Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации Ордена Трудового Красного Знамени

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Отчет по лабораторной работе №4

на тему: «Реализация стека и дека»

Выполнил: студент группы БВТ1903

Белов Сергей Павлович

Проверил:

Павликов Артём Евгеньевич

# **Задание №1**

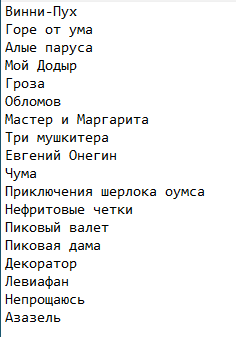
Задача:

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

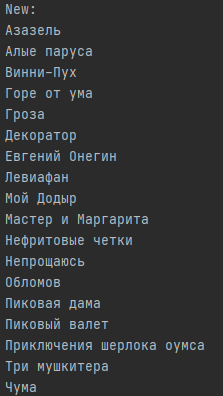
Листинги программ:

public class Ex1 {  
 public static <ex> void main (String[] arg){  
 //создаем два дека  
 int count=0;  
 Deque <String> first =new LinkedList<String>();  
 Deque <String> second =new LinkedList<String>();  
 //создаеи ридер файла  
 System.*out*.println("Text:");  
 try (FileReader reader = new FileReader("#1.txt");  
 BufferedReader br = new BufferedReader(reader)) {  
 // read line by line  
 String line;  
 while ((line = br.readLine()) != null) {  
 System.*out*.println(line);  
 //  
 first.addLast(line);//записываем строки в дек с записью в зад!!!  
 // }else {  
 // second.addLast(line);//записываем строки в дек с записью в зад!!!  
 // }  
 count++;  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
  
 //основная работа:  
  
 for (int i=0;i<count\*5;i++){  
 if (i%2==0){//Четное  
 String line = first.pollLast();//переносим первую книгу с первого на второй  
 second.add(line);  
 while (first.isEmpty() != true) {  
 if (*Coast*((first.peekLast()).charAt(0)) < *Coast*((second.peekLast()).charAt(0))) {//если текущее (1) меньше или равно первого (2)  
 line = first.pollFirst();//переносим с первого на второй в начало  
 second.addFirst(line);  
 } else {//если текущее (1) больше первого (2)  
 line = first.pollLast();//переносим с первого на второй в конец  
 second.addLast(line);  
 }  
 }  
 //перезаписываем все обратно на первый  
 while (second.isEmpty() != true) {  
 line = second.pollFirst();//переносим с первого на второй в конец  
 first.addLast(line);  
 }  
 }else {//НЕ Четное  
 String line = first.pollFirst();//переносим с первого на второй  
 second.add(line);  
 while (first.isEmpty() != true) {  
 if (*Coast*((first.peekFirst()).charAt(0)) <= *Coast*((second.peekFirst()).charAt(0))) {//если текущее (1) меньше или равно первого (2)  
 line = first.pollFirst();//переносим с первого на второй в начало  
 second.addFirst(line);  
 } else {//если текущее (1) больше первого (2)  
 line = first.pollFirst();//переносим с первого на второй в конец  
 second.addLast(line);  
 }  
 }  
 //перезаписываем все обратно на первый  
 while (second.isEmpty() != true) {  
 line = second.pollFirst();//переносим с первого на второй в конец  
 first.addLast(line);  
 }  
 }  
 }  
 //VIVOD  
 System.*out*.println();  
 System.*out*.println();  
 System.*out*.println("New:");  
 while (first.isEmpty()!=true) {//пока первый не пустой  
 String line = first.pollFirst();//переносим с первого на второй в обратном порядке  
 System.*out*.println(line);  
 }  
  
  
 }  
 public static int Counter (Deque <String> first1){  
 */\*\*принцип работы:  
 \* номер шага 1 2 3  
 \* первый ВГАМ ГАМ АМ  
 \* второй \*пусто\* В ГВ  
 \* сравниваем сравниваем  
 \* первые первые  
 \* Г больше В -> они по порядку  
 \* А меьше Г -> они не по порядку  
 \*/* int count = 0;  
 Deque <String> second2 =new LinkedList<String>();  
 while (first1.isEmpty()!=true){//пока первый не пустой  
 String line = first1.pollFirst();//переносим с первого на второй в обратном порядке  
 second2.add(line);  
 if(first1.isEmpty()!=true){  
 if(*Coast*((first1.peekFirst()).charAt(0))>*Coast*((second2.peekFirst()).charAt(0)))//проверяем, первые буквы  
 count++;  
 }  
 }  
 return count;  
 }  
 public static int Coast(char ch) {//возвращает стоимость буквы  
 char[] arr = {'А', 'Б', 'В', 'Г', 'Д', 'Е', 'Ё', 'Ж', 'З', 'И', 'Й', 'К', 'Л', 'М', 'Н', 'О', 'П', 'Р', 'С', 'Т', 'У', 'Ф', 'Х', 'Ц', 'Ч', 'Ш', 'Щ', 'Э', 'Ю', 'Я'};  
 int g=0;  
 for (int i = 0; i < arr.length; i++) {  
 if (ch == arr[i]){  
 g=i;  
 break;  
 }  
 }  
 return g;  
 }  
 public static Deque <String> SORT (Deque <String> deque){  
 String str = deque.pollFirst();  
 if (*Coast*(str.charAt(0)) <= *Coast*((deque.peekFirst()).charAt(0))) {//если текущее (1) меньше или равно первого (2)  
 deque.addFirst(str);  
 } else {//если текущее (1) больше первого (2)  
 deque.addLast(str);  
 }  
 return deque;  
 }  
}

Входные данные:



Результаты работы программы:



# **Задание №2**

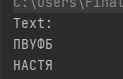
Задача:

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.

