

МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное
учреждение высшего
образования
НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий
Кафедра информатики и систем управления

ОТЧЕТ
по лабораторной работе
по дисциплине:
Алгоритмы и структуры данных

РУКОВОДИТЕЛЬ:
Капранов С.Н.

(подпись)
(фамилия, и.,о.)

СТУДЕНТ:
Еричев Д.А.

(подпись)
(фамилия, и.,о.)

(шифр группы)

Работа защищена «__» _____
С оценкой _____

Задача 10.

Дана разреженная матрица (CCS). Зеркальное отображение относительно диагонали, проходящей с левого нижнего угла к правому верхнему углу.

1) Программный код.

Класс Main.java:

```
import java.io.FileNotFoundException;
import java.io.File;
import java.util.InputMismatchException;
import java.util.Scanner;
public class Main {
    public static void main(String[] args) throws Exception {
        System.out.println("Задача 10.\nДана разреженная матрица (CCS). Зеркальное
отображение относительно\n" +
            "диагонали, проходящей с левого нижнего угла к правому верхнему углу.");
        Scanner scanner1 = new Scanner(new
File("/home/denis/JavaPrograms/src/Laba2/src/matrix.txt"));
        int row = scanner1.nextInt(); //считываем количество строк из файла
        int col = scanner1.nextInt(); //считываем количество столбцов из файла
        int n=0;
        int count = 1; //переменная для подсчета ненул. элементов
        int length = 0; //количество элементов для массивов : A, LI
        int[] LI; //массив номеров столбцов
        int[] LJ; //массив местоположения первого элемента каждого столбца
        int[] A; //массив для ненул. элементов
        int index = 0;
        int[][] matrix = readMatrixInFile("/home/denis/JavaPrograms/src/Laba2/src/matrix.txt");
//считываем матрицу из файла
        for(int i=0;i<row;i++) {
            for(int j=0;j<col;j++) {
                if(matrix[i][j] != 0){
                    length++; //подсчитываем количество ненул. элементов
                }
            }
        }
        A = new int[length]; //создаем динамический массив
        LI = new int[length]; //создаем динамический массив
        for(int j=0;j<col;j++) {
            for(int i=0;i<row;i++) {
                if(matrix[i][j] != 0){
                    A[index] = matrix[i][j]; //запоминаем ненул. элементы
                    LI[index] = i+1; //запоминаем столбцы+1
                    index++;
                }
            }
        }
        int[] LJ_copy = new int[length]; //копия массива LJ, чтобы потом избавиться от нулей
        for(int j=0;j<col;j++) {
            for(int i=0;i<row;i++) {
                if(matrix[i][j] != 0) { //если встретили столбец где есть ненул. элемент.
                    for(int k=0;k<row;k++) { //заново проходим по этому столбцу с 0 индекса
                        if (matrix[k][j] != 0) {
```

```

        count++;          //запоминаем количество ненул. элементов в столбце
    }
}
LJ_copy[n+1] = count;    //запоминаем в ячейку n+1 количество элементов, т.к.
n++ делаем в конце
break;                  //выходим из цикла и переходим к след.столбцу
}
}
if(LJ_copy[n] == 0 && n>0){    //если не зашли в цикл выше
    LJ_copy[n] = LJ_copy[n - 1]; //в пустую ячейк LJ запоминаем n-1 элемент
}
else if(LJ_copy[n] == 0) {    //если LJ элемент = 0
    LJ_copy[n] = n+1;        //записываем в массив n+1, т.к. в первой ячейке LJ всегда
будет значение 1
}
n++;                      //увеличиваем индекс LJ массива
}
int LJ_length = 0;        //переменная длины LJ массива, без нулей
for(int i=0;i<LJ_copy.length;i++) {
    if(LJ_copy[i] != 0) {
        LJ_length++;        //длина LJ массива, без нулей
    }
}
LJ = new int[LJ_length];    //создаем массив LJ без нулей
for(int i=0;i<LJ.length;i++) {
    if(LJ_copy[i] != 0) {
        LJ[i] = LJ_copy[i];    //запоминаем ненул. элементы массива LJ
    }
}
System.out.println("Разреженная матрица:");
printMatrix(matrix,row,col); //вывод начальной матрицы
printInfoSparseMatrix(A,LJ,LJ); //вывод информации о разреженной матрицы
int temp;                    //переменная для запоминания промежуточного элемента
массива
int newMatrix[][] = new int[row][col];    //создаем динамич. массив для записи в него
новой матрицы
for(int i=0;i<row;i++){
    for(int j=0;j<col;j++){
        temp = matrix[i][j];    //запоминаем элемент нач. матрицы
        newMatrix[i][j] = matrix[row - j - 1][col - i - 1]; //идем по столбцам с конца матрицы,
снизу вверх
        newMatrix[row - j - 1][col - i - 1] = temp;    //меняем элемент
    }
}
System.out.println("\nЗеркальное отображение относительно диагонали," + "проходящей
с левого нижнего угла к правому верхнему углу.");
printMatrix(newMatrix,row,col); //вывод новой матрицы
}
static void printElements(int[] array) {
    for(Integer elem: array) {
        System.out.print(elem+" | ");
    }
}

