**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

**Курсовая РАБОТА**

**по дисциплине «Программирование»**

**Тема: Работа с квадратными матрицами в текстовых файлах**

|  |  |  |
| --- | --- | --- |
| Студентка гр. 6304 |  | Иванкова В.М. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2017

**ЗАДАНИЕ на курсовую работу**

|  |  |  |
| --- | --- | --- |
| Студентка Иванкова В.М. | | |
| Группа 6304 | | |
| Тема работы: Работа с квадратными матрицами в текстовых файлах | | |
| Содержание пояснительной записки:   * Содержание * Введение * Описание функций * Примеры работы программы * Заключение * Список использованных источников * Приложение: Исходный код программы | | |
| Предполагаемый объем пояснительной записки:  15 страниц. | | |
| Дата выдачи задания: \_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
| Дата сдачи реферата: \_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
| Дата защиты реферата: \_\_\_\_\_\_\_\_\_\_\_\_\_ | | |
| Студентка |  | Иванкова В.М. |
| Преподаватель |  | Берленко Т.А. |

# 

## **Аннотация**

В данной работе был создан проект на языке программирования C, который позволяет отсортировывать строки заданной квадратной матрицы по возрастанию первого элемента и сохранять результат в новом файле.

Были созданы и описаны необходимые функции, позволяющие: считывать параметры из входного потока; проверять, что матрица является квадратной. Приведено полное описание исходного кода и примеры работы программы.

**Оглавление**

[**Аннотация** 3](#_Toc484133417)

[**Введение** 5](#_Toc484133418)

[**Цель работы** 5](#_Toc484133419)

[**Формулировка задачи** 5](#_Toc484133420)

[**Ход работы** 6](#_Toc484133421)

[**Примеры использования программы** 8](#_Toc484133422)

[**Заключение** 9](#_Toc484133423)

[**Список используемых источников** 10](#_Toc484133424)

[**Приложение** 11](#_Toc484133425)

## **Введение**

### **Цель работы**

Создание проекта, позволяющего отсортировывать строки квадратной матрицы по возрастанию первого элемента и сохранение результата в новом файле. Реализация функций для: считывания параметров из входного потока; проверки матрицы на то, является ли она квадратной или нет.

### **Формулировка задачи**

Требуется написать программу, на вход которой подается квадратная матрица. Программа должна отсортировать строки матрицы по возрастанию первого элемента (если элементы равны, по второму и т.д.) и сохранять результат в новом файле.

Программа получает параметры из входного потока (проверять на корректность не требуется). Параметр:

* input\_file  - имя текстового файла

Программа должна проверять, что матрица является квадратной. В случае, если это не так:

* не создается выходного в файла
* выводится сообщение об ошибке “Fail”.

## **Ход работы**

#### **Функция для считывания матрицы из файла**

void readMatrixFromFile(char\* file\_name, int row, int column, int\*\* array)

{

FILE\* input\_file = fopen(file\_name, "r");

for (int i = 0; i < row; i++)

{

for (int j = 0; j < column; j++)

{

// Считывание элемента из файла с матрицей

fscanf(input\_file, "%d", &array[i][j]);

}

}

}

#### **Функция вывода матрицы на консоль**

void printMatrix(int\*\* array, int row, int column)

{

printf("\nArray:\n");

for (int i = 0; i < row; i++)

{

for (int j = 0; j < column; j++)

{

printf("%d ", array[i][j]);

}

printf("\n");

}

}

#### **Функция сортировки матрицы**

void sortMatrix(int\*\* array, int row, int column)

{

for (int i=0;i<column;i++)

{

for (int j=0;j<row;j++)

{

if (array[i][0]<array[j][0])

{

int temp=array[j][0];

array[j][0]=array[i][0];

array[i][0]=temp;

for (int k=1;k<column;k++)

{

temp=array[j][k];

array[j][k]=array[i][k];

array[i][k]=temp;

}

}

}

}

}

#### **Функция сохранения отсортированной матрицы в файл**

void saveToFileMatrix(int\*\* array, int row, int column)

