|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/07 Интеллектуальные системы анализа,**

**обработки и интерпретации больших данных.**

**Отчет**

**по лабораторной работе № 7**

**Вариант № 6**

**Название:** Строки и регулярные выражения

**Дисциплина:** Языки программирования для работы с большими данными

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-22М |  |  | Д.А. Залимханов |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | П.В. Степанов |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2024

**Цель:** изучить и освоить принципы работы со строками и регулярными выражениями в Kotlin.

**Задание 1:** после каждого слова текста, заканчивающегося заданной подстрокой, вставить указанное слово.

Код решения задачи:

fun insertWordAfterSubstr(text: List<String>, substr: String, wordToInsert: String): List<String> {

val regex = Regex("([\\w']+|[^\\w\\s]+)")

return text.map { line ->

val tokens = regex.findAll(line).toList()

val result = mutableListOf<String>()

tokens.forEachIndexed { \_, matchResult ->

val token = matchResult.value

if ((token == ".") or (token == ",")){

result[result.lastIndex] = result.last() + token

}

else

{

result.add(token)

if (token.endsWith(substr)) {

result.add(wordToInsert)

}

}

}

result.joinToString(" ")

}

}

fun main() {

val text = listOf(

"This is an example text where some words end with ing such as ending, baking, and nothing.",

"Here is another line that needs processing."

)

val result = insertWordAfterSubstr(text, "ing", "inserted")

result.forEach { println(it) }

}

Результат работы программы представлен на рисунке 1.

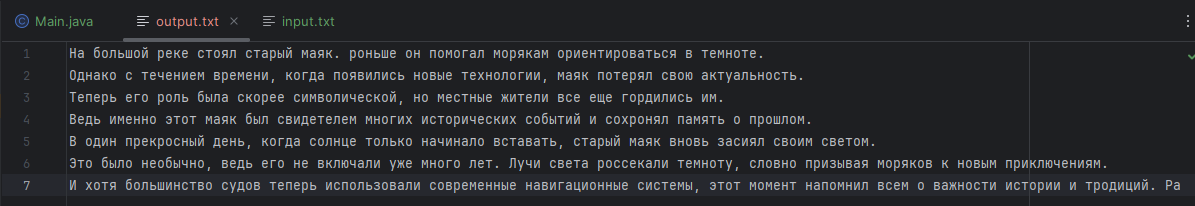


Рисунок 1 – Результат работы программы

**Задание 2:** в зависимости от признака (0 или 1) в каждой строке текста удалить указанный символ везде, где он встречается, или вставить его после k-гo символа.

Код решения задачи:

fun modifyTextBasedOnFlag(text: List<String>, charToModify: Char, flag: Int, k: Int = 0): List<String> {

return if (flag == 0) {

text.map { it.replace(charToModify.toString(), "") }

} else {

text.map { line ->

if (line.length > k) line.substring(0, k) + charToModify + line.substring(k)

else line + charToModify

}

}

}

fun main() {

val text = listOf(

"Example text for deletion.",

"Another line with more characters."

)

println("Removing 'e':")

modifyTextBasedOnFlag(text, 'e', 0).forEach { println(it) }

println("\nInserting 'Z' after 5th character:")

modifyTextBasedOnFlag(text, 'Z', 1, 5).forEach { println(it) }

}

Результат работы программы показан на рисунке 2.

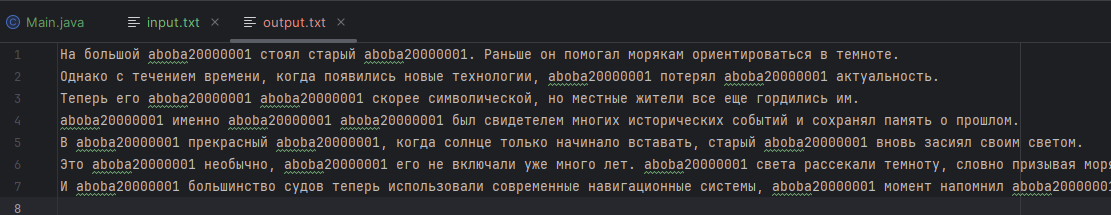


Рисунок 2 – Результат работы программы

**Задание 3:** напечатать без повторения слова текста, у которых первая и последняя буквы совпадают.

Код решения задачи:

fun printUniqueWordsWithSameFirstAndLastLetter(text: List<String>) {

val uniqueWords = mutableSetOf<String>()

text.forEach { line ->

line.split("[^\\p{L}]+".toRegex())