Листинг программы:

public class Ex2 {  
 //создадим дек, хранящий алфавит  
 //так как текстовый файл зашифрован буква алфавита+2  
 static Deque<String> *first* =new LinkedList<String>();  
 public static void Code (){  
 char[] arr = {'А', 'Б', 'В', 'Г', 'Д', 'Е', 'Ё', 'Ж', 'З', 'И', 'Й', 'К', 'Л', 'М', 'Н', 'О', 'П', 'Р', 'С', 'Т', 'У', 'Ф', 'Х', 'Ц', 'Ч', 'Ш', 'Щ','Ъ','Ы','Ь', 'Э', 'Ю', 'Я'};  
 for (int i=0;i< arr.length;i++)  
 *first*.addLast(String.*valueOf*(arr[i]));  
 }  
 static String *str* = "";  
 public static void main (String [] arg){  
 *Code*();//активируем дек  
 //прочтаем файл и запишем его в строку  
 System.*out*.println("Text:");  
 try (FileReader reader = new FileReader("#2.txt");  
 BufferedReader br = new BufferedReader(reader)) {  
 // read line by line  
 String len;  
 while ((len = br.readLine()) != null) {  
 System.*out*.println(len);  
 *str*=*str* + len;  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 for (int i=0;i<*str*.length();i++){  
 if (*str*.charAt(i)==' ')continue;  
 int char\_coast = *Coast*(*str*.charAt(i));  
 while (*Coast*(*first*.peekFirst().charAt(0))!=char\_coast){  
 *One*();  
 }  
 *TwoFirst*();  
 *str*=*Replace*(i);  
 }  
 System.*out*.println(*str*);  
 }  
 //Напишем метод, перекручивающий дек на два символа алфавита назад, для постановки правильной буквы С начала  
 public static void TwoFirst(){  
 String s1 = *first*.pollLast();  
 *first*.addFirst(s1);  
 String s2=*first*.pollLast();  
 *first*.addFirst(s2);  
 }  
 //Напишем метод, перекручивающий дек на два символа алфавита назад, для постановки правильной буквы С конца  
 public static void TwoLast(){  
 String s1 = *first*.pollFirst();  
 *first*.addLast(s1);  
 String s2 = *first*.pollFirst();  
 *first*.addLast(s2);  
 }  
 //Напишем метод, перекручивающий дек на один символа алфавита вперед, для поиска нужной буквы  
 public static void One(){  
 String s1 = *first*.pollFirst();  
 *first*.addLast(s1);  
 }  
 public static int Coast(char ch) {//возвращает стоимость буквы  
 char[] arr = {'А', 'Б', 'В', 'Г', 'Д', 'Е', 'Ё', 'Ж', 'З', 'И', 'Й', 'К', 'Л', 'М', 'Н', 'О', 'П', 'Р', 'С', 'Т', 'У', 'Ф', 'Х', 'Ц', 'Ч', 'Ш', 'Щ','Ъ','Ы','Ь','Э', 'Ю', 'Я'};  
 int g=0;  
 for (int i = 0; i < arr.length; i++) {  
 if (ch == arr[i]){  
 g=i;  
 break;  
 }  
 }  
 return g;  
 }  
 public static String Replace (int i){  
 String temp1="";  
 for (int j=0;j<i;j++){  
 temp1+=*str*.charAt(j);  
 }  
 String temp2="";  
 for (int j=i+1;j<*str*.length();j++){  
 temp2+=*str*.charAt(j);  
 }  
 String temp3 = temp1+*first*.peekFirst()+temp2;  
 return temp3;  
 }  
}

Результат работы программы:



# **Задание №3**

Задача:

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;

