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 №1

на тему

**Основы языка Kotlin**

Выполнил: Проверил:

ст. гр. 214302 Усенко Ф.В.

Коршов М.И.

Минск 2024

Вариант 15.Форматирование и разбор текстовых документов: Создайте программу, которая принимает текстовый файл, разбивает его на предложения, преобразует каждое предложение в новый формат (например, делает первые буквы всех слов заглавными) и сохраняет результат в новый файл.

Листинг кода:

import java.io.File

import java.io.FileNotFoundException

import java.io.IOException

import java.util.\*

fun main() {

val fileText = readFile("file.txt")

if (fileText.isBlank()) {

println("File is blank")

return

}

val fileSentences = fileText.trim().split(".").toMutableList()

fileSentences.forEach { sentence ->

val sentenceWords = sentence.split(" ").toMutableList()

sentenceWords.forEach { word ->

sentenceWords[sentenceWords.indexOf(word)] = word.replaceFirstChar { if (it.isLowerCase()) it.titlecase(Locale.getDefault()) else it.toString() }

}

fileSentences[fileSentences.indexOf(sentence)] = sentenceWords.joinToString(" ")

}

val resultString = fileSentences.joinToString(".")

try {

File("new-file.txt").writeText(resultString)

println("New refactored file was successfully created!")

} catch (e: IOException) {

println(e.message)

}

}

fun readFile(fileName: String): String {

try {

val file = File(fileName)

println("File was successfully read!")

return file.readText()

} catch (e: FileNotFoundException) {

println(e.message)

return ""

}

}

Контрольные вопросы:

6. Как в *Kotlin* можно обработать возможность появления null?

В *Kotlin* по умолчанию переменные не могут принимать значение *null*. Однако если необходимо, можно указать, что переменная может быть *null*, добавив знак вопроса после типа:

var name: String? = null

7. Как организован ввод данных с консоли в *Kotlin*?

Ввод данных от пользователя в *Kotlin* осуществляется с помощью функции *readLine()*, которая возвращает введенную строку. Для преобразования строки в число используются методы *toInt()*, *toDouble()*, и т.д:

println("Enter your age:")

val age = readLine()?.toIntOrNull()

if (age != null) {

println("Your age is $age")

} else {

println("Invalid input")

}

Здесь *toIntOrNull()* возвращает *null*, если ввод не является числом.

8. Чем отличаются функции *print()* и *println()*?

Функция *println()* добавляет перевод строки после вывода текста, тогда как *print()* этого не делает.

15. Можно ли передать функцию в качестве аргумента другой функции? Приведите пример.

В *Kotlin* функции могут передаваться как параметры другим функциям:

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

return operation(a, b)

}

fun main() {

val sum = operate(5, 3) { x, y -> x + y }

println(sum) // Выводит 8

}