Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации

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

на тему: «Реализация стека/дека»

Выполнила:

Студентка группы БФИ1902

Струкова А.В.

Вариант 18

Проверил:

Москва, 2021 г.

Оглавление

[1. Цель работы 3](#_Toc72412872)

[2. Задание на лабораторную работу 3](#_Toc72412873)

[2.1. Задание 1 3](#_Toc72412874)

[2.2. Задание 2 3](#_Toc72412875)

[2.3. Задание 3 3](#_Toc72412876)

[2.4. Задание 4 3](#_Toc72412877)

[2.5. Задание 5 3](#_Toc72412878)

[2.6. Задание 6 4](#_Toc72412879)

[2.7. Задание 7 4](#_Toc72412880)

[2.8. Задание 8 4](#_Toc72412881)

[2.9. Задание 9 4](#_Toc72412882)

[2.10. Задание 10 4](#_Toc72412883)

[2.11. Задание 11 4](#_Toc72412884)

[3. Листинг программы 4](#_Toc72412885)

[4. Результат работы программы 16](#_Toc72412886)

[Список использованных источников 22](#_Toc72412887)

# **1. Цель работы**

Цель работы: реализовать следующие структуры данных: стек (stack) и дек (двусторонняя очередь, deque).

# **2. Задание на лабораторную работу**

## 2.1. Задание 1

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

## 2.2. Задание 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.

## 2.3. Задание 3

Даны три стержня и n дисков различного размера. Диски можно надевать на стержни, образуя из них башни. Перенести n дисков со стержня А на стержень С, 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 В. Реализовать алгоритм, используя три стека вместо стержней А, В, С. Информация о дисках хранится в исходном файле.

## 2.4. Задание 4

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

## 2.5. Задание 5

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

## 2.6. Задание 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.

## 2.7. Задание 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.

## 2.8. Задание 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.

## 2.9. Задание 9

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

## 2.10. Задание 10

Дан текстовый файл. В текстовом файле записана формула следующего вида: ::= | M(,) | N(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.

## 2.11. Задание 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. Листинг программы**

package com.company;  
  
*//Отсортировать строки файла, содержащие названия книг, в алфавитном порядке с  
//использованием двух деков*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\\Королева\\Desktop\\LR4\\text1.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;  
  
*//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*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\\Королева\\Desktop\\LR4\\text2.txt")));  
 BufferedWriter writer = new BufferedWriter(new FileWriter(new File("C:\\Users\\Королева\\Desktop\\LR4\\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]; *//меняем через один* break;  
 }  
 }  
 if (outchar == '0')  
 outchar = ch;  
 return outchar;  
 }  
}

package com.company;  
  
*//Даны три стержня и n дисков различного размера. Диски можно надевать на  
//стержни, образуя из них башни. Перенести n дисков со стержня А на стержень С,  
//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 В.  
//Реализовать алгоритм, используя три стека вместо стержней А, В, С. Информация  
//о дисках хранится в исходном файле.*public class Task3 {  
 *//Структура для представления стека* static class Stack {  
 int capacity; *//отвечает за емкость* int top;  
 int[] array;  
}  
 *//Создание 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, если вершина равна stack.capacity - 1* boolean Full(Stack stack) {  
 return (stack.top == stack.capacity - 1);  
 }  
  
 *// Стек пуст, если вершина равна -1* boolean Empty(Stack stack) {  
 return (stack.top == -1);  
 }  
  
 *// Функция для добавления элемента в стек, что увеличивает вершину на 1* void push(Stack stack, int item) {  
 if (Full(stack))  
 return;  
 stack.array[++stack.top] = item;  
 }  
  
 *// Функция для удаления элемента из стека, что уменьшает вершину на 1* int pop(Stack stack) {  
 if (Empty(stack))  
 return Integer.MIN\_VALUE;  
 return stack.array[stack.top--];  
 }  
  
 *// Функция для реализации движения между стержнями* void moveDiscs (Stack src, Stack dest, char s, char d) {  
 int rod1 = pop(src);  
 int rod2 = pop(dest);  
 *// Когда стержень 1 пуст* if (rod1 == Integer.MIN\_VALUE) {  
 push(src, rod2);  
 motionDisplay(d, s, rod2);  
 }  
 *// Когда стержень 2 пуст* else if (rod2 == Integer.*MIN\_VALUE*) {  
 push(dest, rod1);  
 motionDisplay(s, d, rod1);  
 }  
 *// Когда верхний диск стержня 1> верхний диск стержня 2* else if (rod1 > rod2) {  
 push(src, rod1);  
 push(src, rod2);  
 motionDisplay(d, s, rod2);  
 }  
 *// Когда верхний диск стержня 1 <верхний диск стержня 2* else {  
 push(dest, rod2);  
 push(dest, rod1);  
 motionDisplay(s, d, rod1);  
 }  
 }  
  
 *// Функция для отображения движения дисков* void motionDisplay(char fromPeg, char toPeg, int disk) {  
 System.*out*.println("Move the disk " + disk +  
 " from " + fromPeg + " to " + toPeg);  
 }  
  
 *// Функция для реализации загадки* 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  
 // стержень назначения и вспомогательный стержень* 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)  
 moveDiscs(src, dest, s, d);  
 else if (i % 3 == 2)  
 moveDiscs(src, aux, s, a);  
 else if (i % 3 == 0)  
 moveDiscs(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 ob = new Task3();  
 Stack src, dest, aux;  
 *// Создаем три стека размером num\_of\_disks* 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);  
 }  
}

package com.company;  
  
*//Дан текстовый файл с программой на алгоритмическом языке. За один просмотр  
//файла проверить баланс круглых скобок в тексте, используя стек*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(); *//вытаскиваем* 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\\Королева\\Desktop\\LR4\\text4.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();  
 }  
 }  
 }

package com.company;  
  
*//Дан текстовый файл с программой на алгоритмическом языке. За один просмотр  
//файла проверить баланс квадратных скобок в тексте, используя дек.*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\\Королева\\Desktop\\LR4\\text4.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); *//закидываем в стринг* 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;  
  
*//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*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\\Королева\\Desktop\\LR4\\text6.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))) { *//определяет, является ли указанное значение типа char цифрой* 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;  
  
*//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*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\\Королева\\Desktop\\LR4\\text7.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;  
  