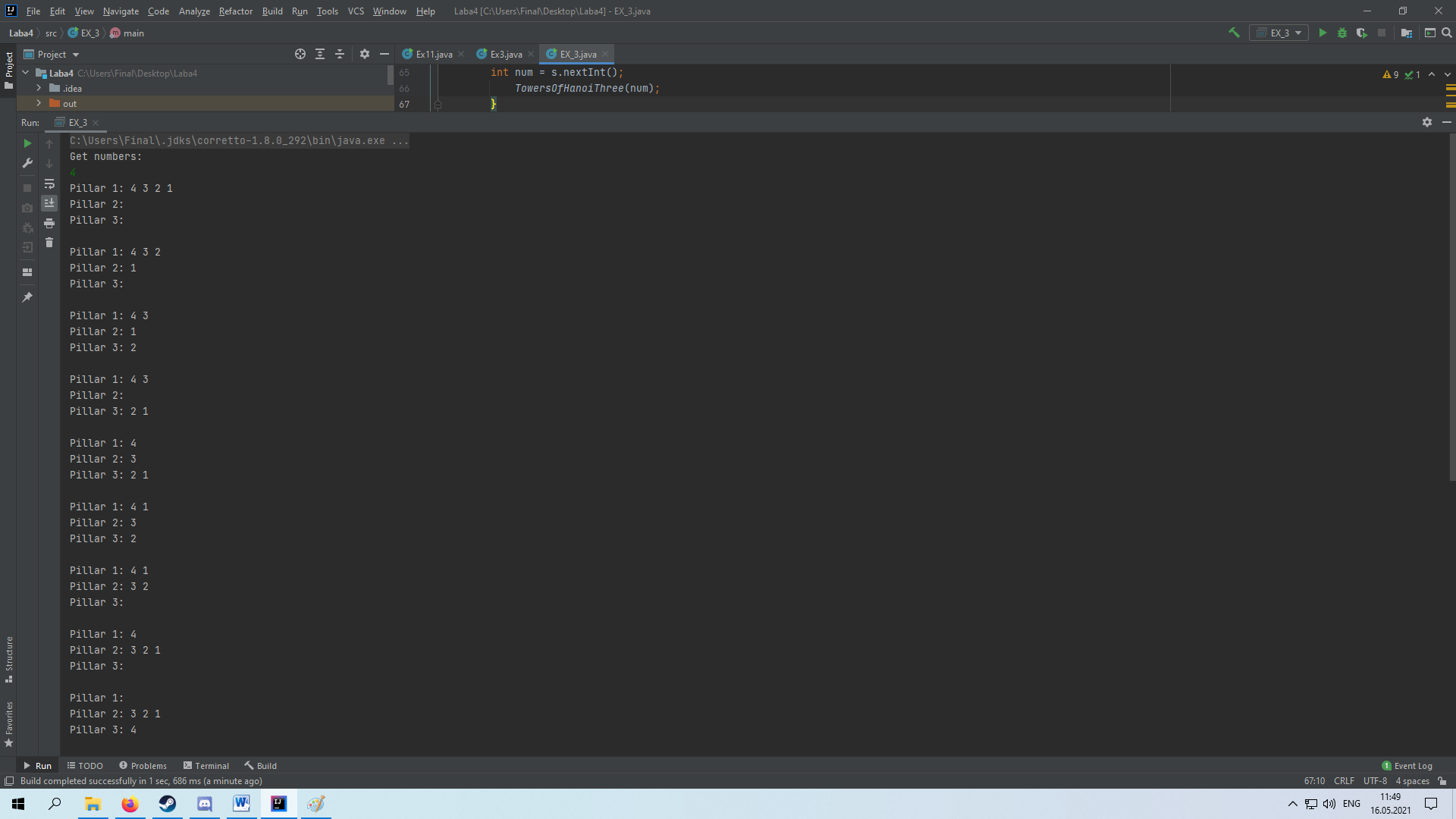
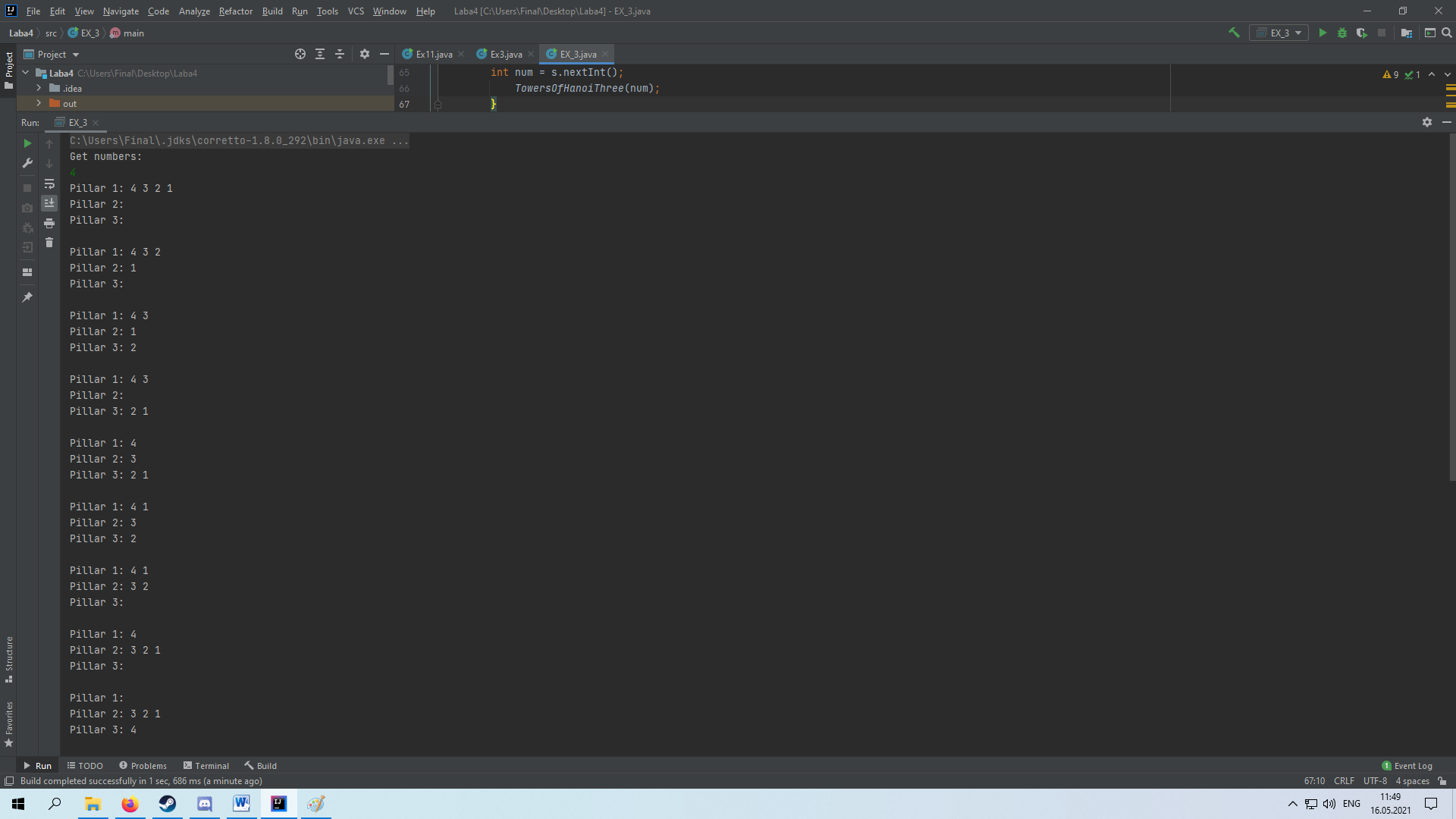
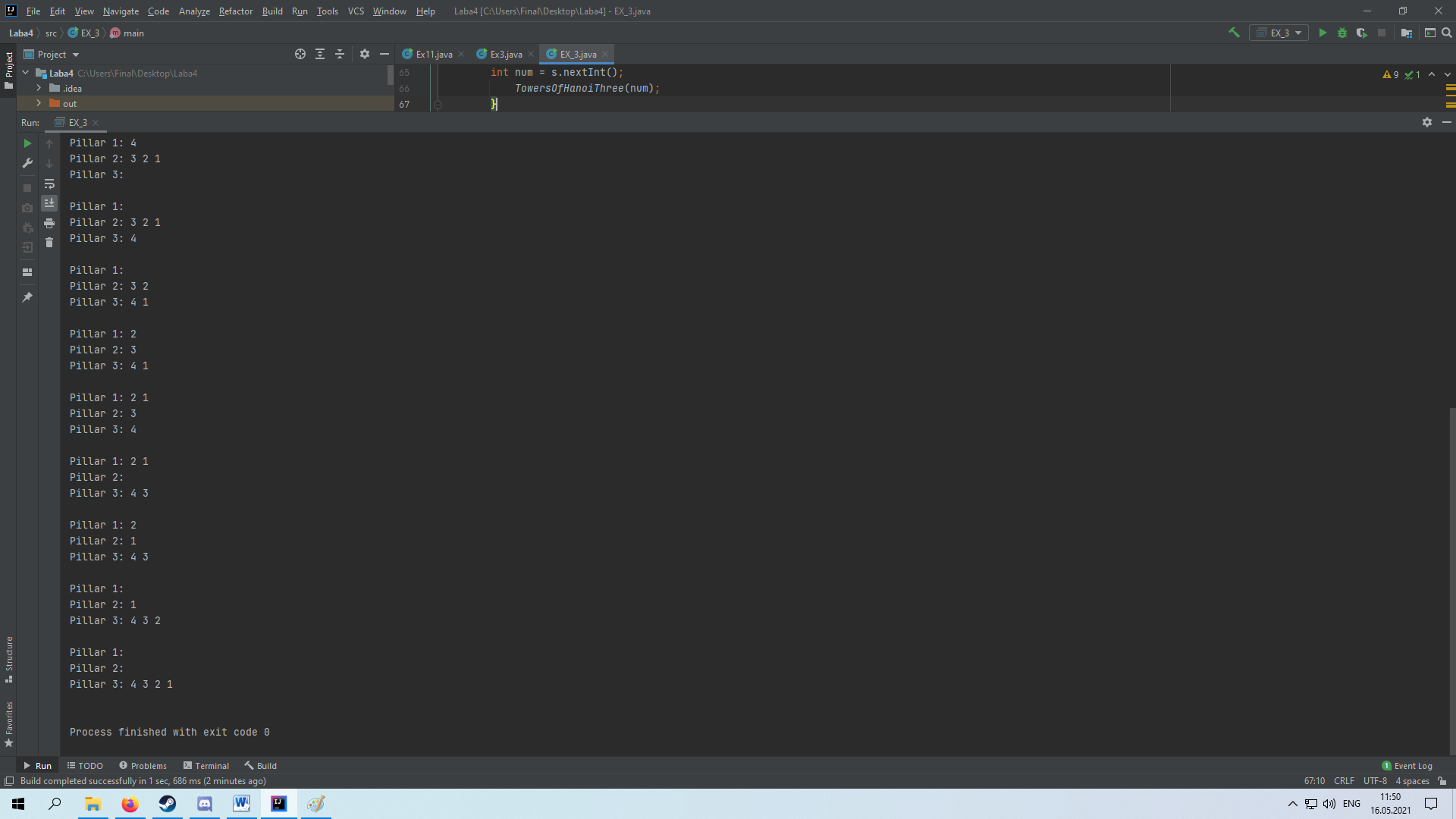
