Федеральное агенство связи
Ордена Трудового Красного Знамени федеральное государственное
бюджетное учреждение высшего образования
«Московский технический университет связи и информатики»

Кафедра Математической кибернетики и
информационных технологий

Лабораторная работа №4

по дисциплине: «Реализация стека/дека.»

Выполнил студент
группы БФИ1902
Рахимов Е.К.
Проверила:
Мосева М.С.

Москва, 2021 г.

Оглавление

1. Цель лабораторной работы

Цель данной лабораторной работы — изучить стек и дек и реализовать задачи.

2. Задание на лабораторную работу

1. Отсортировать строки файла, содержащие названия книг, в алфавитном порядке с использованием двух ***деков.***

2. ***Дек*** содержит последовательность символов для шифровки сообщений. Дан текстовый файл, содержащий зашифрованное сообщение. Пользуясь ***деком***, расшифровать текст. Известно, что при шифровке каждый символ сообщения заменялся следующим за ним в ***деке*** по часовой стрелке через один.

3. Даны три стержня и *n* дисков различного размера. Диски можно надевать на стержни, образуя из них башни. Перенести *n* дисков со стержня *A* на стержень *C*, сохранив их первоначальный порядок. При переносе дисков необходимо соблюдать следующие правила:
   - на каждом шаге со стержня на стержень переносить только один диск;
   - диск нельзя помещать на диск меньшего размера;
   - для промежуточного хранения можно использовать стержень *B*.

   Реализовать алгоритм, используя три ***стека*** вместо стержней *A*, *B*, *C*. Информация о дисках хранится в исходном файле.

4. Дан текстовый файл с программой на алгоритмическом языке. За один просмотр файла проверить баланс круглых скобок в тексте, используя ***стек.***

5. Дан текстовый файл с программой на алгоритмическом языке. За один просмотр файла проверить баланс квадратных скобок в тексте, используя ***дек.***

6. Дан файл из символов. Используя ***стек***, за один просмотр файла напечатать сначала все цифры, затем все буквы, и, наконец, все остальные символы, сохраняя исходный порядок в каждой группе символов.

7. Дан файл из целых чисел. Используя ***дек***, за один просмотр файла напечатать сначала все отрицательные числа, затем все положительные числа, сохраняя исходный порядок в каждой группе.

8. Дан текстовый файл. Используя ***стек***, сформировать новый текстовый файл, содержащий строки исходного файла, записанные в обратном порядке: первая строка становится последней, вторая – предпоследней и т.д.

9. Дан текстовый файл. Используя ***стек***, вычислить значение логического выражения, записанного в текстовом файле в следующей форме:
< ЛВ > ::= **T** | **F** | (**N**<ЛВ>) | (<ЛВ>**A**<ЛВ>) | (<ЛВ>**X**<ЛВ>) | (<ЛВ>**O**<ЛВ>),
где буквами обозначены логические константы и операции:
T – True, **F** – False, **N** – Not, **A** – And, **X** – Xor, **O** – Or.

10. Дан текстовый файл. В текстовом файле записана формула следующего вида:
<Формула> ::= <Цифра> | **M**(<Формула>,<Формула>) | **N**(Формула>,<Формула>)
< Цифра > ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
где буквами обозначены функции:
M – определение максимума, **N** – определение минимума.
Используя ***стек***, вычислить значение заданного выражения.

11. Дан текстовый файл. Используя ***стек***, проверить, является ли содержимое текстового файла правильной записью формулы вида:
< Формула > ::= < Терм > | < Терм > + < Формула > | < Терм > - < Формула >
< Терм > ::= < Имя > | (< Формула >)
< Имя > ::= **x** | **y** | **z**