*//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*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\\Королева\\Desktop\\LR4\\textout.txt", "UTF-8");  
 Stack<String>list = new Stack<>();  
 try (Scanner scan = new Scanner(new File("C:\\Users\\Королева\\Desktop\\LR4\\text8.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;  
  
*//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*import java.io.\*;  
import java.util.Iterator;  
import java.util.Stack;  
  
public class Task9 {  
 public static void main(String[] args) throws IOException {  
 BufferedReader reader = new BufferedReader(new FileReader(new File("C:\\Users\\Королева\\Desktop\\LR4\\text9.txt")));  
 Stack<Character> st = new Stack<Character>();  
 Stack<Boolean> num = new Stack<Boolean>();  
 Stack<Character> letter = new Stack<Character>();  
 Stack<Boolean> preanswer = new Stack<Boolean>();  
 int i = 0;  
  
 while ((i = reader.read()) != -1) {  
 char ch = (char) i;  
 st.add(ch);  
 }  
  
 for (char r : st) {  
 if (Character.*isDigit*(r)) {  
 if (r == '0')  
 num.push(false);  
 else  
 num.push(true);  
 }  
 }  
  
 for (char r : st) {  
 if (Character.*isAlphabetic*(r)) {  
 letter.push(r);  
 }  
 }  
  
 while (preanswer.size() != 6) {  
 char s = letter.pop();  
 switch (s) {  
 case ('O'):  
 boolean num1 = num.pop();  
 boolean num2 = num.pop();  
 preanswer.add(num1 || num2);  
 break;  
 case ('X'):  
 boolean num3 = num.pop();  
 boolean num4 = num.pop();  
 if (num3 != num4) {  
 preanswer.add(true);  
 } else {  
 preanswer.add(false);  
 }  
 break;  
 case ('A'):  
 boolean num5 = num.pop();  
 boolean num6 = num.pop();  
 preanswer.add(num5 && num6);  
 break;  
 case ('N'):  
 boolean num7 = num.pop();  
 if (num7) {  
 preanswer.add(false);  
 } else {  
 preanswer.add(true);  
 }  
 break;  
 case ('F'):  
 preanswer.add(false);  
 break;  
 case ('T'):  
 preanswer.add(true);  
 break;  
 }  
 }  
  
 Iterator <Boolean> iterator2 = preanswer.iterator();  
 boolean answer = false;  
 while (iterator2.hasNext()) { *//есть ли след элемент* if (iterator2.next()){ *//след элемент* answer = true;  
 }  
 }  
 System.*out*.println(answer);  
 }  
}

package com.company;  
  
*//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.*import java.util.Stack;  
  
public class Task10 {  
 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 (String value : s) {  
 *slov*.push(value);  
 }  
 }  
  
 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);  
 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);  
 continue;  
 }  
  
 if(d.equals("m")&&*first*!=-1&&*second*!=-1){  
 *top*=Math.*min*(*first*,*second*);  
 *first*=-1;  
 *second*=-1;  
 continue;  
 }  
  
 if(d.equals("m")&&*first*!=-1){  
 *top*=Math.*min*(*first*,*top*);  
 *first*=-1;  
 continue;  
 }  
  
 if(d.equals("M")&&*first*!=-1&&*second*!=-1){  
 *top*=Math.*max*(*first*,*second*);  
 *first*=-1;  
 *second*=-1;  
 continue;  
 }  
  
 if(d.equals("M")&&*first*!=-1){  
 *top*=Math.*max*(*first*,*top*);  
 *first*=-1;  
 }  
 }  
 }  
  
 public static void main(String[] args) {  
 *zapoln*(new String[]{"M", "(", "5", ",", "m", "(", "6", ",", "8", ")", ")"});  
 *start*();  
 System.out.println(top);  
 }  
}

package com.company;  
  
*//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*import java.io.\*;  
import java.util.Iterator;  
import java.util.Stack;  
  
public class Task11 {  
 public static void main(String[] args) throws IOException {  
 BufferedReader reader = new BufferedReader(new FileReader(new File("C:\\Users\\Королева\\Desktop\\LR4\\text11.txt")));  
 Stack<Character> st = new Stack<>();  
 Stack<Character> letter = new Stack<Character>();  
 Stack<Character> symbols = new Stack<Character>();  
 int open = 0,close = 0;  
 int i = 0;  
 while ((i = reader.read()) != -1) {  
 char ch = (char) i;  
 st.add(ch);  
 }  
 for (char r : st) {  
 if (Character.isAlphabetic(r)) {  
 letter.push(r);  
 }  
 }  
 for (char r : st) {  
 if (!(Character.isDigit(r) || Character.isAlphabetic(r))) {  
 symbols.push(r);  
 }  
 }  
 int kol = 0;  
 while (symbols.size() != 0) { *//колво на 1 больше чем символы* char s = symbols.pop();  
 switch (s) {  
 case ('+'):  
 case ('-'):  
 kol++;  
 break;  
 case ('('):  
 open++;  
 break;  
 case (')'):  
 close++;  
 break;  
 }  
 }  
  
 int w = 0;  
 while (letter.size() != 0) {  
 char s = letter.pop();  
 switch (s) {  
 case ('x'):  
 case ('y'):  
 case ('z'):  
 w++;  
 break;  
 }  
 }  
  
 if ((close==open) && (w-1 == kol))  
 System.*out*.println("Формула имеет правильный вид");  
 else  
 System.*out*.println("Формула имеет не правильный вид");  
 }  
}

# **4. Результат работы программы**

Результат работы программы представлен на Рисунках 1-11.