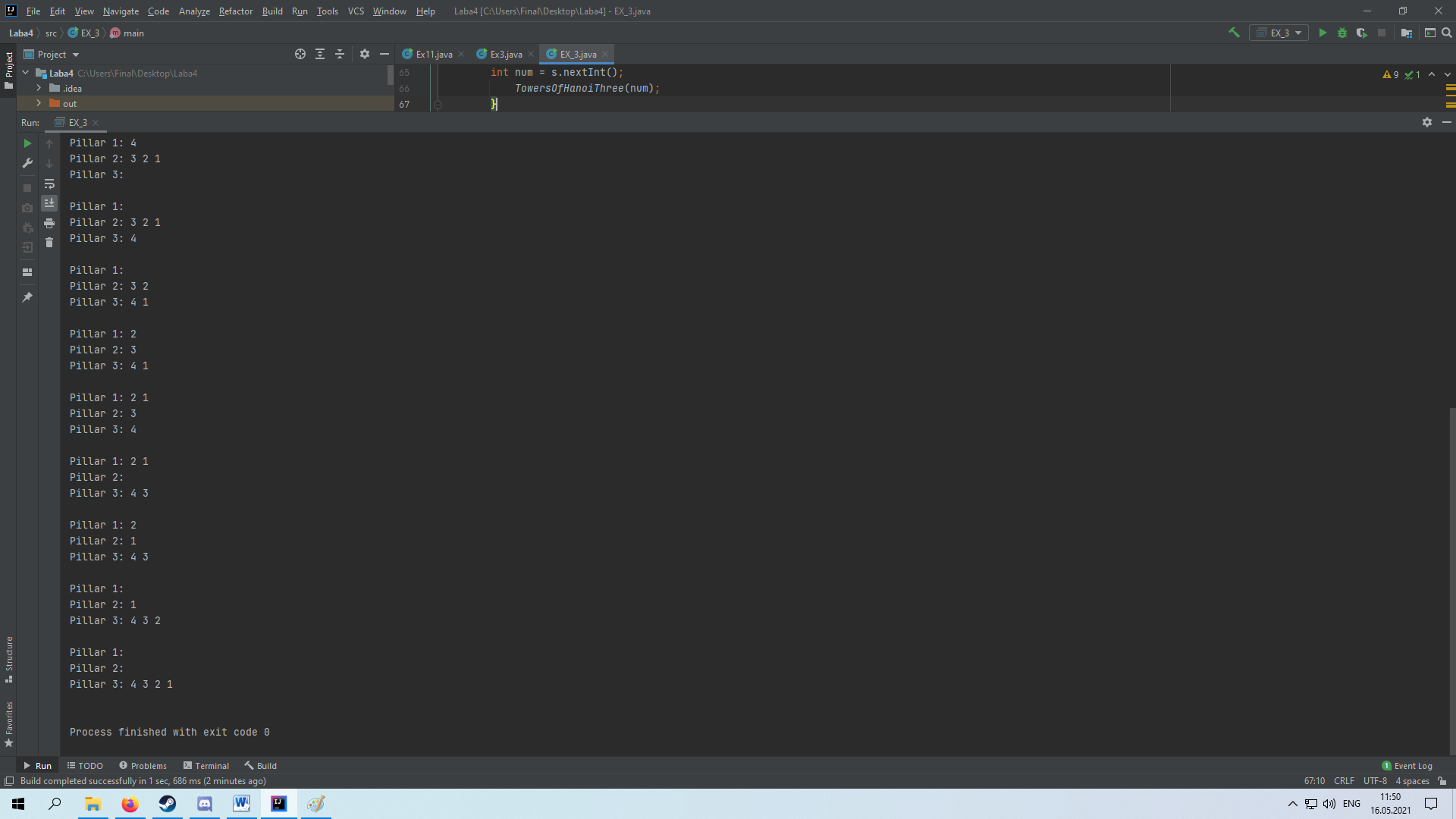
- для промежуточного хранения можно использовать стержень В.

