**Министерство науки и высшего образования Российской Федерации**

Федеральное государственное автономное образовательное

учреждение высшего образования

**Национальный исследовательский**

**Томский политехнический университет**

Инженерная школа информационных технологий и робототехники

Отделение информационных технологий

Отчет по лабораторной работе №1 по дисциплине

**«Язык Kotlin и основы разработки»**

Базовые операции и функции

****

Выполнил:

Студент группы 1А22 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.К. Кравцов

Проверил:

Ст. преп. ОИТ ИШИТР \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.А. Дорофеев

Томск 2024

# Задание

Напишите приложение, которое на входе получает строку с текстом, обрабатывает её в соответствии с требованиями, приведёнными ниже, и выводит исходную и обработанную строку в консоль. В строке должны быть произведены следующие изменения:

* Кавычки (") должны быть заменены на лапки («»). Если кавычка стоит перед словом, то она заменяется на открывающую лапку, а если после слова, то на закрывающую. Следует учитывать знаки препинания: после слова может стоять точка, запятая, скобка и т.д. Кавычка может также стоять в начале или конце строки. Пример: Я живу в гостинице "Звездный путь". → Я живу в гостинице «Звездный путь».
* Пробелы перед и после знаков препинания должны быть приведены к следующему виду: НЕ ставится пробел перед точкой, запятой, открывающей скобки, после закрывающей скобкой и т.д. Ставится пробел после точки, запятой, закрывающей скобки, перед открывающей скобкой и т.д. Пример: Я( да и ты ) , в целом , молодцы ! → Я (да и ты), в целом, молодцы!
* Тире – должно с обеих сторон окружаться пробелами, если, конечно, слева и справа не стоят буквы. Пример: Ты-то молодец -хвалю! → Ты-то молодец – хвалю!
* Двойные пробелы должны быть заменены на одинарные.

Пример текста, который можно использовать для проверки правильности работы

программы:

" Лето , как обычно ,пролетело незаметно..."-грустно сказал Ваня .Он( и его друзья

)сидели на берегу речки с поэтичным названием "Стремительная ".

После обработки должен получиться следующий текст:

«Лето, как обычно, пролетело незаметно...» - грустно сказал Ваня. Он (и его друзья)

сидели на берегу речки с поэтичным названием «Стремительная».

# Текст программы

fun main() {  
 val inputText = "\" Лето , как обычно ,пролетело незаметно...\"–грустно сказал Ваня .Он( и его друзья )сидели на берегу речки с поэтичным названием \"Стремительная \"."  
 println(inputText)  
 println("-".repeat(inputText.length))  
 println(processText(inputText))  
}  
  
fun processText(input: String): String {  
 var text = replaceQuotes(input)  
 text = removeExtraSpaces(text)  
 text = replaceHyphensWithDashes(text)  
 text = addMissingSpaces(text)  
 return text  
}  
  
fun replaceQuotes(input: String): String {  
 var insideQuotes = false  
 return input.map { char ->  
 when (char) {  
 '"' -> {  
 insideQuotes = !insideQuotes  
 if (insideQuotes) '«' else '»'  
 }  
 else -> char  
 }  
 }.joinToString("")  
}  
  
fun removeExtraSpaces(input: String): String {  
 val regex1 = "\\s+([,.)»])".toRegex() // 1 или более пробелов перед , или . или ) или »  
 val regex2 = "([(«])\\s+".toRegex() // 1 или более пробелов после ( или «  
 return input  
 .replace(regex1, "$1")  
 .replace(regex2, "$1")  
 .replace("\\s+".toRegex(), " ") // убираем множественные пробелы  
}  
  
fun addMissingSpaces(input: String): String {  
 val regex1 = "(?<=\\p{L})([,.)])(?![\\s\\.])".toRegex()  
 val regex2 = "(?<=\\S)[(]".toRegex()   
 var outputText: String  
 outputText = input.replace(regex1) {  
 "${it.value} " // Добавляем пробел после них  
 }  
 outputText = outputText.replace(regex2) {  
 " ${it.value}" // Добавляем пробел перед "("  
 }  
 return outputText  
}  
  
fun replaceHyphensWithDashes(input: String): String {  
 return input.replace("(\\s\*[-–]\\s\*)".toRegex(), " – ")  
 .replace("\\s+".toRegex(), " ") // Избавляемся от лишних пробелов  
 .trim() // Убираем пробелы по краям строки  
}

# Результаты работы

# 

# Выводы

В процессе выполнения лабораторной работы были изучены фундаментальные аспекты языка Kotlin, включая оператор выбора when, строковые методы .replace() и .trim(). Кроме того, было углублено понимание и практика написания регулярных выражений.

В результате было разработано приложение на Kotlin, отвечающее поставленным требованиям. Программа эффективно обрабатывает текст, выполняя следующие операции:

* Замена кавычек
* Корректировка пробелов вокруг знаков препинания
* Добавление пробелов к тире
* Удаление лишних пробелов

Полученные результаты соответствуют заданным критериям, демонстрируя успешное использование языка Kotlin и методов обработки текста.

# Ссылка на код для проверки

<https://pl.kotl.in/7_X4Iv4eo>