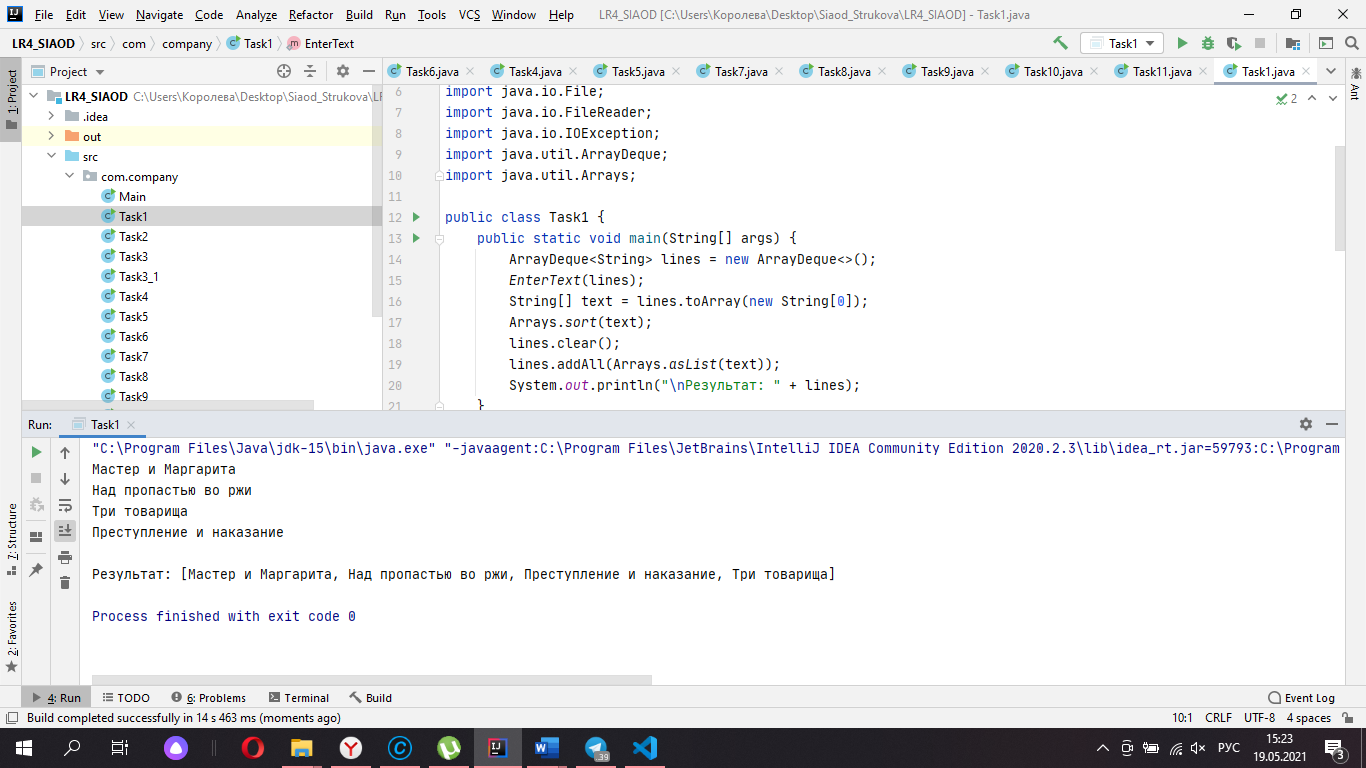


Рисунок 1 - Task 1

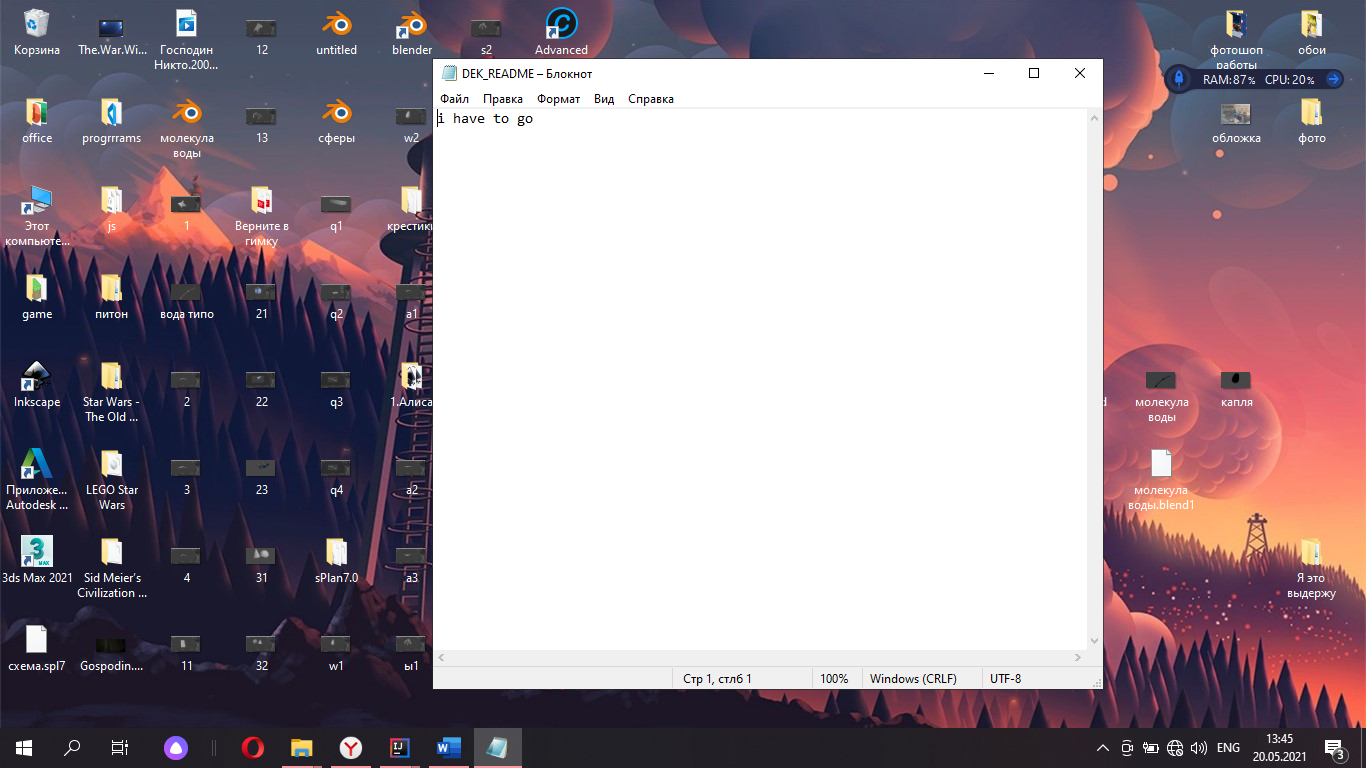


Рисунок 2 - Task 2

