**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

**Курсовая РАБОТА**

**по дисциплине «Программирование»**

**Тема: Умножение матриц**

|  |  |  |
| --- | --- | --- |
| Студент гр. 6303 |  | Филиппенко Д.Р. |
| Преподаватель |  | Берленко Т.А. |

Санкт-Петербург

2017

**ЗАДАНИЕ**

**на курсовую работу**

|  |  |  |
| --- | --- | --- |
| Студент Филиппенко Д.Р. | | |
| Группа 6303 | | |
| Тема работы: Умножение матриц | | |
|  | | |
| Содержание пояснительной записки:   * Содержание * Введение * Описание функции перемножения матриц * Примеры работы программы * Заключение * Список использованных источников * Приложение А. Исходный код программы | | |
| Предполагаемый объем пояснительной записки:  Не менее 10 страниц. | | |
| Дата выдачи задания: 28.04.2017 | | |
| Дата сдачи курсовой работы: 02.06.2017 | | |
| Дата защиты курсовой работы: 02.06.2017 | | |
| Студент |  | Филиппенко Д.Р. |
| Преподаватель |  | Берленко Т.А. |

**Аннотация**

В данной работе была разработана программа на языке программирования C, на вход которой подается файл, содержащий 2 матрицы. Реализована функция перемножения матриц. Результат вычисления записывается в новый файл. В ходе работы программы выполняются проверки на корректность размеров матриц, возможность их умножения. В случае невыполнения одного из условий, на консоль выводится сообщение об ошибке. Приведено полное описание исходного кода.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc484133443)

[1. ФУНКЦИЯ ПЕРЕМНОЖЕНИЯ МАТРИЦ 5](#_Toc484133444)

[1.1. ОБЪЯВЛЕНИЕ ФУНКЦИИ 5](#_Toc484133445)

[1.2. ОПИСАНИЕ ФУНКЦИИ 5](#_Toc484133446)

[1.2.1. ОТКРЫТИЕ ФАЙЛА НА ЗАПИСЬ 5](#_Toc484133447)

[1.2.2. АЛГОРИТМ УМНОЖЕНИЯ МАТРИЦ 5](#_Toc484133448)

[2. ФУНКЦИЯ MAIN 6](#_Toc484133449)

[2.1. ОТКРЫТИЕ ИСХОДНОГО ФАЙЛА НА ЧТЕНИЕ 6](#_Toc484133450)

[2.2. ВЫДЕЛЕНИЕ ПАМЯТИ ПОД МАТРИЦЫ 6](#_Toc484133451)

[2.3. СЧИТЫВАНИЕ СТРОК 6](#_Toc484133452)

[2.4. ВЫЗОВ ФУНКЦИИ MULTIPLICATION 7](#_Toc484133453)

[ПРИМЕРЫ РАБОТЫ ПРОГРАММЫ 8](#_Toc484133454)

[ЗАКЛЮЧЕНИЕ 9](#_Toc484133455)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 10](#_Toc484133456)

[ПРИЛОЖЕНИЕ А. 11](#_Toc484133457)

[ИСХОДНЫЙ КОД ПРОГРАММЫ 11](#_Toc484133458)

# ВВЕДЕНИЕ

Требуется написать программу, на вход которой подаются две матрицы. Программа должна перемножить матрицы и сохранить результат в новый файл.

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

* input\_file  - имя текстового файла

Программа должна проверять корректность размера матриц. В случае, если это не так:

* не создается выходного в файла
* выводится сообщение об ошибке “Fail”.
* выводится сообщение об ошибке “Fail with <номер матрицы>”.
* сообщение об ошибке выводится для каждой матрицы отдельное.