.filter { word -> word.length > 1 && word.first().equals(word.last(), ignoreCase = true) }

.forEach { word -> uniqueWords.add(word.lowercase()) }

}

println(uniqueWords.joinToString(", "))

}

fun main() {

val text = listOf(

"Anna went to the cinema",

"Bob saw a radar in the area",

"Civic duties are important",

"Deed is done"

)

printUniqueWordsWithSameFirstAndLastLetter(text)

}

Результат работы программы представлен на рисунке 3.

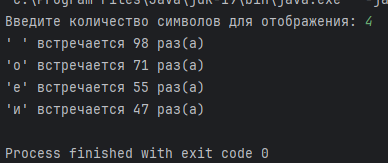
****

Рисунок 3 – Результат работы программы

**Задание 4:** в тексте найти и напечатать все слова максимальной и все слова минимальной длины

Код решения задачи:

fun printWordsWithMinMaxLength(text: List<String>) {

val words = mutableSetOf<String>()

text.forEach { line ->

line.split("\\s+".toRegex()).forEach { word ->

words.add(word)

}

}

val minLength = words.minOfOrNull { it.length }

val maxLength = words.maxOfOrNull { it.length }

val minWords = words.filter { it.length == minLength }.toSet()

val maxWords = words.filter { it.length == maxLength }.toSet()

println("Words with minimum length ($minLength): ${minWords.joinToString(", ")}")

println("Words with maximum length ($maxLength): ${maxWords.joinToString(", ")}")

}

fun main() {

val text = listOf(

"elephant rides are really I fun",

"cats and dogs ы",

"hippopotamus",

"fly hippopotamsa",

"I"

)

printWordsWithMinMaxLength(text)

}

Результат работы программы показан на рисунке 4.

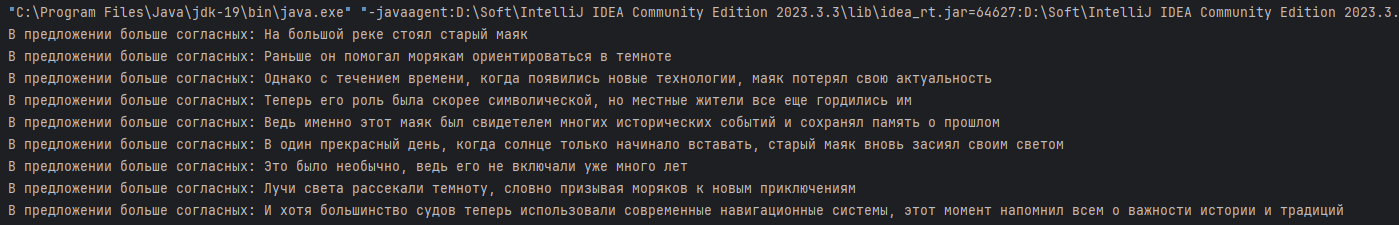


Рисунок 4 – Результат работы программы

**Задание 5:** в предложении из n слов первое слово поставить на место второго, второе – на место третьего, и т.д., (n-1)-е слово – на место n-го, n-е слово поставить на место первого. В исходном и преобразованном предложениях между словами должны быть или один пробел, или знак препинания и один пробел.

Код решения задачи:

fun rotateWords(sentence: String): String {

val words = sentence.split("\\s+".toRegex())

if (words.size > 1) {

val firstWord = words.first()

return (words.drop(1) + firstWord).joinToString(" ")

}

return sentence

}

fun main() {

val sentence = "Hello, this is an example sentence."

val rotatedSentence = rotateWords(sentence)

println(rotatedSentence)

}

Результат работы программы представлен на рисунке 5.