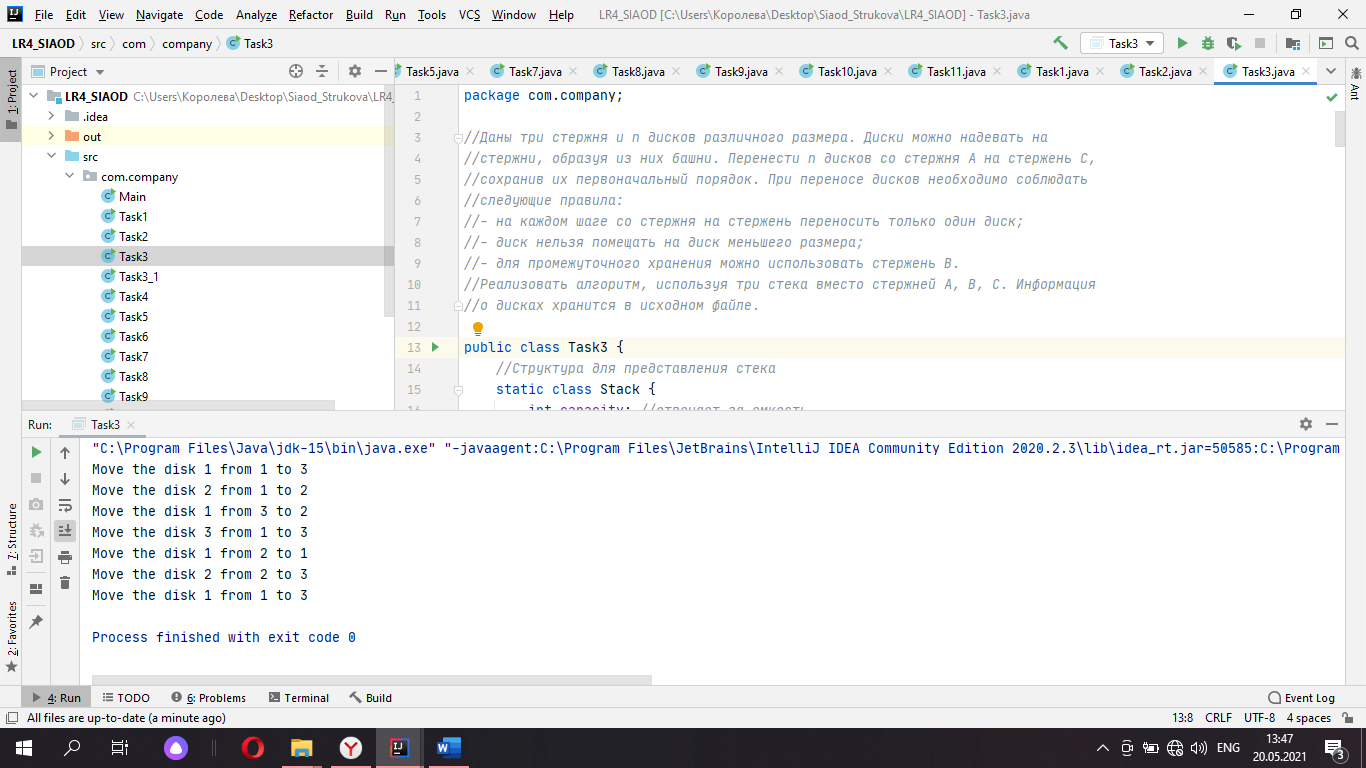


Рисунок 3 - Task 3

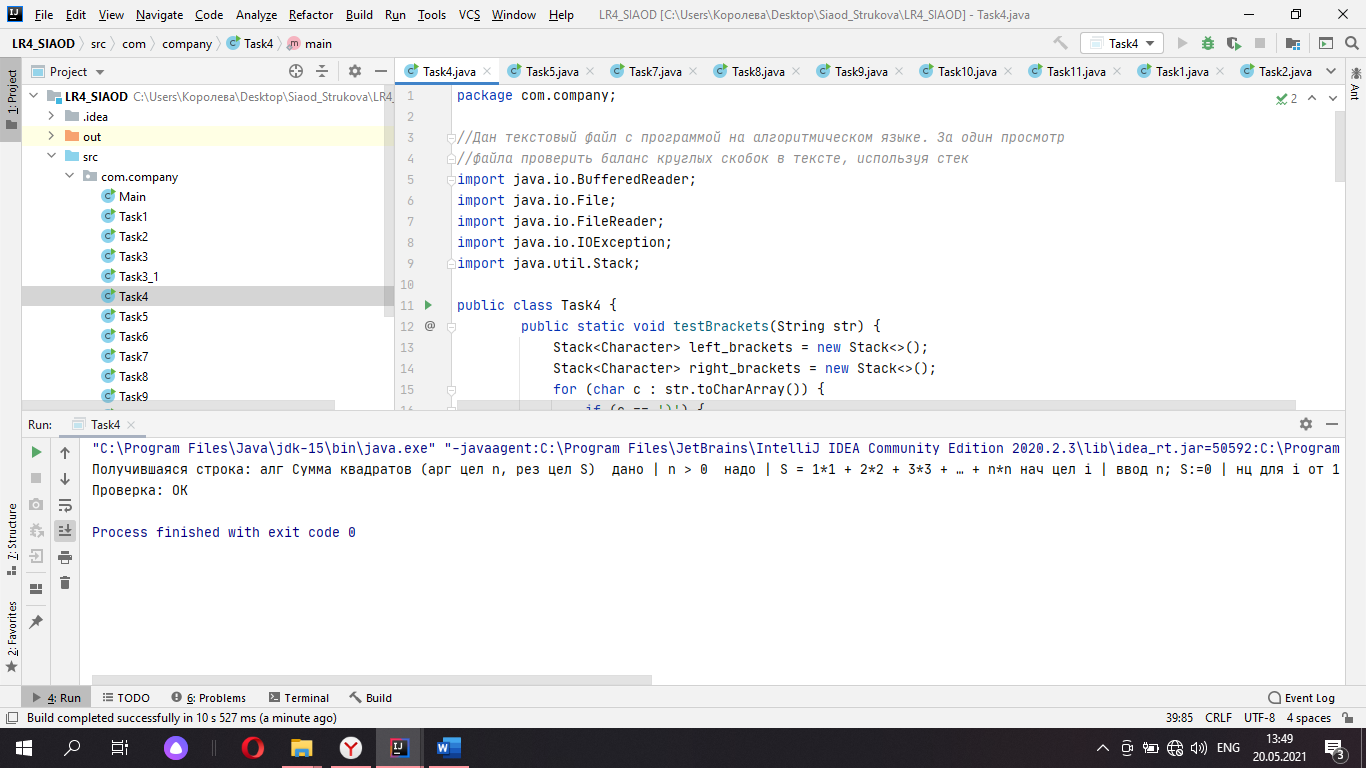


