Федеральное государственное автономное

образовательное учреждение

Высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

|  |
| --- |
| Институт космических и информационных технологий |
| институт |
| Программная инженерия |
| кафедра |

**ОТЧЕТ О ПРАКТИЧЕСКОЙ РАБОТЕ**

|  |
| --- |
| Алгебраические типы данных |
| тема |

Вариант 6(24)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Преподаватель | |  |  |  | К. В. Богданов |
|  | |  | подпись, дата |  | инициалы, фамилия |
| Студент | КИ21-17/1Б, 032156940 |  |  |  | Н. А. Самарин |
|  | номер группы, зачётной книжки |  | подпись, дата |  | инициалы, фамилия |

Красноярск 2024

**СОДЕРЖАНИЕ**

1 Задание............................................................................................................... 3

2 Исходный код разработанного алгоритма..................................................... 3

3 Результат........................................................................................................... 4

**1 Задание**

Написать скрипт, реализующий задание (см. ниже, по вариантам) и  
демонстрирующий работоспособность функций. Тестовые (демонстрационные)  
данные могут находиться в самом скрипте. Можно использовать как Either и  
Option, так и Sealed traits + case objects по необходимости

Вариант 6(24): Определим следующий набор операций над строками:  
очистка (удаление всех символов из строк), удаление (удаление всех вхождений  
указанного символа), замена (замена всех вхождений одного символа на  
другой), добавление (добавление в начало строки указанного символа).  
Разработайте тип данных, характеризующий операции над строками.  
Определите основные функции: a. process - получает в качестве аргумента  
действие и строку, возвращает строку, модифицированную в соответствие с  
указанным действием b. processAll - аналогично предыдущей, но получает  
список действий и выполняет их по порядку c. deleteAll - принимает две строки,  
и удаляет из второй строки все символы первой (использовать при реализации  
processAll)

**2 Исходный код разработанного алгоритма**

Листинг 1 – Исходный код разработанного алгоритма

object Main {  
 def main(args: Array[String]): Unit = {  
 // Демонстрация работы функций  
 val inputString = "Hello, World!"  
 println("inputString = \"Hello, World!\"")  
 println("process(Clean, inputString):")  
 val cleanString = process(Clean, inputString)  
 println(cleanString) // Output: ""  
 println("process(Delete('l'), inputString):")  
 val deleteString = process(Delete('l'), inputString)  
 println(deleteString) // Output: "Heo, Word!"  
 println("process(Replace('o', '0'), inputString):")  
 val replaceString = process(Replace('o', '0'), inputString)  
 println(replaceString) // Output: "Hell0, W0rld!"  
 println("process(Add('X'), inputString):")  
 val addString = process(Add('X'), inputString)  
 println(addString) // Output: "XHello, World!"  
 println("operations = List(Delete('o'), Replace('l', 'L'), Add('X'))")  
 println("processAll(operations, inputString):")  
 val operations = List(Delete('o'), Replace('l', 'L'), Add('X'))  
 val processedString = processAll(operations, inputString)  
 println(processedString) // Output: "HeLLX, W0rld!"  
 println("deleteAll(inputString, \"lo\"):")  
 val deletedString = deleteAll(inputString, "lo")  
 println(deletedString) // Output: "He, Wrd!"  
 }  
 sealed trait StringOperation  
 case object Clean extends StringOperation  
 case class Delete(char: Char) extends StringOperation  
 case class Replace(oldChar: Char, newChar: Char) extends StringOperation  
 case class Add(char: Char) extends StringOperation  
 /\*\*

Окончание листинга 1

\* Выполнение операции над строкой  
 \* @param operation выполняемая операция  
 \* @param input строка для выполнения операции  
 \* @return строка после выполнения операции  
 \*/  
 def process(operation: StringOperation, input: String): String = {  
 operation match {  
 case Clean => input.replaceAll(".", "")  
 case Delete(char) => input.filterNot(\_ == char)  
 case Replace(oldChar, newChar) => input.map(c => if (c == oldChar) newChar  
 else c)  
 case Add(char) => char + input  
 }  
 }  
 /\*\*  
 \* Выполнение списка операций над строкой  
 \* @param operations список выполняемых операций  
 \* @param input строка для выполнения операций  
 \* @return строка после выполнения операций  
 \*/  
 def processAll(operations: List[StringOperation], input: String): String = {  
 operations.foldLeft(input)((acc, op) => process(op, acc))  
 }  
 /\*\*  
 \* Удаление из первой строки все символы второй  
 \* @param deleteFrom строка из которой удаляются символы  
 \* @param deleteChars строка, символы которой удаляются  
 \* @return первая строка без символов из второй  
 \*/  
 def deleteAll(deleteFrom: String, deleteChars: String): String = {  
 val deleteList = deleteChars.map(c => Delete(c)).toList  
 processAll(deleteList ,deleteFrom)  
 }  
}

**3 Результат**

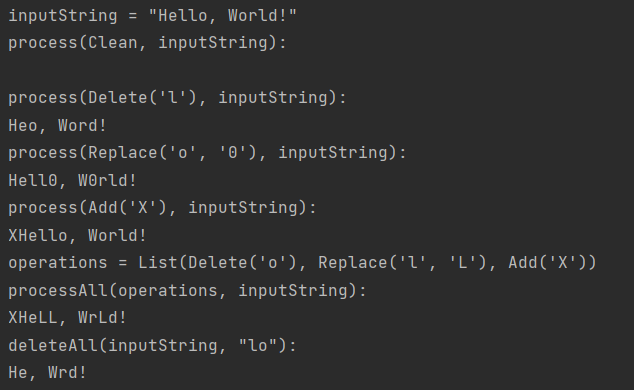


Рисунок 1 – Результаты работы алгоритма