# ФУНКЦИЯ ПЕРЕМНОЖЕНИЯ МАТРИЦ

# ОБЪЯВЛЕНИЕ ФУНКЦИИ

void multiplication(First , Second );

В функцию multiplication передаем две структуры, в которых хранятся матрицы и их размеры

typedef struct First

{

    int \*\*array;

    int m;

    int n;

}First;

typedef struct Second

{

    int \*\*array;

    int m;

    int n;

}Second;

# ОПИСАНИЕ ФУНКЦИИ

# ОТКРЫТИЕ ФАЙЛА НА ЗАПИСЬ

FILE \*result = fopen("result.txt", "w");

Создаем файл, в который будет записываться результат перемножения матриц

# АЛГОРИТМ УМНОЖЕНИЯ МАТРИЦ

for (i=0; i < m; i++)

    {

        for (j=0; j < n; j++)

        {

            int sum=0;

            for (k=0; k < k\_max; k++)

                sum += (FirstMatrix.array)[i][k]\*(SecondMatrix.array)[k][j];

            fprintf(result, "%d ", sum);

        }

        fprintf(result,"\n");

    }

# ФУНКЦИЯ MAIN

# ОТКРЫТИЕ ИСХОДНОГО ФАЙЛА НА ЧТЕНИЕ

FILE \*f = fopen("1.txt", "r");

# ВЫДЕЛЕНИЕ ПАМЯТИ ПОД МАТРИЦЫ

FirstMatrix.array=(int \*\*)malloc(20\*sizeof(int \*));

    for (i = 0; i < 20; ++i)

    {

        (FirstMatrix.array)[i] = (int \*)malloc(20\*sizeof(int));

    }

    SecondMatrix.array=(int \*\*)malloc(20\*sizeof(int \*));

    for (i = 0; i < 20; ++i)

    {

        (SecondMatrix.array)[i] = (int \*)malloc(20\*sizeof(int));

    }

# СЧИТЫВАНИЕ СТРОК

fgets(string, 50, f);

    while((strstr(string, "\n")-string)>=1)

    {

        element = strtok(string," \n");

        while(element)

        {

            (FirstMatrix.array)[n1][m1] = atoi(element);

            m1++;

            element = strtok(NULL, " \n");

        }

        if (n1 == 0)

            temp\_m = m1;

        if ( (temp\_m - m1) != 0)

            log1++;

        m1=0;

  n1++;

        fgets(string, 50, f);

    }

При помощи fgets считываем первую строку матрицы.   
Если строка не пустая, то функцией strtok разбиваем строку на отдельные элементы и записываем их в массив. m – число столбцов. Увеличиваем это значение пока не дойдем до конца строки.

Переменная log1 отвечает за корректность первой матрицы. Если log1 оказался больше нуля, значит, какая-то строка не равна первой, следовательно, матрица не корректна.

В этом случае выводим сообщение об ошибке.

if (log1)

        printf("Fail with first matrix\n");

    m1=temp\_m;

В условии сказано, что матрицы разделены тремя символами перевода строки.

Первый символ уже считан. Считываем два оставшихся и аналогичным способом считываем вторую матрицу и проверяем ее корректность.

# ВЫЗОВ ФУНКЦИИ MULTIPLICATION

    if ( (m1 == n2) && log1 == 0 && log2 == 0 )

        multiplication(FirstMatrix, SecondMatrix);

    else

        printf("Fail\n");

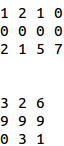
Выполняем проверку соответствия размеров матриц. Если число столбцов первой матрицы соответствует числу строк второй, то выполняем умножение. Иначе выводим сообщение об ошибке.

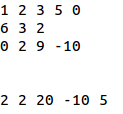
# ПРИМЕРЫ РАБОТЫ ПРОГРАММЫ

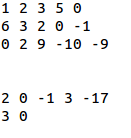
ВВОД











ВЫВОД











# ЗАКЛЮЧЕНИЕ

Была создана программа, считывающая матрицы из файла и умножающая их.

Также были реализованы условия, проверяющие корректность введенных матриц и соответствие их размеров для умножения.