Рисунок 4 - Task 4

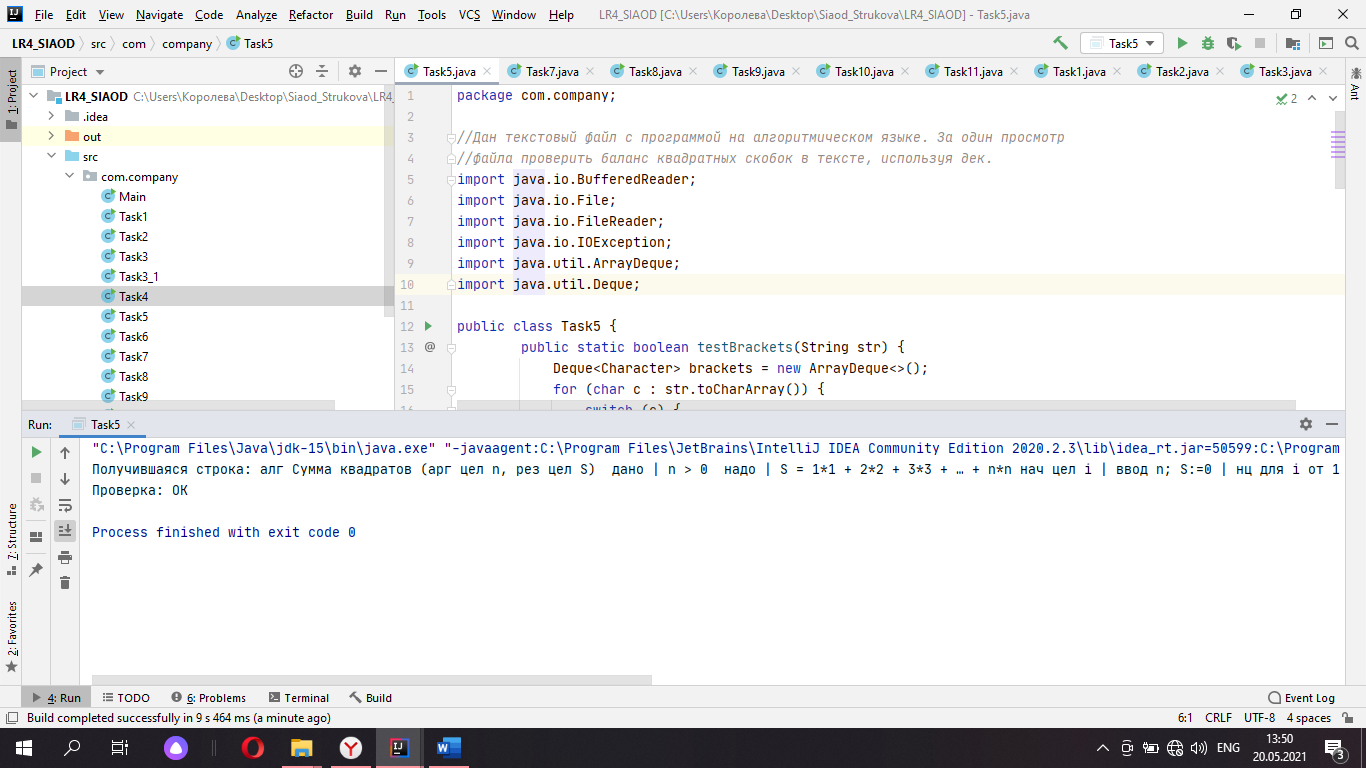


Рисунок 5 - Task 5

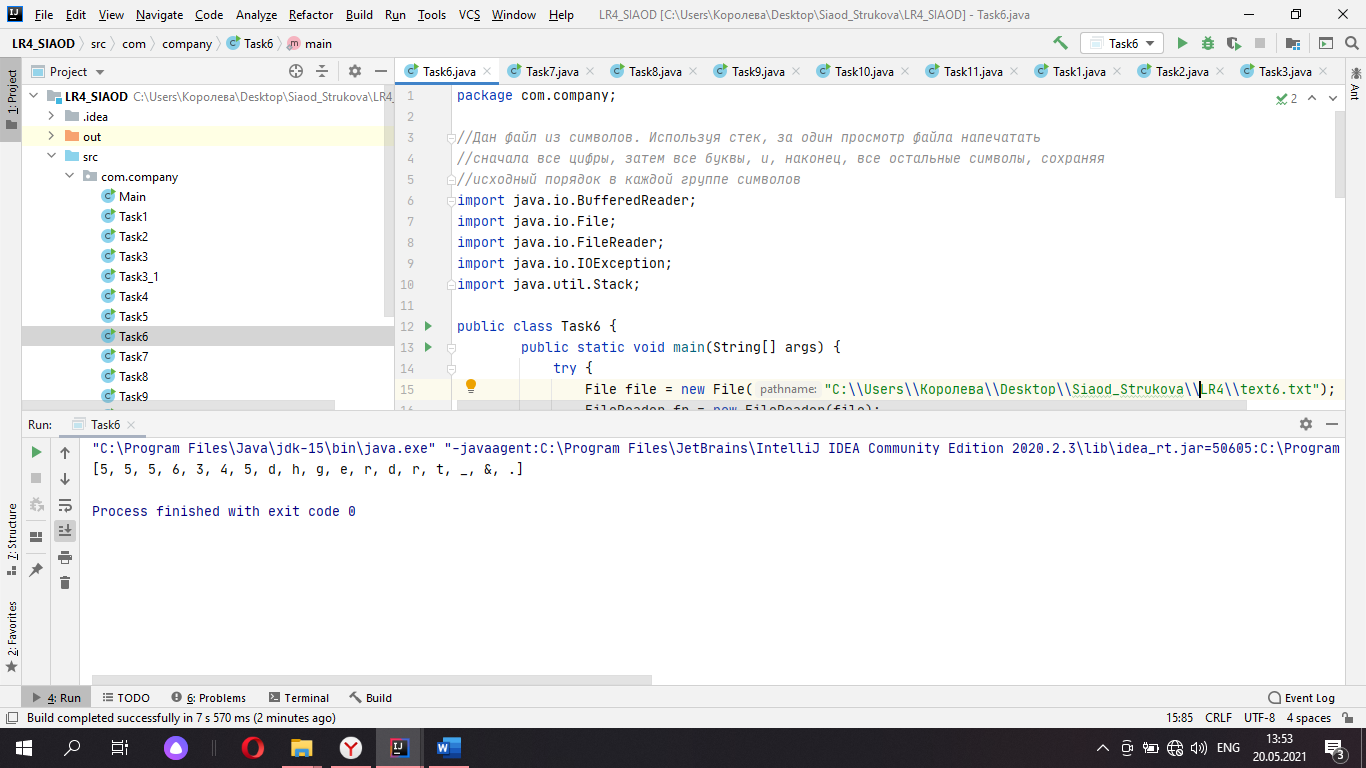


