МГТУ им. Н.Э. Баумана

Отчёт по лабораторной работе №2-3
по курсу «ПиКЯП»

Студент группы ИУ5-31Б
Крюков В. А.

2024 г.

Тема:

Работа с коллекциями, функциями высшего порядка, измерением времени выполнения операций, генерацией случайных данных и сортировкой элементов

Код Программы:
Main.kt

```
fun field(items: List<Map<String, Any?>>, vararg keys: String): String {
    val result = StringBuilder()  // Single result builder for everything

    // Loop through each map in the list
    var isFirst = true
    for (map in items) {
        if (keys.size == 1) {
            // Case when only one key is passed
            val key = keys[0]
            map[key]?.let {
                // Print and append only if value is not null
                if (!isFirst) result.append(", ")
                result.append("$it")
                isFirst = false
            }
        } else {
            // When multiple keys are passed, build filtered key-value
pairs
            val filtered = map.filter { (k, v) -> k in keys && v != null }
            if (filtered.isNotEmpty()) {
                for ((key, value) in filtered) {
                    if (!isFirst) result.append(", ")
                    result.append("$key: $value")
                    isFirst = false
                }
                result.append("\n")  // Add a newline after each map
                isFirst = true
            }
        }
    }

    return result.toString()
}

fun main() {
    val goods = listOf(
        mapOf("title" to "Rug", "price" to 2000, "color" to "green"),
        mapOf("title" to "Couch", "color" to "black")
    )

    println("Single key:")
    println(field(goods, "title"))  // Should print "Rug" and "Couch"

    println("\nMultiple keys:")
    println(field(goods, "title", "price"))  // Should print title and
price if available
}
```

cmTimer.kt

```
import kotlin.time.measureTime

class CmTimer : AutoCloseable{
    private val startTime = System.currentTimeMillis()
    override fun close() {
        val endTime = System.currentTimeMillis()
        print(endTime - startTime)
    }
}

fun main(){

    val timeTaken = measureTime {
        Thread.sleep(55)
    }
    println(timeTaken)
    CmTimer().use{
        Thread.sleep(55)
    }
}
```

genRandom.kt

```
fun genRandom(n: Int, start: Int, finish:Int):List<Int>{
    val randomNumbers = mutableListOf<Int>()
    repeat(n) {
        randomNumbers.add((start..finish).random())
    }
    return randomNumbers
}

fun main(){
    print(genRandom(5,1,3)) // должен выдать 5 случайных чисел в диапазоне
от 1 до 3
}
```

sort.kt

```
import kotlin.math.abs

fun main(){
    val data = listOf(4, -30, 100, -100, 123, 1, 0, -1, -4)

    val sortedWithLambda = data.sortedByDescending{ abs(it) }
    println("Sorted using Lambda Function:\n$sortedWithLambda")


    fun absoluteValue(x: Int): Int {
        return abs(x)
    }
    val sortedWithoutLambda = data.sortedByDescending(::absoluteValue)
    print("Sorted without Lambda Function:\n$sortedWithoutLambda")
}
```

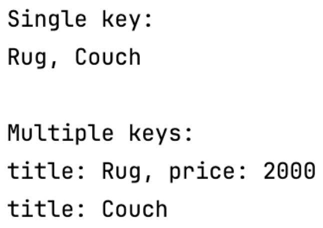
printResult.kt

```
fun test_1(): Int {
    return 1
}

fun test_2(): String {
    return "iu5"
}

fun test_3(): Map<String, Int> {
    return mapOf("a" to 1, "b" to 2)
}
```

```
fun test_4(): List<Int> {
    return listOf(1, 2)
}

fun <T> decorator(function: () -> T): T {
    val result = function()


    when (result) {
        is List<*> -> println(result)
        is Map<*, *> -> result.forEach { (key, value) -> println("$key =
$value") }
        else -> println(result)
    }
    return result
}


fun main() {
    println("test_1")
    decorator(::test_1)
    println("test_2")
    decorator(::test_2)
    println("test_3")
    decorator(::test_3)
    println("test_4")
    decorator(::test_4)
}
```

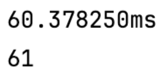
Результаты программы:

Main.kt



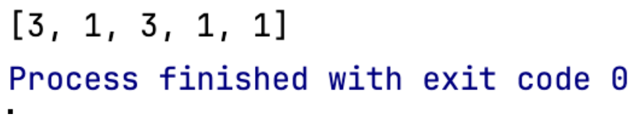
```
Single key:
Rug, Couch

Multiple keys:
title: Rug, price: 2000
title: Couch
```

cmTimer.kt



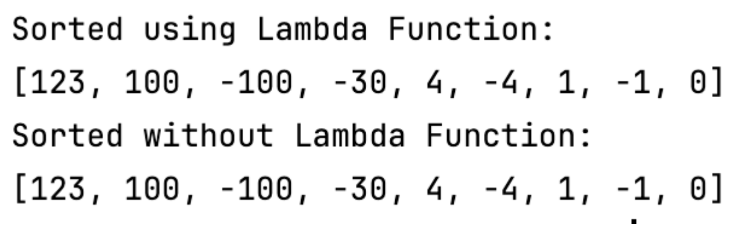
```
60.378250ms
61
```

genRandom.kt



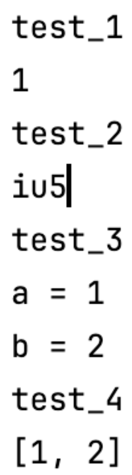
```
[3, 1, 3, 1, 1]
Process finished with exit code 0
```

Sort.kt



```
Sorted using Lambda Function:
[123, 100, -100, -30, 4, -4, 1, -1, 0]
Sorted without Lambda Function:
[123, 100, -100, -30, 4, -4, 1, -1, 0]
```

Decorator.kt



```
test_1
1
test_2
iu5
test_3
a = 1
b = 2
test_4
[1, 2]
```