3. Ход лабораторной работы

3.1 Листинг программы

```
package com.company.Lab4;

import java.util.Stack;

public class Labbb {
    public static Stack<String> slov= new Stack<>();
    public static int first=-1;
    public static int second=-1;
    public static int top=-1;
    public static void zapoln(String [] s) {
```

```
        for (int i = 0; i < s.length; i++) {
            slov.push(s[i]);
        }
    }

    public static void start() {
        String d;
        int l=slov.size();
        for (int i = 0; i <l; i++) {
            d=slov.pop();
            if(first==-
1&&(d.equals("0")||d.equals("1")||d.equals("2")||d.equals("3")||d.equals("4")||d
.equals("5")||d.equals("6")||d.equals("7")||d.equals("8")||d.equals("9")))
            {
                first=Integer.parseInt(d);
             //   System.out.println("firs="+first);
                continue;
            }
            if(second==-
1&&(d.equals("0")||d.equals("1")||d.equals("2")||d.equals("3")||d.equals("4")||d
.equals("5")||d.equals("6")||d.equals("7")||d.equals("8")||d.equals("9")))
            {
                second=Integer.parseInt(d);

               // System.out.println("second="+second);
                continue;
            }
            if(d.equals("m")&&first!=-1&&second!=-1){
                top=Math.min(first,second);

              //  System.out.println("firs="+first+"second="+second
+"top="+top+"m");
                first=-1;
                second=-1;
                continue;
            }
            if(d.equals("m")&&first!=-1){
                top=Math.min(first,top);
                //System.out.println("firs="+first+"top="+top+"M");
                first=-1;

                continue;
            }

            if(d.equals("M")&&first!=-1&&second!=-1){
                top=Math.max(first,second);
               // System.out.println("firs="+first+"second="+second
+"top="+top+"M");
                first=-1;
                second=-1;
                continue;
            }
            if(d.equals("M")&&first!=-1){
                top=Math.max(first,top);
                first=-1;
                //System.out.println("firs="+first +"top="+top+"M");
```

```
            }
        }
    }
    public static void main(String[] args) {
        zapoln(new String[]{"M", "(", "5", ",", "m", "(", "6", ",", "8", ")",
")"});
        start();
        System.out.println(top);


    }
}
```

```
package com.company.Lab4;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayDeque;
import java.util.Arrays;

public class Task1 {
    public static void main(String[] args) {
        ArrayDeque<String> lines = new ArrayDeque<>();
        EnterText(lines);
        String[] text = lines.toArray(new String[0]);
        Arrays.sort(text);
        lines.clear();
        lines.addAll(Arrays.asList(text));
        System.out.println("\nРезультат: " + lines);
    }

    public static void EnterText(ArrayDeque<String> lin1) {
        try {
            File file = new File("C:\\Users\\Erop\\IdeaProjects\\Lab
2\\src\\com\\company\\Lab4\\input\\input.txt");
            FileReader fr = new FileReader(file);
            BufferedReader reader = new BufferedReader(fr);
            String line = reader.readLine();
            while (line != null) {
                System.out.println(line);
                lin1.add(line);
                line = reader.readLine();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

```
package com.company.Lab4;

import java.io.*;
```

```
public class Task2 {
    private final static char[] DEK = {'a', 'b', 'g', 'u', 'i', 'o', 'e', 't',
'n', 's', 'h', 'v', 'c', 'y'};

    public static void main(String[] args) throws IOException {
        BufferedReader reader = new BufferedReader(new FileReader(new
File("C:\\Users\\Erop\\IdeaProjects\\Lab
2\\src\\com\\company\\Lab4\\input\\input2.txt")));
        BufferedWriter writer = new BufferedWriter(new FileWriter(new
File("DEK_README.txt")));

        int i;
        while ((i = reader.read()) != -1) {
            char ch = (char) i;
            writer.append(switchLetter(ch));
            writer.flush();
        }
        reader.close();
        writer.close();
    }

    private static char switchLetter(char ch) {
        char outchar = '0';
        for (int i = 2; i < DEK.length; i++) {
            char c = DEK[i];
            if (c == ch) {
                outchar = DEK[i - 2];
                break;
            }
        }
        if (outchar == '0')
            outchar = ch;
        return outchar;
    }
}
package com.company.Lab4;

public class Task3_1 {
    // Структура для представления стека
    static class Stack {
        int capacity;
        int top;
        int[] array;
    }

    // функция для создания стека заданной емкости.
    Stack createStack(int capacity) {
        Stack stack = new Stack();
        stack.capacity = capacity;
        stack.top = -1;
        stack.array = new int[capacity];
        return stack;
    }

    // Стек заполнен, когда вершина равна last index
    boolean isFull(Stack stack) {
        return (stack.top == stack.capacity - 1);
    }
```

```
// Стек пуст, когда вершина равна -1
boolean isEmpty(Stack stack) {
    return (stack.top == -1);
}

// Функция для добавления элемента в стек. Это увеличивается сверху на 1
void push(Stack stack, int item) {
    if (isFull(stack))
        return;
    stack.array[++stack.top] = item;
}