Рисунок 6 - Task 6

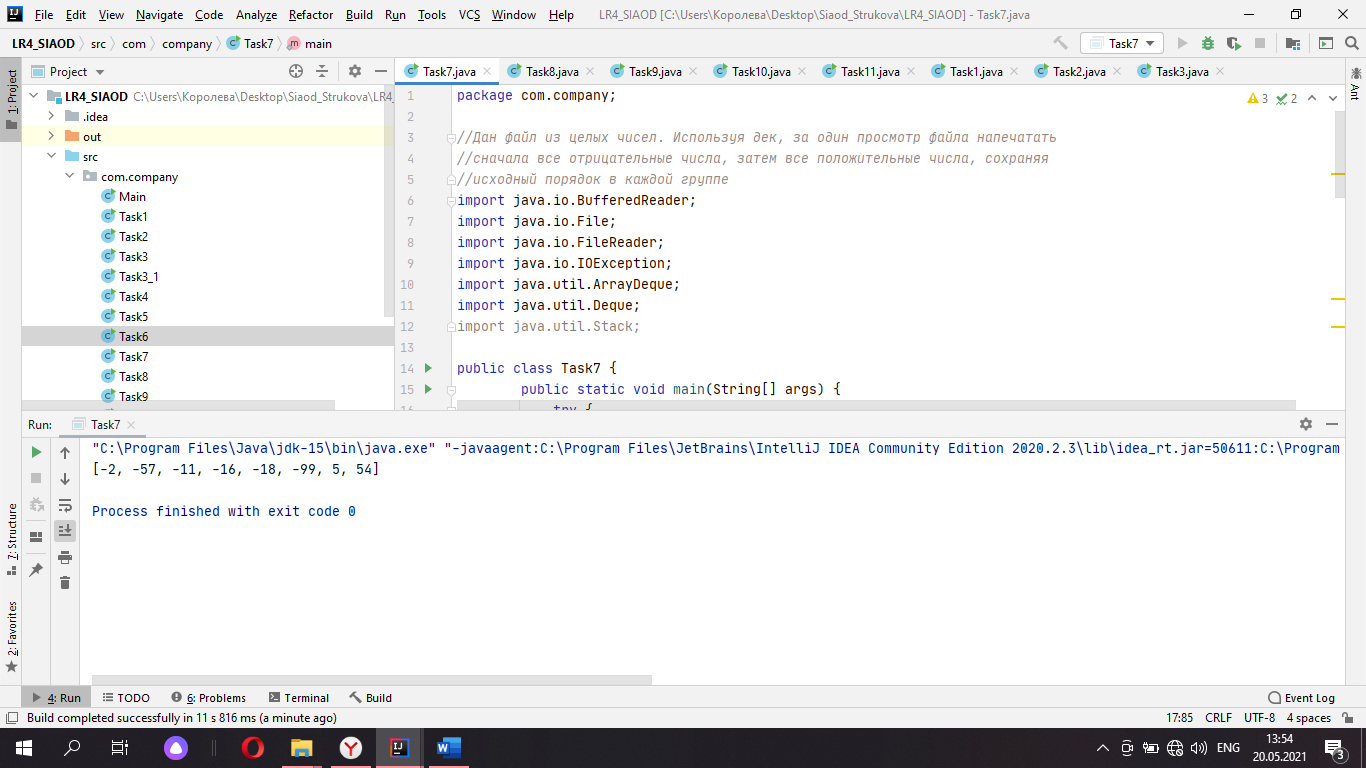


Рисунок 7 - Task 7

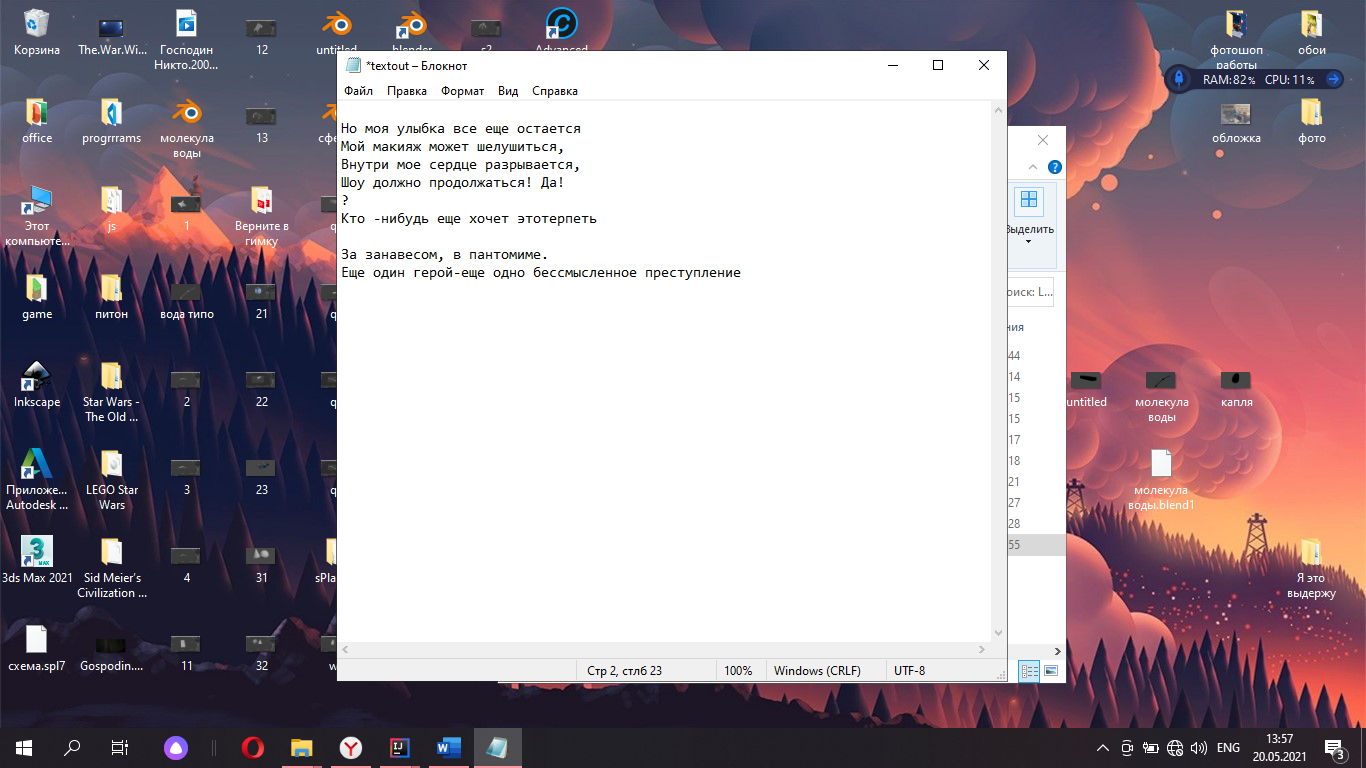