{

FILE\* output\_file = fopen("./output.txt", "w");

for (int i = 0; i < row; i++)

{

for (int j = 0; j < column; j++)

{

fprintf(output\_file, "%d ", array[i][j]);

}

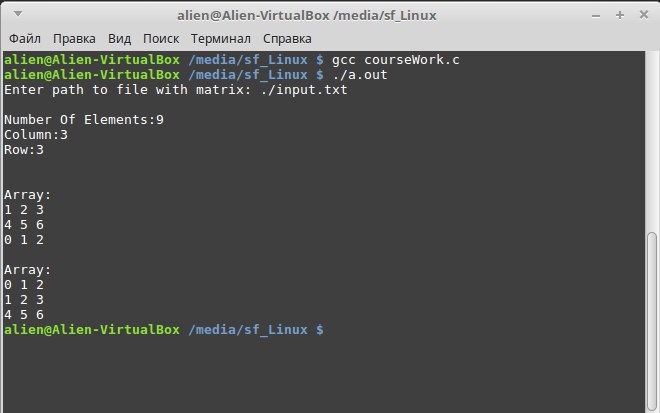
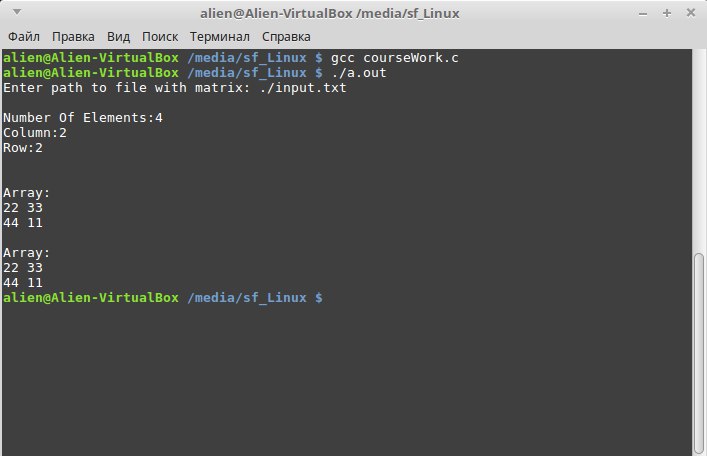
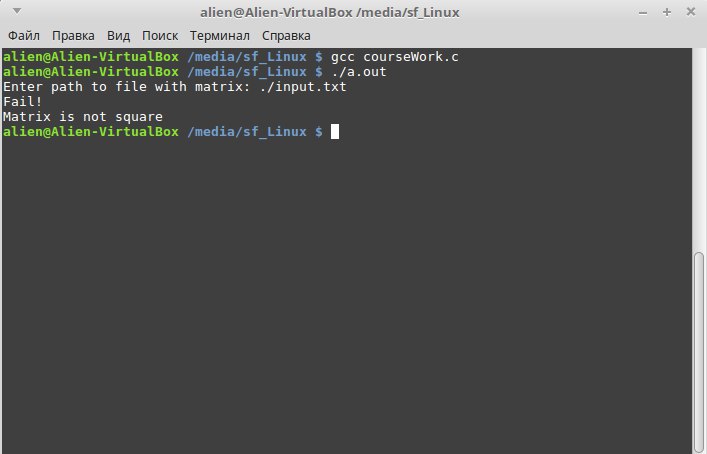
fprintf(output\_file, "\n");

}

fclose(output\_file);

}

## **Примеры использования программы**



## **Заключение**

В ходе выполнения данной курсовой работы, были закреплены на практике знания о работе с матрицами и файлах, посредством создания функций для работы с ними на примере программы, обрабатывающей квадратную матрицу, сортирующую ее и сохраняющую результат в новый файл.

Были реализованы функции, такие как: считывание входного потока; проверка матрицы на то, является ли она квадратной или нет; вывод матрицы на консоль; сортировка матрицы и сохранение отсортированной матрицы в файл.