Реализовать алгоритм, используя три стека вместо стержней А, В, С. Информация о дисках хранится в исходном файле.

Листинг программы:

public class Ex3 {  
 //создаем три стека  
 static Stack<Integer> *A* = new Stack<>();  
 static Stack<Integer> *B* = new Stack<>();  
 static Stack<Integer> *C* = new Stack<>();  
 static int *count*=0;  
 public static <ex> void main(String[] arg) {  
 System.*out*.println("Rings number?");  
 Scanner s = new Scanner(System.*in*);  
 *count* = s.nextInt();  
 for (int i = *count*; i > 0; i--) {  
 *A*.push(i);  
 }  
 *Honoi*(*count*, *A*, *B*, *C*);  
 }  
 public static int Size(Stack<Integer> S) {  
 int counter = 0;  
 while (S.isEmpty() != true) {  
 S.pop();  
 counter++;  
 }  
 return counter;  
 }  
 public static void Honoi(int count\_H, Stack<Integer> A\_H, Stack<Integer> B\_H, Stack<Integer> C\_H) {  
 if (count\_H == 1) {  
 C\_H.push(A\_H.pop());  
 *Print*(A\_H,B\_H,C\_H);  
 } else {  
 *Honoi*(count\_H - 1, A\_H, C\_H, B\_H);  
 *Print*(A\_H, B\_H, C\_H);  
 *Honoi*(count\_H - 1, B\_H, A\_H, C\_H);  
 }  
 }  
 public static void Print(Stack<Integer> A\_T, Stack<Integer> B\_T, Stack<Integer> C\_T) {  
 String Pillar1 = "Pillar1: ";  
 String Pillar2 = "Pillar2: ";  
 String Pillar3 = "Pillar3: ";  
 String emptyStr1 = "";  
 String emptyStr2 = "";  
 String emptyStr3 = "";  
 for (int x = 0; x <= *count* - 1; x++) {  
 try{  
 emptyStr1 += String.*valueOf*(A\_T.get(x)) + " ";  
 } catch (Exception e) {  
 }  
 try{  
 emptyStr2 += String.*valueOf*(B\_T.get(x)) + " ";  
 } catch (Exception e) {  
 }  
 try {  
 emptyStr3 += String.*valueOf*(C\_T.get(x)) + " ";  
 } catch (Exception e) {  
 }  
 }  
 System.*out*.println(Pillar1 + emptyStr1);  
 System.*out*.println(Pillar2 + emptyStr2);  
 System.*out*.println(Pillar3 + emptyStr3);  
 System.*out*.println();  
 }  
}