Рисунок 8 - Task 8

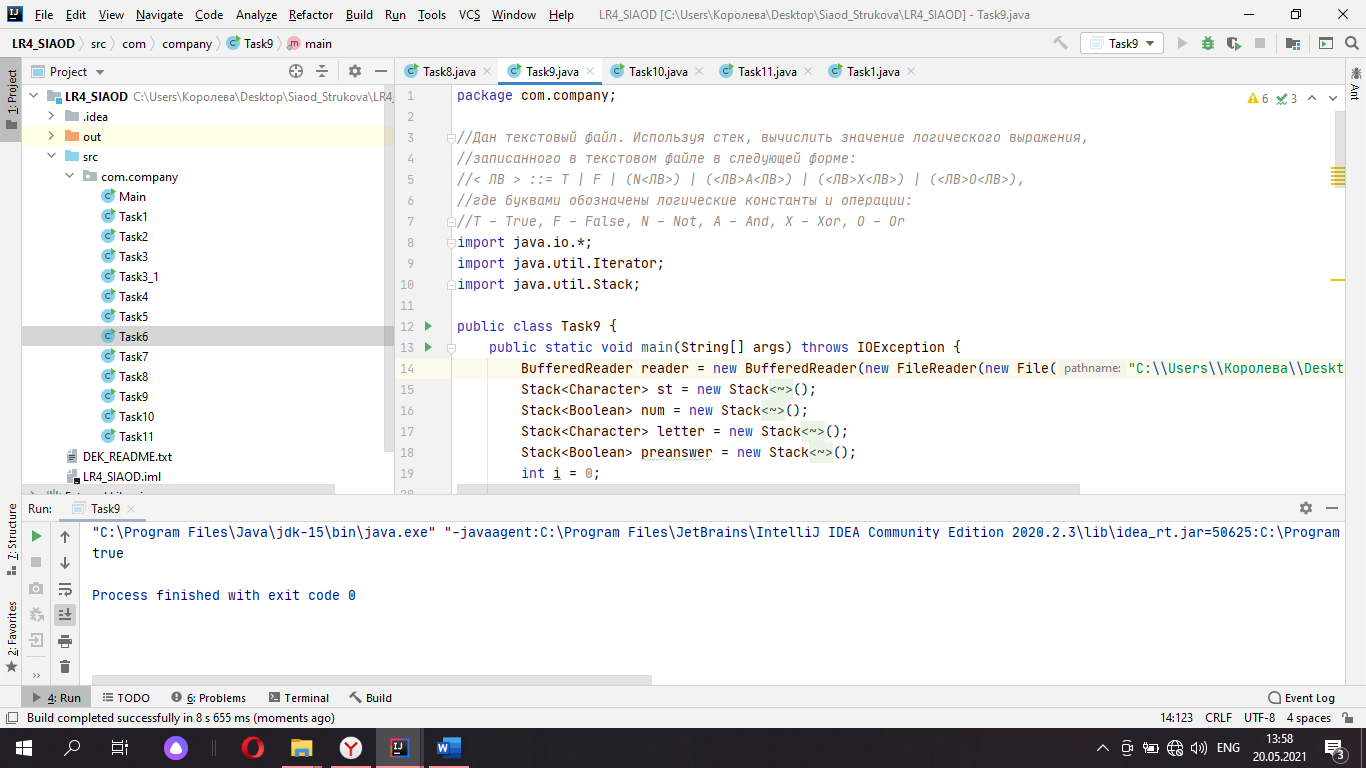
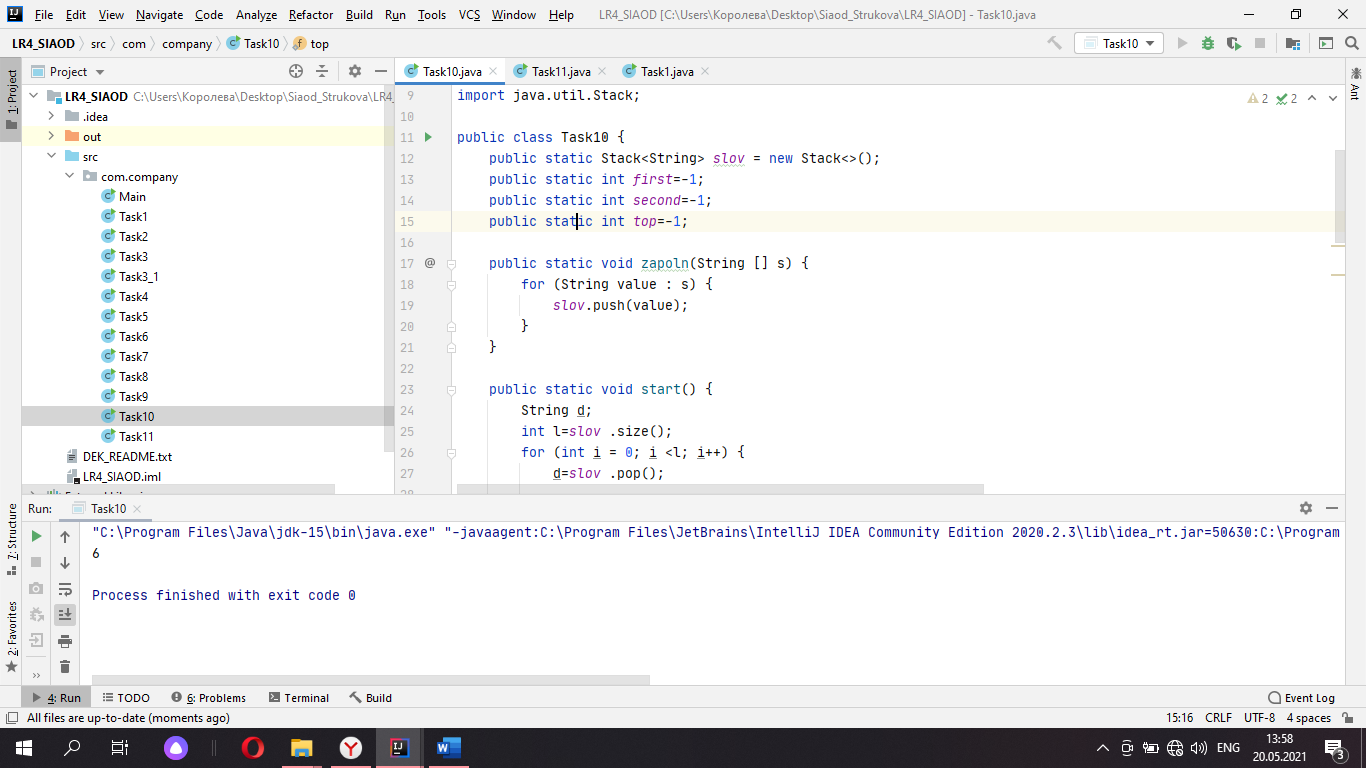