// Функция для удаления элемента из стека. Это уменьшает вершину на 1
int pop(Stack stack) {
    if (isEmpty(stack))
        return Integer.MIN_VALUE;
    return stack.array[stack.top--];
}

// Функция для реализации легального движения между полюсами
void moveDisksBetweenTwoPoles(Stack src, Stack dest, char s, char d) {
    int pole1TopDisk = pop(src);
    int pole2TopDisk = pop(dest);
    // Когда полюс 1 пуст
    if (pole1TopDisk == Integer.MIN_VALUE) {
        push(src, pole2TopDisk);
        moveDisk(d, s, pole2TopDisk);
    }
    // Когда полюс pole2 пуст
    else if (pole2TopDisk == Integer.MIN_VALUE) {
        push(dest, pole1TopDisk);
        moveDisk(s, d, pole1TopDisk);
    }
    // Когда верхний диск pole1> верхний диск pole2
    else if (pole1TopDisk > pole2TopDisk) {
        push(src, pole1TopDisk);
        push(src, pole2TopDisk);
        moveDisk(d, s, pole2TopDisk);
    }
    // Когда верхний диск pole1 <верхний диск pole2
    else {
        push(dest, pole2TopDisk);
        push(dest, pole1TopDisk);
        moveDisk(s, d, pole1TopDisk);
    }
}

// Функция для отображения движения дисков
void moveDisk(char fromPeg, char toPeg, int disk) {
    System.out.println("Move the disk " + disk +
            " from " + fromPeg + " to " + toPeg);
}

// Функция для реализации загадки ТОН
void tohIterative(int num_of_disks, Stack
        src, Stack aux, Stack dest) {
    int i, total_num_of_moves;
```

```
        char s = '1', d = '3', a = '2';
        // Если количество дисков четное, то чередуем
        // полюс назначения и вспомогательный полюс
        if (num_of_disks % 2 == 0) {
            char temp = d;
            d = a;
            a = temp;
        }
        total_num_of_moves = (int) (Math.pow(2, num_of_disks) - 1);
        // Большие диски будут вставлены первыми
        for (i = num_of_disks; i >= 1; i--)
            push(src, i);
        for (i = 1; i <= total_num_of_moves; i++) {
            if (i % 3 == 1)
                moveDisksBetweenTwoPoles(src, dest, s, d);
            else if (i % 3 == 2)
                moveDisksBetweenTwoPoles(src, aux, s, a);
            else if (i % 3 == 0)
                moveDisksBetweenTwoPoles(aux, dest, a, d);
        }
    }

    // Программа драйвера для проверки вышеуказанных функций
    public static void main(String[] args) {
        // Ввод: количество дисков
        int num_of_disks = 3;
        Task3_1 ob = new Task3_1();
        Stack src, dest, aux;
        // Создаем три стека размером num_of_disks держать диски
        src = ob.createStack(num_of_disks);
        dest = ob.createStack(num_of_disks);
        aux = ob.createStack(num_of_disks);
        ob.tohIterative(num_of_disks, src, aux, dest);
    }
}


import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.util.Stack;

public class Task4 {
    public static void testBrackets(String str) {
        Stack<Character> left_brackets = new Stack<>();
        Stack<Character> right_brackets = new Stack<>();
        for (char c : str.toCharArray()) {
            if (c == ')') {
                right_brackets.push(c);
            } else if (c == '(') {
                left_brackets.push(c);
            }
        }
        while (!left_brackets.empty() && !right_brackets.empty()) {
            char left = left_brackets.peek();
            char right = right_brackets.peek();
            if (left == '(' && right == ')') {
```

```
                left_brackets.pop();
                right_brackets.pop();
            } else
                break;
        }
        if (left_brackets.empty() && right_brackets.empty())
            System.out.println("OK");
        else
            System.out.println("FAIL");
    }

    public static void main(String[] args) {
        try {
            File file = new File("C:\\Users\\Егор\\IdeaProjects\\Lab
2\\src\\com\\company\\Lab4\\input\\input4.txt");
            FileReader fr = new FileReader(file);
            BufferedReader reader = new BufferedReader(fr);
            StringBuilder line = new StringBuilder(reader.readLine());
            String tempLine = "";
            boolean bool = true;
            while (bool) {
                line.append(tempLine);
                tempLine = reader.readLine();
                if (tempLine == null)
                    bool = false;
            }
            System.out.println("Получившаяся строка: " + line);
            System.out.print("Проверка: ");
            testBrackets(line.toString());
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayDeque;
import java.util.Deque;

public class Task5 {
    public static boolean testBrackets(String str) {
        Deque<Character> brackets = new ArrayDeque<>();
        for (char c : str.toCharArray()) {
            switch (c) {
                case '(':
                    brackets.addFirst(c);
                    break;
                case ')':
                    if (brackets.isEmpty() ||
!brackets.removeFirst().equals('('))
                        return false;
                    break;
                default:
                    break;
            }
        }
```