Результат работы программы:

# **Задание №4**

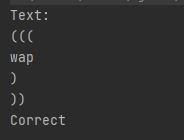
Задача:

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

Листинг программы:

public class Ex4 {  
 public static <ex> void main(String[] arg) {  
 //создаем стек  
 Stack<String> first = new Stack<String>();  
 System.*out*.println("Text:");  
 try (FileReader reader = new FileReader("#4.txt");  
 BufferedReader br = new BufferedReader(reader)) {  
 String line;  
 while ((line = br.readLine()) != null) {  
 System.*out*.println(line);  
 char[] text = line.toCharArray();  
 for (int i=0; i<text.length; i++){  
 if (text [i] == '(' ){  
 first.push(line);//записываем строки в стек с записью наверх!!!  
 }  
 if (text [i]== ')'){  
 if (first.isEmpty()== true){  
 System.*out*.println("Wrong");  
 break;  
 }  
 first.pop();  
 }  
 }  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 if (first.isEmpty() == true){  
 System.*out*.println("Correct");  
 }  
 else {  
 System.*out*.println("Wrong");  
 }  
 }  
}

Результат работы программы:



# **Задание №5**

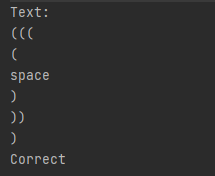
Задача:

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

Листинг программы:

public class Ex5 {  
 public static <ex> void main(String[] arg) {  
 //создаем стек  
 Deque <String> first =new LinkedList<String>();  
 //создаеи ридер файла  
 System.*out*.println("Text:");  
 try (FileReader reader = new FileReader("#5.txt");  
 BufferedReader br = new BufferedReader(reader)) {  
 // read line by line  
 String line;  
  
 while ((line = br.readLine()) != null) {  
 System.*out*.println(line);  
 char[] text = line.toCharArray();  
 for (int i=0; i<text.length; i++){  
 if (text [i] == '(' ){  
 first.push(line);//записываем строки в стек с записью наверх!!!  
 }  
 if (text [i]== ')'){  
 if (first.isEmpty()== true){  
 System.*out*.println("Wrong");  
 break;  
 }  
 first.pop();  
 }  
 }  
  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 if (first.isEmpty() == true){  
 System.*out*.println("Correct");  
 }  
 else {  
 System.*out*.println("Wrong");  
 }  
 }  
}

Результат работы программы:



# **Задание №6**

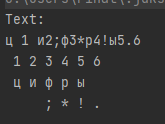
Задача:

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.

Листинг программы:

public class Ex6 {  
 public static <ex> void main (String[] arg) {  
 Stack<Integer> digits = new Stack<>();  
 Stack<String> letters = new Stack<>();  
 Stack<String> others = new Stack<>();  
 System.*out*.println("Text:");  
 try (FileReader reader = new FileReader("#6.txt");  
 BufferedReader br = new BufferedReader(reader)) {  
 // read line by line  
 String line;  
 while ((line = br.readLine()) != null) {  
 System.*out*.println(line);  
 for (int i = 0; i < line.length(); i++) {  
 if (line.charAt(i) >= '0' && line.charAt(i) <= '9') {  
 digits.push(Integer.*valueOf*(String.*valueOf*(line.charAt(i))));  
 } else if (line.charAt(i) >= 'A' && line.charAt(i) <= 'Z' || line.charAt(i) >= 'a' && line.charAt(i) <= 'z' || line.charAt(i) >= 'А' && line.charAt(i) <= 'Я' || line.charAt(i) >= 'а' && line.charAt(i) <= 'я') {  
 letters.push(String.*valueOf*(line.charAt(i)));  
 } else {  
 others.push(String.*valueOf*(line.charAt(i)));  
 }  
 }  
  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
  
 }  
 String s1 = " ";  
 String s2 = " ";  
 String s3 = " ";  
 Stack<Integer> temp1 = new Stack<>();  
 while (digits.isEmpty()!=true) temp1.push(digits.pop());  
 while (temp1.isEmpty()!=true)s1 = s1 + String.*valueOf*(temp1.pop()) + " ";  
 Stack<String> temp2 = new Stack<>();  
 while (letters.isEmpty()!=true) temp2.push(letters.pop());  
 while (temp2.isEmpty()!=true) s2 = s2 + String.*valueOf*(temp2.pop()) + " ";  
 Stack<String> temp3 = new Stack<>();  
 while (others.isEmpty()!=true) temp3.push(others.pop());  
 while (temp3.isEmpty()!=true) s3 = s3 + String.*valueOf*(temp3.pop()) + " ";  
 System.*out*.println(s1);  
 System.*out*.println(s2);  
 System.*out*.println(s3);  
 }  
}

Результат работы программы:



# **Задание №7**

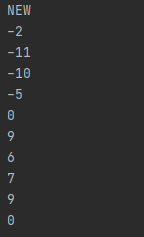
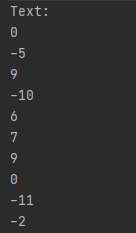
Задача:

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.