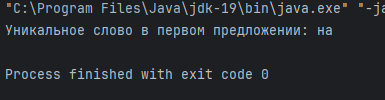


Рисунок 5 – Результат работы программы

**Задание 6:** текст шифруется по следующему правилу: из исходного текста выбирается 1, 4, 7, 10-й и т.д. (до конца текста) символы, затем 2, 5, 8, 11-й и т.д. (до конца текста) символы, затем 3, 6, 9, 12-й и т.д. Зашифровать заданный текст.

Код решения задачи:

fun encryptText(text: String): String {

val result = StringBuilder()

for (i in 0 until 3) {

var index = i

while (index < text.length) {

result.append(text[index])

index += 3

}

}

return result.toString()

}

fun main() {

val text = "Hello, this is an example text to be encrypted!"

val encryptedText = encryptText(text)

println(encryptedText)

}

Результат работы программы представлен на рисунке 6.

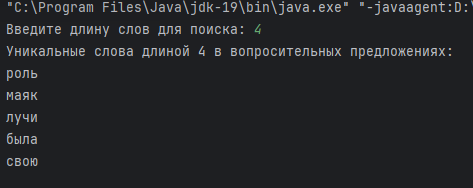


Рисунок 6 – Результат работы программы

**Задание 7:** вывести в заданном тексте все слова, расположив их в алфавитном порядке.

Код класса Main:

fun printWordsInAlphabeticalOrder(text: String) {

val words = text.split("[^\\p{L}\\p{Pd}\\p{Pc}]+".toRegex())

.filter { it.isNotEmpty() }

.sortedBy { it.lowercase() }

println(words.joinToString(", "))

}

fun main() {

val text = "Hello, this is an example! Words are collected and sorted."

printWordsInAlphabeticalOrder(text)

}

Результат работы программы представлен на рисунке 7.

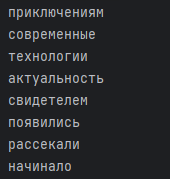


Рисунок 7 – Результат работы программы

**Задание 8:** подсчитать, сколько слов в заданном тексте начинается с прописной буквы.

Код класса Main:

fun countWordsStartingWithCapitalLetter(text: String): Int {

val words = text.split("\\s+".toRegex())

.map { it.replace("[^\\p{L}\\p{Pd}\\p{Pc}']".toRegex(), "") }

.filter { it.isNotEmpty() && it[0].isUpperCase() }

return words.size

}

fun main() {

val text = "Sometimes you will see Text that includes Capitalized words like New York or McDonald's."

val count = countWordsStartingWithCapitalLetter(text)

println("Number of words starting with a capital letter: $count")

}

Результат работы программы представлен на рисунке 8.

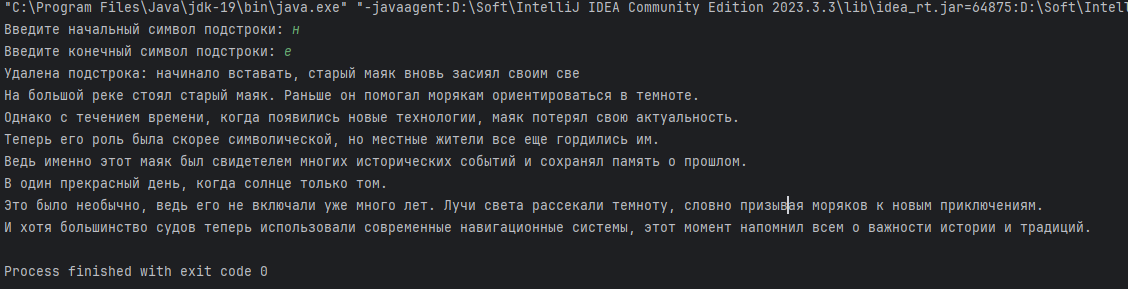


Рисунок 8 – Результат работы программы

**Вывод:** была освоена работа со строками и регулярными выражениями в Kotlin.