## **Список используемых источников**

* 1. Язык программирования СИ / Керниган Б., Ритчи Д. СПб: Издательство «Невский Диалект», 2001. 352 с.
  2. UNIX. Программное окружение / Керниган Б., Пайк Р. СПб: Символ Плюс, 2003. 416с.

## **Приложение**

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

// Считывание матрицы из файла

void readMatrixFromFile(char\* file\_name, int row, int column, int\*\* array)

{

FILE\* input\_file = fopen(file\_name, "r");

for (int i = 0; i < row; i++)

{

for (int j = 0; j < column; j++)

{

// Считывание элемента из файла с матрицей

fscanf(input\_file, "%d", &array[i][j]);

}

}

}

// Вывод матрицы на консоль

void printMatrix(int\*\* array, int row, int column)

{

printf("\nArray:\n");

for (int i = 0; i < row; i++)

{

for (int j = 0; j < column; j++)

{

printf("%d ", array[i][j]);

}

printf("\n");

}

}

// Сортировка матрицы

void sortMatrix(int\*\* array, int row, int column)

{

for (int i=0;i<column;i++)

{

for (int j=0;j<row;j++)

{

if (array[i][0]<array[j][0])

{

int temp=array[j][0];

array[j][0]=array[i][0];

array[i][0]=temp;

for (int k=1;k<column;k++)

{

temp=array[j][k];

array[j][k]=array[i][k];

array[i][k]=temp;

}

}

}

}

}

// Сохранение отсортированной матрицы в файл

void saveToFileMatrix(int\*\* array, int row, int column)

{

FILE\* output\_file = fopen("./output.txt", "w");

for (int i = 0; i < row; i++)

{

for (int j = 0; j < column; j++)

{

fprintf(output\_file, "%d ", array[i][j]);

}

fprintf(output\_file, "\n");

}

fclose(output\_file);

}

int main()

{

printf("Enter path to file with matrix: ");

char file\_name[100];

fgets(file\_name, 100, stdin);

(\*strstr(file\_name, "\n")) = 0;

// Открываем файл с матрицей

FILE\* input\_file = fopen(file\_name, "r");

// Выводим сообщение, если файл не существует

if (!input\_file)

{

printf("Fail!\n");

printf("Error with file: %s\n", file\_name);

return 0;

}

int NumberOfElements = 0; // Количество элементов в матрице

int row = 0; // Количество строк в матрице

int column = 0; // Количество столбцов в матрице

// Считываем все символы из файла

while (!feof(input\_file))

{

char currentSymbol;

fscanf(input\_file, "%c", &currentSymbol);

// Если считанный символ - \n - увеличиваем количество строк

if (currentSymbol == '\n')

column++;

/\*

Если считанный символ - пробел - увеличиваем количество

элементов матрицы, так как после каждого числа идет

пробел, кроме последних чисел в строке - там абзац

\*/

if (currentSymbol == ' ')

NumberOfElements++;

}

/\*

Увеличиваем количество стоблов на 1, так

как после последней строки не стоит знак абзаца

\*/

column++;

/\*

Добавляем к NumberOfElements количество последних числел в строке

после которых идет абазц, они не были посчтина в цикле

\*/

NumberOfElements += column;

// Высчитываем количество строк в матрице

row = NumberOfElements / column;

// Проверка на квадратность матрицы

if (row != column)

{

printf("Fail!\n");

printf("Matrix is not square\n");

return 0;

}

printf("\nNumber Of Elements:%d\nColumn:%d\nRow:%d\n\n", NumberOfElements, column, row);

fclose(input\_file);

int\*\* array = (int\*\*)malloc(sizeof(int\*)\*row);

for (int i = 0; i < row; i++)

{

array[i] = (int\*)malloc(sizeof(int)\*column);

}

readMatrixFromFile(file\_name, row, column, array);

printMatrix(array, row, column);

sortMatrix(array, row, column);

printMatrix(array, row, column);

saveToFileMatrix(array, row, column);

return 0;

}