Листинг программы:

public class Ex7 {  
 public static <ex> void main (String[] arg) {  
 Deque<Integer> first = new LinkedList<>();  
 System.*out*.println("Text:");  
 try (FileReader reader = new FileReader("#7.txt");  
 BufferedReader br = new BufferedReader(reader)) {  
 String line;  
 while ((line = br.readLine()) != null) {  
 System.*out*.println(line);  
 int num = Integer.*valueOf*(line);  
 if(num>=0) first.addLast(num);  
 else first.addFirst(num);  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 System.*out*.println("NEW");  
 while (first.isEmpty()!=true){  
 System.*out*.println(first.pollFirst()+" ");  
 }  
 }  
}

Результат работы программы:



# **Задание №8**

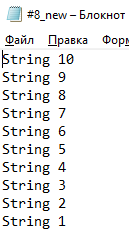
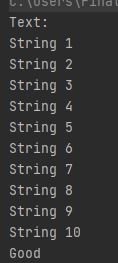
Задача:

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.

Листинг программы:

public class Ex8 {  
 public static <ex> void main (String[] arg) {  
 Stack<String> stack = new Stack<>();  
 System.*out*.println("Text:");  
 try (FileReader reader = new FileReader("#8.txt");  
 BufferedReader br = new BufferedReader(reader)) {  
 String line;  
 while ((line = br.readLine()) != null) {  
 System.*out*.println(line);  
 stack.push(line);  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 *Create\_File*();  
 try{  
 FileWriter writer = new FileWriter("#8\_new.txt",true);  
 while (stack.isEmpty()!=true) {  
 writer.write(stack.pop()+'\n');  
 }  
 writer.flush();  
 }catch (IOException ex){  
 System.*out*.println(ex.getMessage());  
 }  
 }  
 public static void Create\_File(){  
 try{  
 File file = new File ("C:\\Users\\lapte\\OneDrive\\Desktop\\Laba4\\#8\_new.txt");  
 if (file.createNewFile())  
 System.*out*.println("Good");  
 else System.*out*.println("Error");  
 }catch (Exception e){  
 System.*err*.println(e);}}  
}

Результат работы программы:



# **Задание №9**

Задача:

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.

Листинг программы:

public class Ex9 {  
 public static <ex> void main (String[] arg) throws ScriptException {  
 String string = "";  
 //создаеи ридер файла  
 System.*out*.println("Text:");  
 try (FileReader reader = new FileReader("#9.txt");  
 BufferedReader br = new BufferedReader(reader)) {  
 // read line by line  
 String line;  
 while ((line = br.readLine()) != null) {  
 System.*out*.println(line);  
 string+=line;  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 new Ex9(string);  
 }  
 public Ex9 (String Str) throws ScriptException {  
 ScriptEngineManager manager = new ScriptEngineManager();  
 ScriptEngine engine = manager.getEngineByName("JavaScript");  
 StringBuilder str1 = new StringBuilder();  
 Stack stk= new Stack();  
 Stack stk2 = new Stack();  
 for (int i = 0; i < Str.length(); i++) {  
 stk.add(Str.charAt(i));  
 }  
 for (int i = 0; i < Str.length(); i++) {  
 stk2.add(stk.pop());  
 }  
 for (int i = 0; i < Str.length(); i++) {  
 if (*Compare*(stk2.peek().toString(), "T"))  
 str1.append("true ");  
 if (*Compare*(stk2.peek().toString(), "F"))  
 str1.append("false ");  
 if (*Compare*(stk2.peek().toString(), "N"))  
 str1.append("! ");  
 if (*Compare*(stk2.peek().toString(), "A"))  
 str1.append("&& ");  
 if (*Compare*(stk2.peek().toString(), "X"))  
 str1.append("!= ");  
 if (*Compare*(stk2.peek().toString(), "O"))  
 str1.append("|| ");  
 if (*Compare*(stk2.peek().toString(), "("))  
 str1.append("( ");  
 if (*Compare*(stk2.peek().toString(), ")"))  
 str1.append(")");  
 stk2.pop();  
 }  
  
 Object o;  
 o = engine.eval(str1.toString());  
 System.*out*.println(o);  
 }  
 public static boolean Compare (String str1,String str2){//сравниваем строки  
 if(str1.charAt(0)==str2.charAt(0))  
 return true;  
 return false;  
 }  
 public static boolean Calc(String input) throws ScriptException {  
 ScriptEngineManager mgr = new ScriptEngineManager();  
 ScriptEngine engine = mgr.getEngineByName("JavaScript");  
 return (boolean)engine.eval(input);  
 }  
}

Результат работы программы:



# **Задание №10**

Задача:

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:

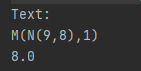
M – определение максимума, N – определение минимума.

Используя стек, вычислить значение заданного выражения.