Рисунок 9 - Task 9



**Рисунок 10 - Task 10**

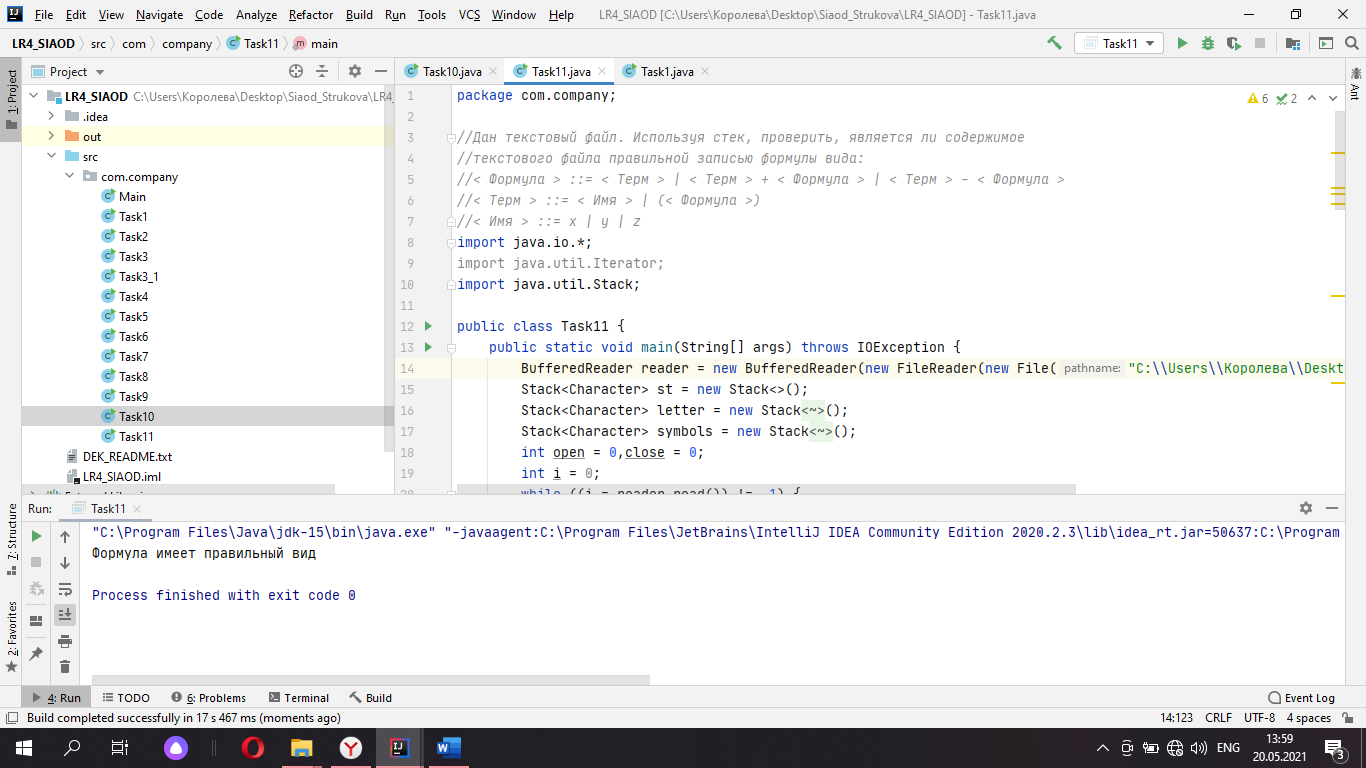


Рисунок 11 - Task 11

**Вывод:** мы научились реализовывать следующие структуры данных: стек (stack) и дек (двусторонняя очередь, deque).

# **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