```
        return brackets.isEmpty();
    }

    public static void main(String[] args) {
        try {
            File file = new File("C:\\Users\\Егор\\IdeaProjects\\Lab
2\\src\\com\\company\\Lab4\\input\\input4.txt");
            FileReader fr = new FileReader(file);
            BufferedReader reader = new BufferedReader(fr);
            StringBuilder line = new StringBuilder(reader.readLine());
            String tempLine = "";
            boolean bool = true;
            while (bool) {
                line.append(tempLine);
                tempLine = reader.readLine();
                if (tempLine == null)
                    bool = false;
            }
            System.out.println("Получившаяся строка: " + line);
            System.out.println("Проверка: " + (testBrackets(line.toString()) ?
"OK" : "FAIL"));
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
package com.company.Lab4;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.util.Stack;

public class Task6 {
    public static void main(String[] args) {
        try {
            File file = new File("C:\\Users\\Егор\\IdeaProjects\\Lab
2\\src\\com\\company\\Lab4\\input\\input5.txt");
            FileReader fr = new FileReader(file);
            BufferedReader reader = new BufferedReader(fr);
            String line = String.valueOf(reader.readLine());
            Chain(line);
        }
        catch(IOException e) {
            e.printStackTrace();
        }
    }

    public static void Chain(String line) {
        Stack<Character> chain = new Stack<>();
        for(int i = 0; i < line.length(); i++) {
            if(Character.isDigit(line.charAt(i))) {
                chain.push(line.charAt(i));
            }
        }
        for(int i = 0; i < line.length(); i++) {
            if(Character.isLetter(line.charAt(i))) {
```

```
                chain.push(line.charAt(i));
            }
        }
        for(int i = 0; i < line.length(); i++) {
            if(!Character.isDigit(line.charAt(i)) &&
!Character.isLetter(line.charAt(i)) ) {
                chain.push(line.charAt(i));
            }
        }
        System.out.println(chain);
    }
}
```

```
package com.company.Lab4;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayDeque;
import java.util.Deque;
import java.util.Stack;

public class Task7 {
    public static void main(String[] args) {
        try {
            File file = new File("C:\\Users\\Erop\\IdeaProjects\\Lab
2\\src\\com\\company\\Lab4\\input\\input7.txt");
            FileReader fr = new FileReader(file);
            BufferedReader reader = new BufferedReader(fr);
            String line = String.valueOf(reader.readLine());
            Chain1(line);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static void Chain1(String line) {
        Deque<Integer> chain = new ArrayDeque<>();
        String[] strArr = line.split(" ");
        int[] numArr = new int[strArr.length];
        for (int i = 0; i < strArr.length; i++) {
            numArr[i] = Integer.parseInt(strArr[i]);
        }
        for (int i = 0; i < numArr.length; i++) {
            if(numArr[i] < 0) {
                chain.addLast(numArr[i]);
            }
        }
        for (int i = 0; i < numArr.length; i++) {
            if(numArr[i] > 0) {
                chain.addLast(numArr[i]);
            }
        }
        System.out.println(chain);
    }
}
```

```
package com.company.Lab4;

import java.io.*;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Scanner;
import java.util.Stack;

public class Task8 {
    public static void main(String[] args) throws FileNotFoundException,
UnsupportedEncodingException {
        PrintWriter writer = new PrintWriter("C:\\Users\\Erop\\IdeaProjects\\Lab
2\\src\\com\\company\\Lab4\\input\\output8.txt", "UTF-8");
        Stack<String> list = new Stack<>();
        try (Scanner scan = new Scanner(new
File("C:\\Users\\Erop\\IdeaProjects\\Lab
2\\src\\com\\company\\Lab4\\input\\input8.txt"))) {
            while (scan.hasNextLine()) {
                list.push(scan.nextLine());
            }
            while(!list.empty()) {
                String out = list.pop();
                writer.println(out);
            }
            System.out.println("Выполнено!");
            writer.close();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
    }
}
package com.company.Lab4;

