Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

**«Программирование мобильных информационных систем»**

**Лабораторная работа № 3**

«Функциональное программирование и лямбда-выражения»

Проверил: Выполнил:

Усенко Ф.В. Шамрило И.Г.

Минск 2024

**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.

Вариант № 8

Задание: Напишите программу, которая использует саморекурсивные лямбда-выражения для обхода и обработки бинарного дерева (например, подсчет листьев, поиск максимального значения).

data class TreeNode<T : Comparable<T>>(val value: T, val left: TreeNode<T>? = null, val right: TreeNode<T>? = null)

fun main() {

val tree = TreeNode(

10,

TreeNode(5, TreeNode(2), TreeNode(7)),

TreeNode(15, right = TreeNode(20))

)

val countLeaves: (TreeNode<\*>?) -> Int = run {

var countLeaves: (TreeNode<\*>?) -> Int = { \_ -> 0 }

countLeaves = { node ->

when {

node == null -> 0

node.left == null && node.right == null -> 1

else -> countLeaves(node.left) + countLeaves(node.right)

}

}

countLeaves

}

val findMax: (TreeNode<Int>?) -> Int? = run {

var findMax: (TreeNode<Int>?) -> Int? = { null }

findMax = { node ->

when {

node == null -> null

else -> {

val leftMax = findMax(node.left)

val rightMax = findMax(node.right)

maxOf(node.value, leftMax ?: node.value, rightMax ?: node.value)

}

}

}

findMax

}

println("Количество листьев: ${countLeaves(tree)}")

println("Максимальное значение: ${findMax(tree)}")

}

Вывод: в ходе работы ознакомился с основами функционального программирования в Kotlin, изучил лямбда-выражения, анонимные функции и замыкания. Научился использовать эти концепции для написания более гибкого и читаемого кода. Реализовал программу, которая использует саморекурсивные лямбда-выражения для обхода и обработки бинарного дерева.