Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

Информатики и радиоэлектроники»

Учебная дисциплина: «Программирование мобильных информационных систем»

Отчет

по выполнению лабораторной работы «Функциональное программирование и лямбда-выражения»

Выполнил студент: гр. 210101 Войтович Е.А.

Проверил: Усенко Ф.В.

Минск 2024

**1. Цель работы:** ознакомиться с основами функционального программирования в Kotlin, изучить лямбда-выражения, анонимные функции и замыкания. Научиться использовать эти концепции для написания более гибкого и читаемого кода.

**2. Задание:** напишите программу, которая создает магический квадрат заданного размера (матрица, где суммы чисел в каждой строке, столбце и диагонали равны). Программа должна проверять, что введенное число является допустимым для создания магического квадрата.

**3. Оснащение работы:** техническое задание, технический проект, ЭВМ, Kotlin, Itelij IDEA.

import java.io.File

import javax.script.ScriptEngineManager

val engine = ScriptEngineManager().getEngineByExtension("kts")

?: throw IllegalStateException("Kotlin Script Engine не найден.")

fun extractVariables(expression: String): List<String> {

val regex = Regex("\\b[a-zA-Z]+\\b(?!\\s\*\\()")

return regex.findAll(expression).map { it.value }.distinct().toList()

}

fun createCustomOperation(expression: String, variables: List<String>): (Map<String, Double>) -> Double {

return { values: Map<String, Double> ->

var formula = expression

variables.forEach { variable ->

formula = formula.replace("\\b$variable\\b".toRegex(), values[variable].toString())

}

val enhancedFormula = formula.replace("pow", "Math.pow")

engine.eval(enhancedFormula) as Double

}

}

fun calculate(values: Map<String, Double>, operation: (Map<String, Double>) -> Double): Double {

return operation(values)

}

fun saveOperations(operations: Map<String, String>, fileName: String) {

File(fileName).printWriter().use { out ->

operations.forEach { (name, expression) -> out.println("$name=$expression") }

}

}

fun loadOperations(fileName: String): MutableMap<String, String> {

val operations = mutableMapOf<String, String>()

if (File(fileName).exists()) {

File(fileName).forEachLine { line ->

val (name, expression) = line.split("=")

operations[name] = expression

}

}

return operations

}

fun main() {

val operations = loadOperations("operations.txt").toMutableMap().apply {

putIfAbsent("add", "a + b")

putIfAbsent("subtract", "a - b")

putIfAbsent("multiply", "a \* b")

putIfAbsent("divide", "a / b")

}

while (true) {

try {

println("Доступные операции: ${operations.entries.joinToString { "${it.key}: ${it.value}" }}")

println("Введите своё выражение или выберите одну из доступных операций:")

val operationName = readLine()!!

val operationExpression = operations[operationName] ?: run {

println("Введите своё выражение с переменными (например, pow(a, b)):")

val expression = readLine()!!

operations[operationName] = expression

expression

}

val variables = extractVariables(operationExpression)

println("Введите значения для переменных: ${variables.joinToString()}")

val values = mutableMapOf<String, Double>()

variables.forEach { variable ->

println("Введите значение для $variable:")

values[variable] = readLine()!!.toDouble()

}

val operation = createCustomOperation(operationExpression, variables)

val result = calculate(values, operation)

println("Результат: $result")

println("Хотите выполнить еще одну операцию? (да/нет)")

val answer = readLine()

if (answer?.lowercase() == "нет") {

break

}

} catch (e: Exception) {

println("Ошибка: ${e.message}")

continue

}

}

saveOperations(operations, "operations.txt")

println("Загруженные операции: ${operations.keys.joinToString()}")

}

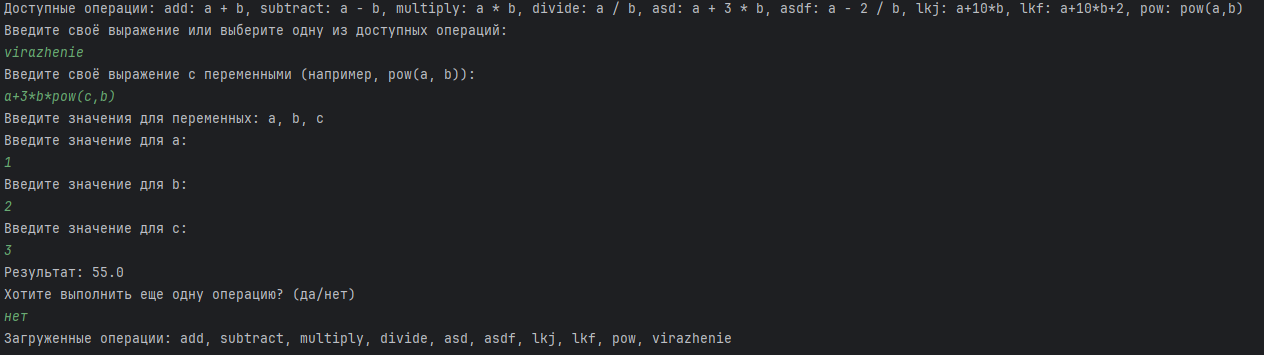


Рисунок 1 – Результат исполнения кода программы

**Вывод:** в ходе выполнения лабораторной работы была изучена концепция функционального программирования на языке Kotlin, включая такие элементы, как лямбда-выражения, анонимные функции и замыкания.