Листинг программы:

public class Ex10 {  
 public static <ex> void main (String[] arg) throws ScriptException {  
 String string = "";  
 //создаеи ридер файла  
 System.*out*.println("Text:");  
 try (FileReader reader = new FileReader("#10.txt");  
 BufferedReader br = new BufferedReader(reader)) {  
 // read line by line  
 String line;  
 while ((line = br.readLine()) != null) {  
 System.*out*.println(line);  
 string+=line;  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 new Ex10(string);  
 }  
 public Ex10(String Str) throws ScriptException {  
 ScriptEngineManager manager = new ScriptEngineManager();  
 ScriptEngine engine = manager.getEngineByName("JavaScript");  
 StringBuilder str1 = new StringBuilder();  
 Stack stk = new Stack();  
 Stack stk2 = new Stack();  
 for (int i = 0; i < Str.length(); i++) {  
 stk.add(Str.charAt(i));  
 }  
 for (int i = 0; i < Str.length(); i++) {  
 stk2.add(stk.pop());  
 }  
 for (int i = 0; i < Str.length(); i++) {  
 if (*cpv*(stk2.peek().toString(), "0"))  
 str1.append("0");  
 if (*cpv*(stk2.peek().toString(), "1"))  
 str1.append("1");  
 if (*cpv*(stk2.peek().toString(), "2"))  
 str1.append("2");  
 if (*cpv*(stk2.peek().toString(), "3"))  
 str1.append("3");  
 if (*cpv*(stk2.peek().toString(), "4"))  
 str1.append("4");  
 if (*cpv*(stk2.peek().toString(), "5"))  
 str1.append("5");  
 if (*cpv*(stk2.peek().toString(), "6"))  
 str1.append("6");  
 if (*cpv*(stk2.peek().toString(), "7"))  
 str1.append("7");  
 if (*cpv*(stk2.peek().toString(), "8"))  
 str1.append("8");  
 if (*cpv*(stk2.peek().toString(), "9"))  
 str1.append("9");  
 if (*cpv*(stk2.peek().toString(), "M"))  
 str1.append("Math.max");  
 if (*cpv*(stk2.peek().toString(), "N"))  
 str1.append("Math.min");  
 if (*cpv*(stk2.peek().toString(), ","))  
 str1.append(",");  
 if (*cpv*(stk2.peek().toString(), "("))  
 str1.append("( ");  
 if (*cpv*(stk2.peek().toString(), ")"))  
 str1.append(")");  
 stk2.pop();  
 }  
  
 Object o = engine.eval(str1.toString());  
 System.*out*.println(o);  
 }  
 public static boolean cpv(String str1, String str2) {  
 boolean y1 = false;  
 if (str1.charAt(0) == str2.charAt(0))  
 y1 = true;  
 return y1;  
 }  
}

Результат работы программы:



# **Задание №11**

Задача:

Дан текстовый файл. Используя стек, проверить, является ли содержимое текстового файла правильной записью формулы вида:

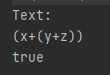
< Формула > ::= < Терм > | < Терм > + < Формула > | < Терм > - < Формула >

< Терм > ::= < Имя > | (< Формула >) < Имя > ::= x | y | z

Листинг программы:

public class Ex11 {  
 public static <ex> void main (String[] arg) throws ScriptException {  
 String string = "";  
 //создаеи ридер файла  
 System.*out*.println("Text:");  
 try (FileReader reader = new FileReader("#11.txt");  
 BufferedReader br = new BufferedReader(reader)) {  
 // read line by line  
 String line;  
 while ((line = br.readLine()) != null) {  
 System.*out*.println(line);  
 string+=line;  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 new Ex11(string);//вызываем метод  
 }  
  
  
 public Ex11(String Str) throws ScriptException {  
 ScriptEngineManager manager = new ScriptEngineManager();  
 ScriptEngine engine = manager.getEngineByName("JavaScript");  
 //Object o =engine.eval(code);  
 StringBuilder str1 = new StringBuilder();  
 Stack stk = new Stack();  
 Stack stk2 = new Stack();  
 for (int i = 0; i < Str.length(); i++) {  
 stk.add(Str.charAt(i));  
 }  
 for (int i = 0; i < Str.length(); i++) {  
 stk2.add(stk.pop());  
 }  
 for (int i = 0; i < Str.length(); i++) {  
 if (*Compare*(stk2.peek().toString(), "x"))  
 str1.append("1");  
 if (*Compare*(stk2.peek().toString(), "y"))  
 str1.append("2");  
 if (*Compare*(stk2.peek().toString(), "z"))  
 str1.append("3");  
 if (*Compare*(stk2.peek().toString(), "("))  
 str1.append("( ");  
 if (*Compare*(stk2.peek().toString(), ")"))  
 str1.append(")");  
 if (*Compare*(stk2.peek().toString(), "+"))  
 str1.append("+");  
 if (*Compare*(stk2.peek().toString(), "-"))  
 str1.append("-");  
 stk2.pop();  
 }  
 try {  
 Object o = engine.eval(str1.toString());  
 System.*out*.println("true" );  
 } catch (Throwable e) {  
 System.*out*.println("false" );  
 }  
  
 }  
 public static boolean Compare (String str1,String str2){//сравниваем строки  
 if(str1.charAt(0)==str2.charAt(0))  
 return true;  
 return false;  
 }  
}

Результат работы программы:



# **Вывод**

В результате выполнения лабораторной работы я научился работать со стеками и деками, понял как работает ввод данных из файла и как с помощью программы создавать файлы и записывать в них текст.