import java.util.Stack;

public class zad9 {
    public static Stack<String> slov= new Stack<>();
    public static String first="NULL";
    public static String second="NULL";
    public static String top="NULL";
    public static String logZnach="NULL";
    public static String tokenNot="NULL";
    public static void zapoln(String [] s) {

        for (int i = 0; i < s.length; i++) {
            slov.push(s[i]);
        }
    }

    public static void sd() {

        //для одного знач

if(logZnach.equals("N")&&(first.equals("NULL")&&!top.equals("NULL")&&tokenNot.eq
uals("NULL"))){
            if(top.equals("T")){
                top="F";
```

```
                logZnach="NULL";
                tokenNot="YES";
                //System.out.println("SDDDD N firs= "+first +" top="+top+"
logznach="+logZnach+"Token="+tokenNot);

            }
            if(top.equals("F")){
                top="T";
                logZnach="NULL";
                tokenNot="YES";
                //System.out.println("SDDDD N firs= "+first +" top="+top+"
logznach="+logZnach+"Token="+tokenNot);

            }


        }
        if(logZnach.equals("A")&&(!top.equals("NULL")&&!first.equals("NULL"))){

if((first.equals("T")&&top.equals("F"))||(first.equals("F")&&top.equals("T"))){
                top="F";
                first="NULL";
                logZnach="NULL";
               // System.out.println("od A firs= "+first +"top="+top+"
logznach="+logZnach);


            }else{ if(first.equals("T")&&top.equals("T")){
                top="T";
                first="NULL";

                logZnach="NULL";
               // System.out.println("od A firs= "+first +"top="+top+"
logznach="+logZnach);


            }else{
                top="F";
                first="NULL";
                logZnach="NULL";
               // System.out.println("od A firs= "+first +"top="+top+"
logznach="+logZnach);


            }
            }
            //System.out.println("firs="+first+"top="+top+"M");



        }
        if(logZnach.equals("X")&&(!top.equals("NULL")&&!first.equals("NULL"))){

if((first.equals("T")&&top.equals("T"))||(first.equals("F")&&top.equals("F"))){
                top="F";
                first="NULL";
                logZnach="NULL";
                //System.out.println("od X firs= "+first +"top="+top+"
```

```
logznach="+logZnach);

            }
            else{
                top="T";
                first="NULL";
                logZnach="NULL";
                //System.out.println("od X firs= "+first +"top="+top+"
logznach="+logZnach);
            }

        }
        if(logZnach.equals("O")&&(!top.equals("NULL")&&!first.equals("NULL"))){
            if(first.equals("F")&&top.equals("F")){
                top="F";
                first="NULL";
                logZnach="NULL";
              //  System.out.println("od O firs= "+first +"top="+top+"
logznach="+logZnach);
            }
            else{
                top="T";
                first="NULL";
                logZnach="NULL";
                //System.out.println("od O firs= "+first +"top="+top+"
logznach="+logZnach);
            }

        }

    }
    public static void start() {
        String d;
        int l=slov.size();
        for (int i = 0; i <l; i++) {
            d=slov.pop();
            if(first.equals("NULL")&&(d.equals("T")||d.equals("F"))){
                first=d;
              //  System.out.println(" zapolneneie firs= "+first);
                continue;
            }

if(logZnach.equals("NULL")&&(d.equals("N")||d.equals("A")||d.equals("X")||d.equa
ls("O"))){
                logZnach=d;
                tokenNot="NULL";
               // System.out.println(" zapolneneie logZnach "+logZnach);
                continue;
            }
            if(second.equals("NULL")&&(d.equals("T")||d.equals("F"))){
                second=d;

               // System.out.println(" zapolneneie second= "+second);
                continue;
            }

if(logZnach.equals("N")&&(!first.equals("NULL")&&tokenNot.equals("NULL"))){
                if(first.equals("T")){
```

```
                    top="F";
                    first="NULL";
                    logZnach="NULL";
                    tokenNot="YES";
                //    System.out.println(" N firs= "+first +" top="+top+"
logznach="+logZnach+"Token="+tokenNot);continue;

