**Федеральное агентство связи**

**Ордена Трудового Красного Знамени**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский технический университет связи и информатики»**

Кафедра Информатики

****

**Отчет по лабораторной работе № 4**

по дисциплине «СиАОД»

на тему:

«**Реализация стека/дека**»

Выполнила: студентка группы БВТ1802

Лаврухина Елена Павловна

Руководитель:

Кутейников Иван Алексеевич

Москва 2020

# **Цель работы**

Используя технологию модульного программирования разработать программу обработки данных, содержащихся в заранее подготовленном файле, в соответствии с индивидуальным заданием. Применить динамическую структуру указанного в задании вида: стек, очередь или дек. Программа должна включать модуль, содержащий набор всех необходимых средств (типов, подпрограмм и т.д.) для решения поставленной задачи.

Вариант № 19. В текстовом файле хранится выражение, записанное в инфиксной форме. Используя стек, перевести его в постфиксную форму и в таком виде записать в новый текстовый файл. Пример выражения: **a+b/c/d\*e=>abc/d/e\*+**

Выполнение

Код программы

import java.io.\*;  
class Lab4 {  
 private int Size;  
 private char[] Array;  
 private int ind;  
  
 public Lab4(int n) {  
 this.Size = n;  
 this.Array = new char[Size];  
 this.ind = -1;  
 }  
 public void addElement(char element) {  
 Array[++ind] = element;  
 }  
 public char deleteElement() {  
 return Array[ind--];  
 }  
 public boolean isEmpty() {  
 return (ind == -1);  
 }  
 public boolean isFull() {  
 return (ind == Size -1);  
 }  
 public char getTop() {  
 return Array[ind];  
 }  
}  
 public class Main {  
 public static String Handler(Lab4 Stack) {  
 String l = *getRev*(Stack);  
 Lab4 Stack1 = new Lab4(l.length());  
 Lab4 Stack2 = new Lab4(l.length());  
 boolean error = false;  
 int i = 0;  
 while (!Stack2.isFull()) {  
 Stack2.addElement(l.charAt(i));  
 i++;  
 }  
 if ((Stack2.isEmpty())||(!*Ref*(Stack2))) {  
 error = true;  
 }  
 else {  
 boolean log = false;  
 char rem = 0;  
 while (!Stack2.isEmpty()) {  
 if ((Stack2.getTop() != 43) && (Stack2.getTop() != 45)) {  
 if ((Stack2.getTop() != 42) && (Stack2.getTop() != 47)) {  
 Stack1.addElement(Stack2.getTop());  
 Stack2.deleteElement();  
 }  
 else {  
 char c = Stack2.getTop();  
 Stack2.deleteElement();  
 Stack1.addElement(Stack2.getTop());  
 Stack2.deleteElement();  
 Stack1.addElement(c);  
 }  
 }  
 else {  
 if(!log) {  
 log = true;  
 rem = Stack2.getTop();  
 Stack2.deleteElement();  
 }  
 else {  
 Stack1.addElement(rem);  
 rem = Stack2.getTop();  
 Stack2.deleteElement();  
 }  
 }  
 }  
 if (rem != 0)  
 Stack1.addElement(rem);  
 }  
 return error ? "Некорректные данные" : *getRev*(Stack1);  
 }  
 public static boolean Ref(Lab4 rStack) {  
 String line = *get*(rStack);  
 boolean error = true;  
 for (int i = 0;(i < line.length()) && (error); i++) {  
 if ((i == 0)||(i == line.length()-1)) {  
 switch (line.charAt(i)) {  
 case '+':  
 case '-':  
 case '\*':  
 case '/':  
 error = false;  
 break;  
 default:  
 if (!Character.*isLetter*(line.charAt(i)))  
 error = false;  
