Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра инженерной психологии и эргономики

Дисциплина: Программирование мобильных информационных систем

ОТЧЁТ по лабораторной работе №3 на тему:

**«Функциональное программирование и лямбда-выражения»**

Выполнил: Проверил:

ст. гр. 214301 Усенко Ф.В.

Приходько Н.Н.

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 3. Генерация и использование функций на лету:** Создайте программу, которая динамически генерирует функции на основе пользовательского ввода (например, определение функции для нахождения максимума из трех чисел) и затем использует их для обработки данных.

# КОД ПРОГРАММЫ

fun main() {  
 println("Выберите операцию:")  
 println("1. Максимум из трех чисел")  
 println("2. Сумма двух чисел")  
 println("3. Умножение двух чисел")  
  
 val choice = readLine()?.toIntOrNull()  
  
 val maxOrNull: (List<Int>) -> Int? = **{** numbers **->** if (numbers.isEmpty()) null  
 else {  
 var max = numbers[0]  
 for (num in numbers) {  
 if (num > max) max = num  
 }  
 max  
 }  
 **}** val sum: (List<Int>) -> Int = **{** numbers **->** var total = 0  
 for (num in numbers) total += num  
 total  
 **}** val reduce: (List<Int>, (Int, Int) -> Int) -> Int? = **{** numbers, operation **->** if (numbers.isEmpty()) null  
 else {  
 var result = numbers[0]  
 for (i in 1 until numbers.size) {  
 result = operation(result, numbers[i])  
 }  
 result  
 }  
 **}** val operation: ((List<Int>) -> Int)? = when (choice) {  
 1 -> **{** numbers **->** if (numbers.size == 3) maxOrNull(numbers) ?: 0 else {  
 println("Введите три числа.")  
 0  
 }  
 **}** 2 -> **{** numbers **->** if (numbers.size == 2) sum(numbers) else {  
 println("Введите два числа.")  
 0  
 }  
 **}** 3 -> **{** numbers **->** if (numbers.size == 2) reduce(numbers) **{** acc, i **->** acc \* i **}** ?: 0 else {  
 println("Введите два числа.")  
 0  
 }  
 **}** else -> {  
 println("Неверный выбор!")  
 null  
 }  
 }  
  
 if (operation != null) {  
 println("Введите числа через пробел:")  
 val input = readLine()?.split(" ")?.mapNotNull **{** it.toIntOrNull() **}** if (input != null) {  
 val result = operation(input)  
 println("Результат: $result")  
 }  
 else {  
 println("Некорректный ввод.")  
 }  
 }  
}

# ОТВЕТЫ НА КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Как объявить функцию в Kotlin? В чем разница между обычной функцией и однострочной функцией?

fun greet(name: String) = "Hello, $name!"

Разница:

Однострочная функция сокращает синтаксис за счёт автоматического определения возвращаемого типа (если он может быть выведен).

В обычной функции требуется явно использовать return, если есть тело блока { }.

1. Что такое функция высшего порядка, и как её использовать? Приведите пример.

Функция высшего порядка принимает другую функцию в качестве аргумента или возвращает функцию как результат.

**Пример:**

fun operateOnNumbers(a: Int, b: Int, operation: (Int, Int) -> Int): Int {

return operation(a, b)

}

fun add(x: Int, y: Int) = x + y

fun main() {

val result = operateOnNumbers(5, 3, ::add)

println(result) // 8

}

1. Как передать функцию в качестве параметра другой функции? Приведите пример.

Можно передать функцию в качестве параметра, используя ссылку на неё (::) или лямбду.

**Пример:**

fun multiply(a: Int, b: Int) = a \* b

fun calculate(a: Int, b: Int, operation: (Int, Int) -> Int): Int {

return operation(a, b)

}

fun main() {

val result = calculate(4, 6, ::multiply) // Передаём функцию multiply

println(result) // 24

}

1. Чем отличается лямбда-выражение от обычной функции?

**Различия:**

Лямбда-выражение — это анонимная функция, которая не имеет имени.

Лямбда-выражения обычно используются для краткости при передаче функций как параметров.

Лямбда-выражение имеет более компактный синтаксис: параметры указываются перед ->.

1. Как создать лямбда-выражение с двумя параметрами? Приведите пример.

val calculate = { x: Int, y: Int -> x \* y }

fun main() {

val result = calculate(7, 8)

println(result) // 56

}

# ВЫВОД

В ходы выполнения лабораторной работы было проведено ознакомление с основами функционального программирования в Kotlin, изучены лямбда-выражения, анонимные функции и замыкания.