```

```

    }
}
static void printMatrix(int[][] matrix, int row, int col) {
    for(int i=0;i<row;i++){
        for(int j=0;j<col;j++){
            System.out.print(matrix[i][j]+"\\t");
        }
        System.out.println();
    }
}
static void printInfoSparseMatrix(int[] A, int[] LI, int[] LJ) {
    System.out.println("\\nНенулевые элементы разреженной матрицы:");
    printElements(A);
    System.out.println("\\nНомера строк:");
    printElements(LI);
    System.out.println("\\nМестоположение элемента каждого столбца:");
    printElements(LJ);
}
static int[][] readMatrixInFile(String path) throws FileNotFoundException{    //метод
считывания матрицы из файла
    Scanner scanner = new Scanner(new File(path));
    int row=scanner.nextInt();    //считываем количество строк из файла
    int col=scanner.nextInt();    //считываем количество столбцов из файла
    int[][] matrix = new int[row][col]; //создаем двум. динамич. массив
    try {
        for(int i=0;i<row;i++){
            for(int j=0;j<col;j++){
                matrix[i][j]=scanner.nextInt(); //запоминаем элементы по-символьно

            }
        }
    } catch (InputMismatchException e){    //проверка на некорректные значения в файле
        System.out.println("В файле присутствует элемент не соответствующий типу!");
        System.out.println("Данный элемент заменен на 0.");
    }
    return matrix;
}
}

```

2)Описание функций:

1) **void printMatrix(int[][] matrix, int row, int col)** — вывод матрицы, функция получает матрицу, количество строк и столбцов.

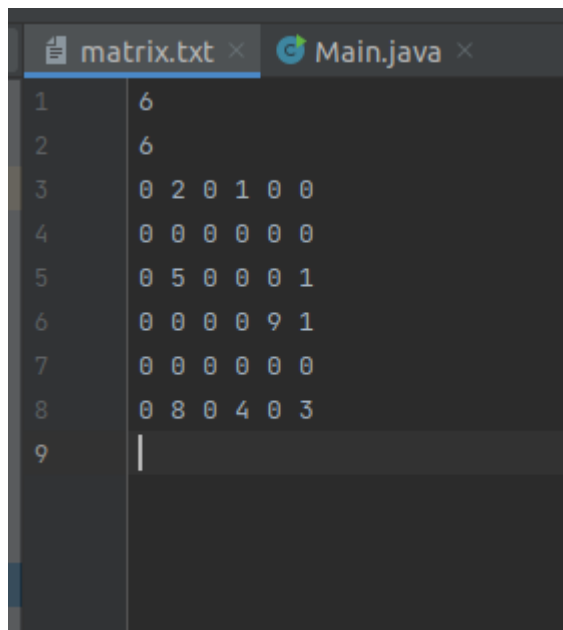
2) **static void printElements(int[] array)** — вывод массива который содержит информацию о разреженной матрице.

3) **static void printInfoSparseMatrix(int[] A, int[] LI, int[] LJ)** — вывод за раз всей информации о разреженной матрице

4) **static int[][] readMatrixInFile(String path)** — метод считывания матрицы из файла.

3) Результаты работы программы.

Файл для считывания с разреженной матрицей:



The image shows a code editor with two tabs: 'matrix.txt' and 'Main.java'. The 'matrix.txt' tab is active and displays a sparse matrix with 9 rows and 7 columns. The matrix is as follows:

1	6					
2	6					
3	0	2	0	1	0	0
4	0	0	0	0	0	0
5	0	5	0	0	0	1
6	0	0	0	0	9	1
7	0	0	0	0	0	0
8	0	8	0	4	0	3
9						

Вывод в консоль:

Задача 10.

Дана разреженная матрица (CCS). Зеркальное отображение относительно диагонали, проходящей с левого нижнего угла к правому верхнему углу.

Разреженная матрица:

0	2	0	1	0	0
0	0	0	0	0	0
0	5	0	0	0	1
0	0	0	0	9	1
0	0	0	0	0	0
0	8	0	4	0	3

Ненулевые элементы разреженной матрицы:

2 | 5 | 8 | 1 | 4 | 9 | 1 | 1 | 3 |

Номера строк:

1 | 3 | 6 | 1 | 6 | 4 | 3 | 4 | 6 |

Местоположение элемента каждого столбца:

1 | 1 | 4 | 4 | 6 | 7 | 10 |

Зеркальное отображение относительно диагонали, проходящей с левого нижнего угла к правому верхнему углу.:

3	0	1	1	0	0
0	0	9	0	0	0
4	0	0	0	0	1
0	0	0	0	0	0
8	0	0	5	0	2
0	0	0	0	0	0

Process finished with exit code 0

|