

|  |  |
| --- | --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  федеральное государственное бюджетное образовательное учреждение высшего образования  **«МОСКОВСКИЙ АВТОМОБИЛЬНО-ДОРОЖНЫЙ ГОСУДАРСТВЕННЫЙ**  **ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ (МАДИ)»** |  |

**Кафедра «Высшая математика»**

**Лабораторная работа № 1**

по дисциплине

«Структуры и алгоритмы обработки данных»

на тему:

**«**Реализация стека/дека**»**

**Выполнил:**

Учебная группа: 1бПМ   
ФИО: Кузнецов А.-С. О.

Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Руководитель лабораторной работы:**

Должность: старший преподаватель

Звание: б/з

ФИО: Кутейников И. А.

Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_г.

Цель:

Разработать программу обработки данных, содержащихся в заранее подготовленном файле,

в соответствии с индивидуальным заданием. Программа должна включать модуль, содержащий

набор всех необходимых средств (типов, подпрограмм и т.д.) для решения поставленной задачи.

Порядок выполнения работы:

1) Получить у преподавателя индивидуальное задание.

2) Построить схему алгоритма решения задачи.

3) Использовать подпрограммы, реализующие полный набор операций

для этой структуры:

- допустимые операции для стека: инициализация, проверка на

пустоту, добавление нового элемента в начало, извлечение элемента из

начала;

- допустимые операции для дека: инициализация, проверка на пустоту,

добавление нового элемента в начало, добавление нового элемента в конец,

извлечение элемента из начала, извлечение элемента из конца.

4) Составить спецификации используемых подпрограмм.

5)Составить программу, включающую модуль обработки соответствующей динамической структуры.

6) Проверить и продемонстрировать преподавателю работу программы

на полном наборе тестов. Обеспечить одновременный показ в окнах на

экране содержимого входного и выходного файлов.

7) Оформить отчет о лабораторной работе.

Вариант 5

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

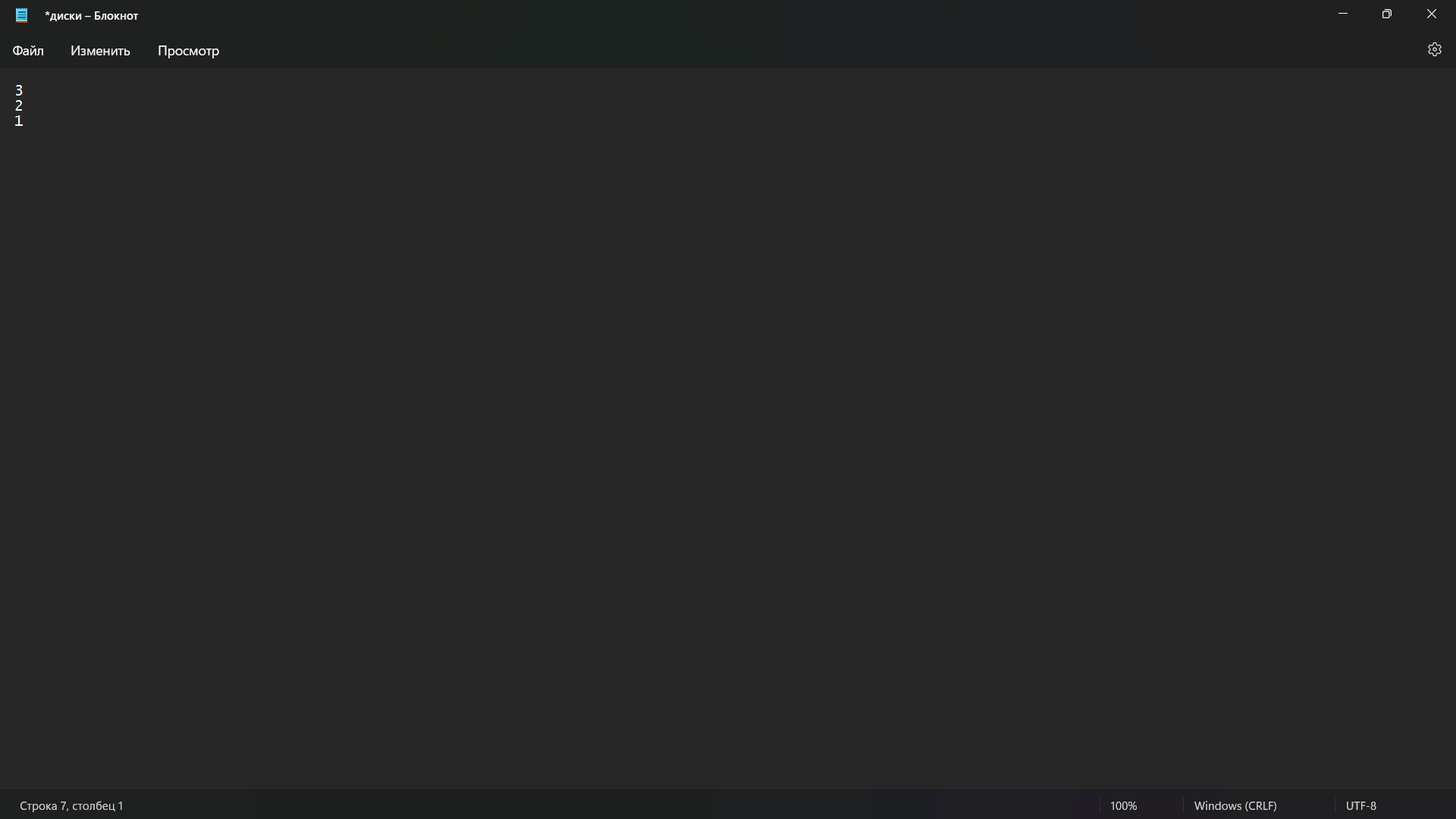
дисках хранится в исходном файле.