                }
                if(first.equals("F")){
                    top="T";
                    first="NULL";
                    logZnach="NULL";
                    tokenNot="YES";
                  //  System.out.println(" N firs= "+first +" top="+top+"
logznach="+logZnach+"Token="+tokenNot);continue;

                }


            }

if(logZnach.equals("A")&&(!first.equals("NULL")&&!second.equals("NULL"))){

if((first.equals("T")&&second.equals("F"))||(first.equals("F")&&second.equals("T
"))){
                    top="F";
                    first="NULL";
                    second="NULL";
                    logZnach="NULL";
                   // System.out.println(" A firs= "+first +"second="+second+ "
top="+top+" logznach="+logZnach);

                }else{ if(first.equals("T")&&second.equals("T")){
                    top="T";
                    first="NULL";
                    second="NULL";
                    logZnach="NULL";
                 //   System.out.println(" A firs= "+first +"second="+second+ "
top="+top+" logznach="+logZnach);

                }else{
                    top="F";
                    first="NULL";
                    second="NULL";
                    logZnach="NULL";
                 //   System.out.println(" A firs= "+first +"second="+second+ "
top="+top+" logznach="+logZnach);

                }
                }
                //System.out.println("firs="+first+"top="+top+"M");


                continue;
            }

if(logZnach.equals("X")&&(!first.equals("NULL")&&!second.equals("NULL"))){
```

```
if((first.equals("T")&&second.equals("T"))||(first.equals("F")&&second.equals("F
"))){
                    top="F";
                    first="NULL";
                    second="NULL";
                    logZnach="NULL";
                    //System.out.println(" X firs= "+first +"second="+second+ "
top="+top+" logznach="+logZnach);

                }
                else{
                    top="T";
                    first="NULL";
                    second="NULL";
                    logZnach="NULL";
                   // System.out.println(" X firs= "+first +"second="+second+ "
top="+top+" logznach="+logZnach);
                }
                continue;
            }

if(logZnach.equals("O")&&(!first.equals("NULL")&&!second.equals("NULL"))){
                if(first.equals("F")&&second.equals("F")){
                    top="F";
                    first="NULL";
                    second="NULL";
                    logZnach="NULL";
                  //  System.out.println(" O firs= "+first +"second="+second+ "
top="+top+" logznach="+logZnach);

                }
                else{
                    top="T";
                    first="NULL";
                    second="NULL";
                    logZnach="NULL";
                  //  System.out.println(" O firs= "+first +"second="+second+ "
top="+top+" logznach="+logZnach);
                }
                continue;
            }
            ///OD

if(logZnach.equals("N")&&(first.equals("NULL")&&!top.equals("NULL")&&tokenNot.eq
uals("NULL"))){
                if(top.equals("T")){
                    top="F";

                    logZnach="NULL";
                    tokenNot="YES";
              //      System.out.println(" N firs= "+first +" top="+top+"
logznach="+logZnach+"Token="+tokenNot);
                    continue;
                }
                if(top.equals("F")){
                    top="T";
                    logZnach="NULL";
                    tokenNot="YES";
```

```
                //    System.out.println(" N firs= "+first +" top="+top+"
logznach="+logZnach+"Token="+tokenNot);
                    continue;
                }


            }

if(logZnach.equals("A")&&(!top.equals("NULL")&&!first.equals("NULL"))){

if((first.equals("T")&&top.equals("F"))||(first.equals("F")&&top.equals("T"))){
                    top="F";
                    first="NULL";
                    logZnach="NULL";
                  //  System.out.println("od A firs= "+first +"top="+top+"
logznach="+logZnach);


                }else{ if(first.equals("T")&&top.equals("T")){
                    top="T";
                    first="NULL";

                    logZnach="NULL";
                  //  System.out.println("od A firs= "+first +"top="+top+"
logznach="+logZnach);


                }else{
                    top="F";
                    first="NULL";
                    logZnach="NULL";
                  //  System.out.println("od A firs= "+first +"top="+top+"
logznach="+logZnach);


                }
                }
                //System.out.println("firs="+first+"top="+top+"M");



            }

if(logZnach.equals("X")&&(!top.equals("NULL")&&!first.equals("NULL"))){

if((first.equals("T")&&top.equals("T"))||(first.equals("F")&&top.equals("F"))){
                    top="F";
                    first="NULL";
                    logZnach="NULL";
                //    System.out.println("od X firs= "+first +"top="+top+"
logznach="+logZnach);