 break;  
 }  
 }  
 else {  
 switch (line.charAt(i)) {  
 case '+':  
 case '-':  
 case '\*':  
 case '/':  
 if ((!Character.*isLetter*(line.charAt(i-1))) || (!Character.*isLetter*(line.charAt(i+1))))  
 error = false;  
 break;  
 default:  
 if (!Character.*isLetter*(line.charAt(i)))  
 error = false;  
 else {  
 if ((line.charAt(i - 1) != '+') && (line.charAt(i - 1) != '-') &&  
 (line.charAt(i - 1) != '\*') && (line.charAt(i - 1) != '/'))  
 error = false;  
 if ((line.charAt(i + 1) != '+') && (line.charAt(i + 1) != '-') &&  
 (line.charAt(i + 1) != '\*') && (line.charAt(i + 1) != '/'))  
 error = false;  
 }  
 break;  
 }  
 }  
 }  
 return error;  
 }  
 public static String get(Lab4 gStack) {  
 String res = "";  
 while (!gStack.isEmpty()) {  
 res += gStack.getTop();  
 gStack.deleteElement();  
 }  
 for (int i = res.length()-1; i >- 1; i--) {  
 gStack.addElement(res.charAt(i));  
 }  
 return res;  
 }  
 public static String getRev(Lab4 grStack) {  
 String tempRes = "";  
 while (!grStack.isEmpty()) {  
 tempRes += grStack.getTop();  
 grStack.deleteElement();  
 }  
 for (int i = tempRes.length()-1; i > -1; i--) {  
 grStack.addElement(tempRes.charAt(i));  
 }  
 String res = "";  
 for (int i = tempRes.length()-1; i > -1; i--)  
 res += tempRes.charAt(i);  
 return res;  
 }  
 public static Lab4 File() {  
 String Str = "";  
 try (FileReader reader = new FileReader("C:\\Users\\Timelord\\Desktop\\dz\\2 k\\siaod\\lab4\\lab4.txt")) {  
 int r;  
 while ((r = reader.read()) != -1) {  
 Str += ((char)r);  
 }  
 }  
 catch(IOException ex) {  
 System.*out*.println(ex.getMessage());  
 }  
 Lab4 temp = new Lab4(Str.length()+4);  
 for (int j = Str.length()-1; j >= 0; j--) {  
 temp.addElement(Str.charAt(j));  
 }  
 System.*out*.print("Исходная строка: ");  
 System.*out*.println(*get*(temp));  
 return temp;  
 }  
 public static void Save(Lab4 sStack, String sStack1) {  
 String text = *get*(sStack) + "=>" + sStack1 + "\n";  
 try(FileOutputStream fos = new FileOutputStream("C:\\Users\\Timelord\\Desktop\\dz\\2 k\\siaod\\lab4\\lab4out.txt", true)) {  
 PrintWriter pw = new PrintWriter("lab4out.txt");  
 pw.close();  
 byte[] buf = text.getBytes();  
 fos.write(buf, 0, buf.length);  
 }  
 catch(IOException ex) {  
 System.*out*.println(ex.getMessage());  
 }  
 }  
 public static void main(String[] args) {  
 Lab4 Stack;  
 Stack = *File*();  
 System.*out*.println("Результат: " + *Handler*(Stack) + "\n");  
 *Save*(Stack, *Handler*(Stack));  
 }  
 }

Скриншоты работы программы

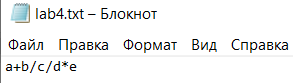


Рисунок 1 – Файл для считывания

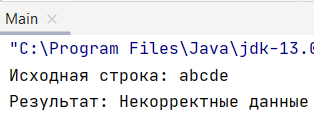


Рисунок 2 – Некорректные данные

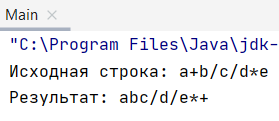


Рисунок 3 – Работа программы

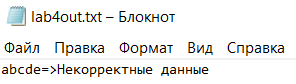


Рисунок 4 – Сохранение в файл часть 1

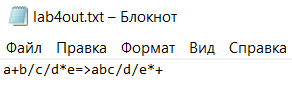


Рисунок 5 – Сохранение в файл часть 2

# **Вывод**

В ходе работы мы ознакомились с созданием и работой динамической структуры – стеком, а также научились считывать информацию из файла и записывать ее в файл.