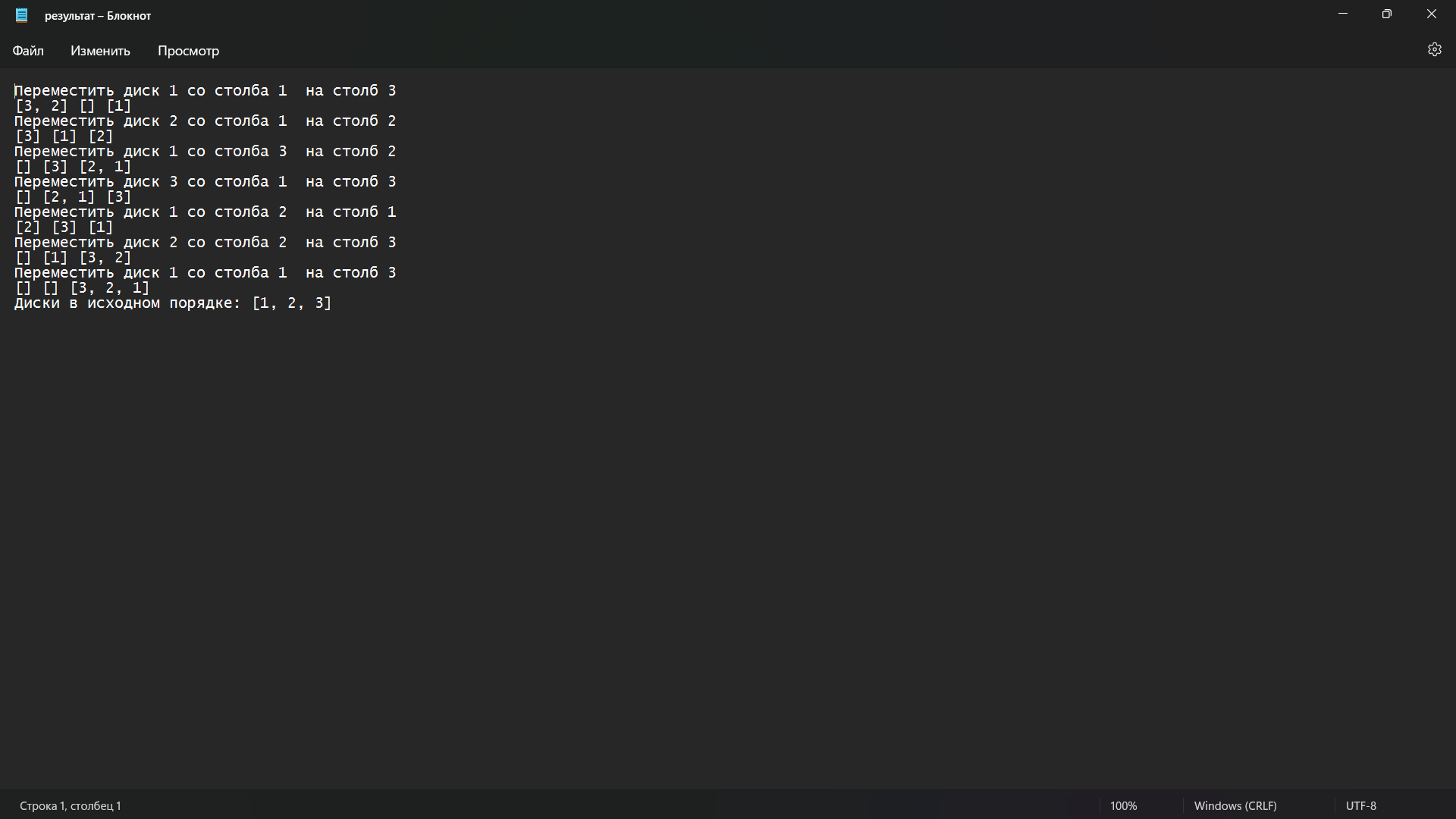
Код:

import java.io.File;import java.io.FileNotFoundException;  
import java.util.Scanner;import java.io.PrintWriter;  
import java.util.Stack;public class Main {  
 static void towerOfHanoi(int n, Stack<Integer> A, Stack<Integer> B, Stack<Integer> C, PrintWriter writer,String num1,String num2,String num3){  
 if(n > 0){  
 *towerOfHanoi*(n-1, A, C, B, writer,num1,num3,num2);  
 int disk = A.pop();  
 C.push(disk); writer.println("Переместить диск " + disk + " сo столба " +num1+" " + " на столб " +num3+" " + " ");  
 writer.println(A+" "+B+" "+C);  
 *towerOfHanoi*(n-1, B, A, C, writer,num2,num1,num3);  
 }  
 }  
 static void restoreOriginalOrder(Stack<Integer> source, Stack<Integer> destination, Stack<Integer> auxiliary) {  
 while (!destination.isEmpty()) {  
 source.push(destination.pop());  
 }  
 while (!auxiliary.isEmpty()) {  
 source.push(auxiliary.pop());  
 }  
 }  
 public static void main(String[] args) throws FileNotFoundException {  
 Stack<Integer> stackA = new Stack<>();  
 Stack<Integer> stackB = new Stack<>();  
 Stack<Integer> stackC = new Stack<>();  
 String num1="1";  
 String num2="2";  
 String num3="3";  
 Scanner sc= new Scanner(System.*in*);  
 File file = new File("диски.txt");  
 PrintWriter writer = new PrintWriter(new File("результат.txt"));  
 PrintWriter righter = new PrintWriter("диски.txt");  
 System.*out*.print("Введите число дисков: ");  
 int kolDis=sc.nextInt();  
 for(int i =kolDis;i>0;i--){  
 righter.println(i);  
 righter.flush();  
 }  
 Scanner in = new Scanner(file);  
 while (in.hasNextInt()) {  
 stackA.push(in.nextInt());  
 }  
 in.close();  
 *towerOfHanoi*(stackA.size(), stackA, stackB, stackC, writer,num1,num2,num3);  
 *restoreOriginalOrder*(stackA, stackC, stackB); // Восстанавливаем исходный порядок  
 writer.println("Диски в исходном порядке: " + stackA); writer.close();  
 }}

Результаты:

Введите число дисков: 3





Вывод:

В ходе лабораторной работы было получено понимание того, как работать со новой структурой, а именно

1. Стэком
2. Дэком

Также было получено понимание того, как оптимальнее создавать код.