# СПИСОК ИСПОЛЬЗОВАННЫХ 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. 416 с.

# ПРИЛОЖЕНИЕ А.

# ИСХОДНЫЙ КОД ПРОГРАММЫ

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

typedef struct First

{

    int \*\*array;

    int m;

    int n;

}First;

typedef struct Second

{

    int \*\*array;

    int m;

    int n;

}Second;

void multiplication(First , Second );

int main()

{

    FILE \*f = fopen("1.txt", "r"); //открываем файл на чтение

    char \*string = (char \*)malloc(51);

    char \*element = (char \*)malloc(5);

    First FirstMatrix;

    Second SecondMatrix;

    int temp\_m = 0; //m - столбцы, n - строки

    int n1 = 0, n2 = 0, m1 = 0, m2 = 0, log1 = 0, log2 = 0;

    int i;

    FirstMatrix.array=(int \*\*)malloc(20\*sizeof(int \*));

    for (i = 0; i < 20; ++i)

    {

        (FirstMatrix.array)[i] = (int \*)malloc(20\*sizeof(int));

    }

    SecondMatrix.array=(int \*\*)malloc(20\*sizeof(int \*));

    for (i = 0; i < 20; ++i)

    {

        (SecondMatrix.array)[i] = (int \*)malloc(20\*sizeof(int));

    }

    fgets(string, 50, f);

    while((strstr(string, "\n")-string)>=1)//проверка на то, пустая(только \n) ли строка

    {

        element = strtok(string," \n"); //разбиваем на строки по пробелам

        while(element)

        {

            (FirstMatrix.array)[n1][m1] = atoi(element); //переводим в число и записываем в массив

            m1++;     //увеличиваем количество столбцов

            element = strtok(NULL, " \n");

        }

        if (n1 == 0)            //первая строка

            temp\_m = m1;       //запись размера первой строки

        if ( (temp\_m - m1) != 0 )   //проверка на размер

            log1++;                                 //если размер этой строки не равен размеру первой увеличиваем итератор

        m1=0;                                       //обнуляем количество элементов в строке

        n1++;                                       //увеличиваем количество строк

        fgets(string, 50, f);                       //считываем следующую строку

    }

    if (log1)

        printf("Fail with first matrix\n"); //если итератор не равен нулю, значит где-то строка <> первой => ошибка

    m1=temp\_m;

    fgets(string, 50, f); //считываем две пустые строки

    fgets(string, 50, f);

    while((strstr(string, "\n")-string)>=1)     //все аналогично первой

    {

        element = strtok(string," \n");

        while(element)

        {

            (SecondMatrix.array)[n2][m2] = atoi(element);

            m2++;

            element = strtok(NULL, " \n");

        }

        if (n2 == 0)

            temp\_m = m2;

        if ( (temp\_m - m2) != 0 )

            log2++;

        m2=0;

        n2++;

        fgets(string, 50, f);

    }

    if (log2)

        printf("Fail with second matrix\n");

    m2=temp\_m;

    fclose(f); //закрываем файл

    //записываем размеры матриц в структуры

    FirstMatrix.n=m1;

    FirstMatrix.m=n1;

    SecondMatrix.n=m2;

    SecondMatrix.m=n2;

    if ( (m1 == n2) && log1 == 0 && log2 == 0 )

        multiplication(FirstMatrix, SecondMatrix);

    else

        printf("Multiplication impossible\n");

    for (i = 0; i < 20; ++i)

    {

        free(SecondMatrix.array)[i]); //освобождаем память

    }

    free(SecondMatrix.array);

    for (i = 0; i < 20; ++i)

    {

        free(FirstMatrix.array)[i]));

    }

    free(FirstMatrix.array);

    return 0;

}

void multiplication(First FirstMatrix, Second SecondMatrix)

{

    FILE \*result = fopen("result.txt", "w"); //открываем файл на запись

    int m = FirstMatrix.m;  //присвоение размеров

    int n = SecondMatrix.n;

    int k\_max=FirstMatrix.n;

    int i,j,k;

    int \*\*n\_array = (int \*\*)malloc(m\*sizeof(int \*)); //выделение памяти

    for(i=0; i < m; i++)

    {

        n\_array[i] = (int \*)malloc(n\*sizeof(int));

    }

    for (i=0; i < m; i++)

    {

        for (j=0; j < n; j++)

        {

            int sum=0;

            for (k=0; k < k\_max; k++)

                sum += (FirstMatrix.array)[i][k]\*(SecondMatrix.array)[k][j]; //запись в новый массив(сразу в файл)

            fprintf(result, "%d ", sum);

        }

        fprintf(result,"\n");

    }

    fclose(result); //закрываем файл

}