                }
                else{
                    top="T";
                    first="NULL";
                    logZnach="NULL";
                //    System.out.println("od X firs= "+first +"top="+top+"
```

```
logznach="+logZnach);
                }

            }

if(logZnach.equals("O")&&(!top.equals("NULL")&&!first.equals("NULL"))){
                if(first.equals("F")&&top.equals("F")){
                    top="F";
                    first="NULL";
                    logZnach="NULL";
                //    System.out.println("od O firs= "+first +"top="+top+"
logznach="+logZnach);
                }
                else{
                    top="T";
                    first="NULL";
                    logZnach="NULL";
                //    System.out.println("od O firs= "+first +"top="+top+"
logznach="+logZnach);
                }

            }
        }
        sd();
    }
    //Проверь при помощи sout
    public static void main(String[] args) {
        zapoln(new String[]{"T","X", "(", "T", "A", "(","N", "(", "F", "O", "T",
")", ")",")"});
        start();
        System.out.println(top);


    }
}
/*
struct List
{
    char data;
    List* next;
};

List* push_back(char, List*);
List* push_front(char, List*);
List* pop_front(List*);
void viewspis(List*);
void clearspis(List*)




////////////////////////////////////////////
#include<iostream>
#include"modulSpis.h"
using namespace std;

List* push_back(char ch, List* head)
```

```
{
    List* newList = new List;
    newList->data = ch;
    newList->next = NULL;

    if(!head)
    {
        head = newList;
    }
    else
    {
        List* tec = head;
        while(tec->next)
        {
            tec = tec->next;
        }
        tec->next = newList;
    }

    return head;
}

List* push_front(char ch, List* head)
{
    List* newList = new List;
    newList->data = ch;
    newList->next = head;
    head = newList;
    return head;
}

void viewspis(List* first)
{
    while(first)
    {
        cout << first->data;
        first = first->next;
    }
}

void clearspis(List* first)
{
    List* temp = NULL;
    while(first)
    {
        temp = first;
        first = first->next;
        delete temp;
    }
}

List* pop_front(List* first)
{
    List* temp1 = first;
    first = first->next;
    delete temp1;
    return first;
}
```

```
//////////
using namespace std;

int main()
{
    setlocale(LC_ALL,"Russian");
    List* first = NULL;
    List* second = NULL;
    string formula;
    cout << "Введите формулу: " << endl;
    cin >> formula;
    int size = formula.size();
    for(int i = 0; i < size;)
    {
        if(formula[i] != ')')
        {
            if(formula[i] != '(')
            {
                first = push_front(formula[i],first);
            }
            i++;
        }

        else if(first->next)
        {
            char var = first->next->data;
            switch(var)
            {
            case 'N':
                {
                    if(first->data == 'T')
                    {

                        i++;
                        for(int t = 0; t < 2; t++)
                        {
                            first = pop_front(first);
                        }
                        first = push_front('F',first);
                    }
                    else
                    {
                        i++;
                        for(int t = 0; t < 2; t++)
                        {
                            first = pop_front(first);
                        }
                        first = push_front('T',first);
                    }
                }
            case 'A':
                {
                    if(first->data == 'T' && first->next->next->data == 'T')
                    {
                        i++;
```

```
                    for(int t = 0; t < 3; t++)
                    {
                        first = pop_front(first);
                    }
                    first = push_front('T',first);
                    break;
                }
                else
                {
                    i++;
                    for(int t = 0; t < 3; t++)
                    {
                        first = pop_front(first);
                    }
                    first = push_front('F',first);
                    break;
                }
            }
        case 'X':
            {
                if(first->data == first->next->next->data)
                {
                    i++;
                    for(int t = 0; t < 3; t++)
                    {
                        first = pop_front(first);
                    }
                    first = push_front('F',first);
                    break;
                }
                else
                {
                    i++;
                    for(int t = 0; t < 3; t++)
                    {
                        first = pop_front(first);
                    }
                    first = push_front('T',first);
                    break;
                }
            }
        case 'O':
            {
                if(first->data == 'F' && first->next->next->data == 'F')
                {
                    i++;
                    for(int t = 0; t < 3; t++)
                    {
                        first = pop_front(first);
                    }
                    first = push_front('F',first);
                    break;
                }
                else
                {
                    i++;
                    for(int t = 0; t < 3; t++)
                    {
```

