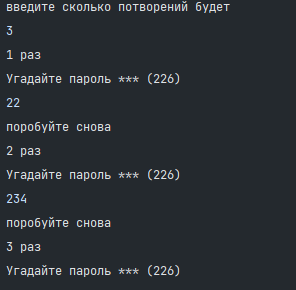
Задание 2:

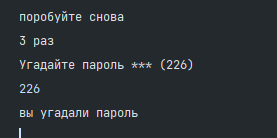
import kotlinx.coroutines.GlobalScope  
import kotlinx.coroutines.delay  
import kotlinx.coroutines.launch  
import kotlinx.coroutines.runBlocking  
  
fun main(){  
 val list = *mutableListOf*<Int>()  
 val a: `class2` = `class2`()  
 var k = a.input()  
 var temp = 0  
 GlobalScope.*launch*{  
 for(i in 1..k){  
 *println*("$i раз")  
 a.info()  
 list.add(a.kol)  
 }  
 }  
 *runBlocking* {  
 delay(20000)  
 a.sort(list)  
 }  
}

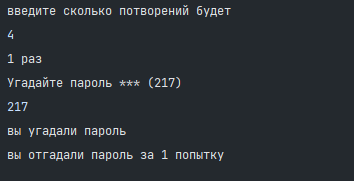
class class2 {  
  
 fun input():Int {  
 try {  
 *println*("введите сколько потворений будет")  
 var sv = *readln*()!!.*toInt*()  
 if(sv>0)  
 return sv  
 else *println*("число должно быть больше 0")  
 return 1  
 } catch (e:Exception) { *println*("некоректный ввод") }  
 return 1  
 }  
  
 var kol = 0  
 var pass = ""  
 var name = ""  
 var rep = 0  
  
 fun info(){  
 try {  
 *println*("введите имя участника")  
 name = *readln*()!!.toString()  
 *println*("репозиторий: приватный - (1), откртый - (2)")  
 rep = *readln*()!!.*toInt*()  
 if (rep == 1) {  
 *println*("введите пароль репозитория")  
 pass = *readln*()  
 *println*("введите сколько у вас репозиториев")  
 kol = *readln*()!!.*toInt*()  
  
 } else if (rep == 2) {  
 pass = ""  
 *println*("введите сколько у вас репозиториев")  
 kol = *readln*()!!.*toInt*()  
  
 } else {  
 *println*("нужно нажать: приватный - (1), откртый - (2)")  
 }  
 } catch (e:Exception) { *println*("некоректный ввод") }  
 }  
  
 fun sort(x: *List*<Int>){  
 x.*sorted*()  
 *println*(x.*sorted*())  
 }  
  
 fun output(t:Int){  
 *println*("имя - $name; репозиторий - $rep; пароль - $pass; кол-во репозиториев - $t")  
 }  
}

**Тестовые ситуации**

Задание 1:







Задание 2:

