МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ФАКУЛЬТЕТ ЭЛЕКТРОННО-ИНФОРМАЦИОННЫХ СИСТЕМ

Кафедра интеллектуальных информационных технологий

Отчет по лабораторной работе №3

Выполнил

Василюк П. О.,

студент группы ПО-5

Проверил

Крощенко А.А.,

ст. преп. Кафедры ИИТ,

«\_\_» \_\_\_\_\_\_\_\_ 2021 г.

Брест, 2021

**Цель работы:** научиться создавать и использовать классы в программах на языке программирования Kotlin

**Вариант 2**

**Задание 1. Равносторонний треугольник, заданный длинами сторон –** Предусмотреть возможность определения площади и периметра, а так же логический метод, определяющий существует или такой треугольник. Конструктор должен позволить создавать объекты с начальной инициализацией. Реализовать метод equals, выполняющий сравнение объектов данного типа.

**Задание 2.**

**Автоматизированная система проката автомобилей**

Составить программу, которая хранит и обрабатывает информацию о прокате автомобилей. О каждом автомобиле (Car) содержится следующая информация:

• id;

• Марка;

• Модель;

• Год выпуска;

• Цвет;

• Цена;

• Регистрационный номер;

• Номер машины.

• ФИО лица, взявшего на прокат (при наличии);

• Номер паспорта лица-арендатора (при наличии).

Программа должна обеспечить вывод списков:

• автомобилей;

• автомобилей заданной марки;

• автомобилей заданной модели, которые эксплуатируются больше n лет;

• автомобилей заданного года выпуска, цена которых больше указанной;

• автомобилей, взятых на прокат;

• автомобилей, взятых на прокат с выводом личной информации об арендаторах.

**Выполнение:**

**Код программы:**

Maint.kt

@file:JvmName("main")

package org.example.main

import java.io.File

import java.io.InputStream

public class Prog() {

fun trng() {

val triangle = Triangle()

triangle.setTriangle(3.0, 3.0, 3.0)

val triangle2: Triangle = Triangle(3.0, 3.0, 3.0)

val triangle3: Triangle = Triangle()

triangle3.Side1 = 5.0

triangle3.Side2 = 5.0

triangle3.Side3 = 5.0

val triangles = arrayOf(triangle, triangle2, triangle3)

for (i in triangles.indices) {

println("Triangle " + (i + 1) + ": " + triangles[i])

if (triangles[0].isRectangular()) println("Triangle " + (i + 1) + " - equilateral") else println("Triangle " + (i + 1) + " - inequilateral")

println("Perimeter of a triangle " + (i + 1) + ": " + triangles[i].perimeter())

System.out.format("Square of a triangle %d: %.3f \n", i + 1, triangles[i].square())

}

if (triangle.equals(triangle2)) println("Triangles 1 and 2 - equals") else println("Triangles 1 and 2 - not equals")

println("\n\n")

}

fun veh() {

val inputStream: InputStream = File("file.txt").inputStream()

val lineList = mutableListOf<String>()

var splitList = arrayListOf<List<String>>()

inputStream.bufferedReader().useLines { lines -> lines.forEach { lineList.add(it)} }

lineList.forEach{splitList.add(it.split("\t|"))}

var veh = arrayListOf<Vehicles>()

for(i in 0 until splitList.size){

if(splitList[i].size < 9){

veh.add(Vehicles(splitList[i][0].toInt(), splitList[i][3].toInt(), splitList[i][5].toInt(), splitList[i][6].toInt(),

splitList[i][7].toInt(), splitList[i][1], splitList[i][2], splitList[i][4]))

} else {

veh.add(Vehicles(splitList[i][0].toInt(), splitList[i][3].toInt(), splitList[i][5].toInt(), splitList[i][6].toInt(),

splitList[i][7].toInt(), splitList[i][1], splitList[i][2], splitList[i][4], splitList[i][8], splitList[i][9]))

}

}

println("List of vehicles:")

for(j in 0 until splitList.size){

veh[j].print()

}

println("\nList of vehicles (brand - BMW):")

for(j in 0 until splitList.size){

if(veh[j].getBrand.contains("BMW"))

veh[j].print()

}

println("\nList of vehicles (brand - Mercedes, exploitation - 10 years)")

for(j in 0 until splitList.size){

if(veh[j].getBrand.contains("Mercedes") && (2021 - veh[j].getYeOfIss) > 10 )

veh[j].print()

}

println("\nList of vehicles (year of issue - 2015, price - 10000)")

for(j in 0 until splitList.size){

if(veh[j].getYeOfIss == 2015 && veh[j].getPrice > 10000 )

veh[j].print()

}

println("\nList of vehicles (rented)")

for(j in 0 until splitList.size){

if(veh[j].getFIO != "none" && veh[j].getPassNum != "none")

veh[j].print()

}

}

}

fun main() {

Prog().trng()

Prog().veh()

}

**Triangle.kt**

package org.example.main

import kotlin.math.pow

public class Triangle {

private var side1: Double = 0.0

private var side2: Double = 0.0

private var side3: Double = 0.0

constructor(side\_1: Double, side\_2: Double, side\_3: Double)

{

this.side1 = side\_1

this.side2 = side\_2

this.side3 = side\_3

}

constructor()

{

this.side1 = 0.0

this.side2 = 0.0

this.side3 = 0.0

}

public fun setTriangle(side\_1: Double, side\_2: Double, side\_3: Double)

{

this.side1 = side\_1

this.side2 = side\_2

this.side3 = side\_3

}

public fun square(): Double

{

return (3.0.pow(1.0/2.0) / 4 \* side1.pow(2.0))

}

public fun perimeter(): Double

{

return side1 \* 3

}

public fun isRectangular(): Boolean {

return (side1 == side2 && side2 == side3)

}

var Side1: Double?

get() = side1;

set(value) {

if (value != null) {

side1 = value

}

}

var Side2: Double?

get() = side2;

set(value) {

if (value != null) {

side2 = value

}

}

var Side3: Double?

get() = side3;

set(value) {

if (value != null) {

side3 = value

}

}

override fun equals(o: Any?): Boolean {

if (this === o) return true

if (o == null || javaClass != o.javaClass) return false

val triangle: Triangle = o as Triangle

return side1.equals(triangle.side1) && side2.equals(triangle.side2) && side3.equals(triangle.side3)

}

override fun toString(): String

{

return "sides = $side1, $side2, $side3"

}

}

**Vehicle.kt**

package org.example.main

class Vehicles {

private var id: Int = 0

private var yearIss: Int = 0

private var pr: Int = 0

private var regNum: Int = 0

private var carNum: Int = 0

private lateinit var pasNum: String

private lateinit var brand: String

private lateinit var mod: String

private lateinit var col: String

private lateinit var fio: String

constructor(id: Int, yearIss: Int, pr: Int, regNum: Int, carNum: Int,

brand: String, mod: String, col: String, fio: String = "none", pasNum: String = "none") {

this.id = id

this.yearIss = yearIss

this.pr = pr

this.regNum = regNum

this.carNum = carNum

this.pasNum = pasNum

this.brand = brand

this.mod = mod

this.col = col

this.fio = fio

}

constructor(){ }

var getBrand: String

get() = brand

set(value) {

brand = value

}

var getYeOfIss: Int

get() = yearIss

set(value) {

yearIss = value

}

var getPrice: Int

get() = pr

set(value) {

pr = value

}

var getFIO: String

get() = fio

set(value) {

fio = value

}

var getPassNum: String

get() = pasNum

set(value) {

pasNum = value

}

override fun toString(): String {

return "Vehicle{" +

"Id='" + id + '\'' +

", Brand='" + brand + '\'' +

", Model='" + mod + '\'' +

", Year of Issue='" + yearIss + '\'' +

", Color=" + col +

", Price=" + pr +

", Registration Number='" + regNum + '\'' +

", Car Number='" + carNum + '\'' +

", FIO='" + fio + '\'' +

", Passport Number='" + pasNum + '\'' +

'}'

}

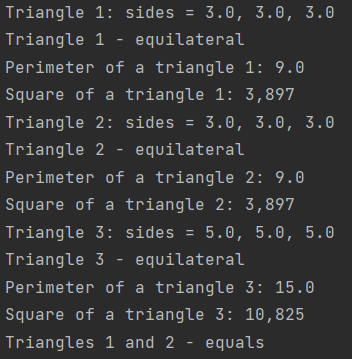
fun print() {

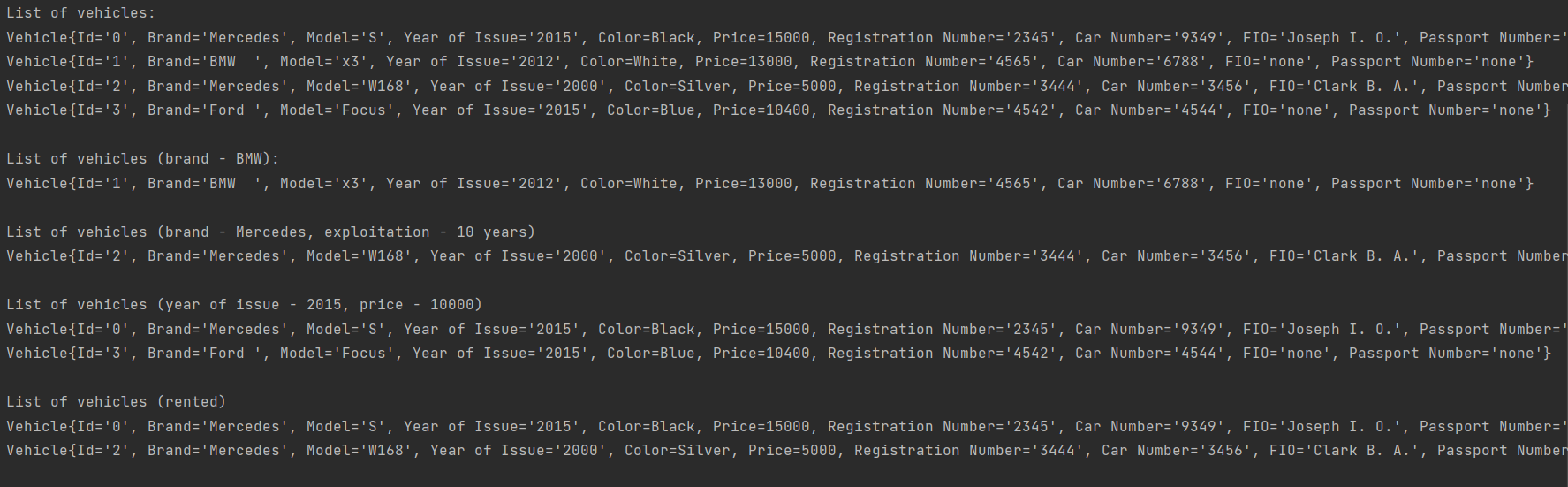
println(this.toString())

}

}

**Результаты:**

****

****

**Вывод:** научились создавать и использовать классы в программах на языке программирования Kotlin