```
                            first = pop_front(first);
                        }
                        first = push_front('T',first);
                        break;
                    }
                }
            }
        }


    viewspis(first);
    cout<<"\n";
    //viewspis(second);


    return 0;
}
 */
package com.company.Lab4;
import java.io.*;
import java.util.Iterator;
import java.util.Stack;
public class zad11 {
    public static void main(String[] args) throws IOException {
        BufferedReader reader = new BufferedReader(new FileReader(new
File(System.getProperty("user.dir")+"\\TEST.txt")));
        Stack st = new Stack();
        Stack letter = new Stack();
        Stack symbols = new Stack();
        int open = 0,close = 0;
        int i = 0;
        while ((i = reader.read()) != -1) {
            char ch = (char) i;
            st.add(ch);
        }
        Iterator iterator1 = st.iterator();
        while (iterator1.hasNext()) {
            char r = (char) iterator1.next();
            if (Character.isAlphabetic(r)) {
                letter.push(r);
            }
        }
        Iterator iterator2 = st.iterator();
        while (iterator2.hasNext()) {
            char r = (char) iterator2.next();
            if (!(Character.isDigit(r) || Character.isAlphabetic(r))) {
                symbols.push(r);
            }
        }
        int kol = 0;
        while (symbols.size() != 0) {
            char s = (char) symbols.pop();
            switch (s) {
                case ('+'):
                case ('-'):
                    kol++;
                    break;
```

```
                case ('('):
                    open++;
                    break;
                case (')'):
                    close++;
                    break;
            }
        }
        int w = 0;
        while (letter.size() != 0) {
            char s = (char) letter.pop();
            switch (s) {
                case ('x'):
                case ('y'):
                case ('z'):
                    w++;

                    break;
            }
        }

        if ((w-1 == kol) && (close==open))
            System.out.println("Формула имеет правильный вид");
        else
            System.out.println("Формула имеет не правильный вид");
    }
}
```

3.2 Результат выполнения программы

```
6
```

Рисунок 1 – результат выполнения

```
Властелин колец
Гордость и предубеждение
Тёмные начала
Автостопом по галактике
Гарри Поттер и Кубок огня
Убить пересмешника
Винни Пух
Лев, колдунья и платяной шкаф
Джейн Эйр
Уловка-22
Грозовой перевал
Пение птиц
Ребекка
Над пропастью во ржи

Результат: [Автостопом по галактике, Винни Пух, Властелин колец, Гарри Поттер и Кубок огня, Гордость и предубеждение, Грозовой перевал, Джейн Эйр, Лев, колдунья и платяной шкаф,
```

Рисунок 2 – результат выполнения

```
Move the disk 1 from 1 to 3
Move the disk 2 from 1 to 2
Move the disk 1 from 3 to 2
Move the disk 3 from 1 to 3
Move the disk 1 from 2 to 1
Move the disk 2 from 2 to 3
Move the disk 1 from 1 to 3
```

Рисунок 3– результат выполнения

```
Получившаяся строка: алг Сумма квадратов (арг цел n, рез цел S)  дано | n > 0  надо | S = 1*1 + 2*2 + 3*3 + … + n*nнач цел i|  ввод n; S:=0|  нц для i от 1 до n|  |  S := S +
Проверка: OK
```

Рисунок 4 – результат выполнения

```
[3, 2, 4, 7, 3, 2, 9, 4, 8, 4, 0, 9, 3, 2, 8, 7, 4, 0, 7, 8, o, r, g, h, r, b, o, u, h, r, g, o, r, g, h, r, i, g, g, d, r, j, h, v, d, r, v, j, r, d, o, v, r, g, i, ;, [, -, -,
```

Рисунок 5 – результат выполнения

```
[-14, -18, -128, -1891951, -5949, 12, 13, 2022, 98491, 6, 6988]
```

Рисунок 6 – результат выполнения

```
Выполнено!
```

Рисунок 7 – результат выполнения

```
T
```

Рисунок 8 – результат выполнения

```
Формула имеет правильный вид
```

Рисунок 9 – результат выполнения

Список использованных источников

1) ГОСТ 7.32-2017 Система стандартов по информации, библиотечному и издательскому делу. Отчёт о научно-исследовательской работе. Структура и правила оформления

2) ГОСТ 7.1-2003 Библиографическая запись. Библиографическое описание. Общие требования и правила составления