«Отчет по ПЗ 1.2. Динамические массивы (матрицы)» Фаустова Анастасия \*\*\*. Вариант № 20.

1. Задание: Определить число столбцов матрицы, сумма элементов которых положительна.

1. Постановка задачи.

Входные данные + ограничения:

А) Переменная n. Тип int. Размерность матрицы (количество строк).

Исключения: буква, буква с цифрой, строка, вещественное, меньше или равно 0, большие числа (зависит от наличия свободной оперативной памяти компьютера).

Б) Переменная m. Тип int. Размерность матрицы (количество столбцов).

Исключения: буква, буква с цифрой, строка, вещественное, меньше или равно 0, большие числа (зависит от наличия свободной оперативной памяти компьютера).

В) Переменная matrix[][]. Тип \*\*float. Элементы матрицы.

Исключения: буква, буква с цифрой, строка.

Выходные данные + ограничения:

А) Переменная matrix[][]. Тип \*\*float. Элементы матрицы.

Исключения: буква, буква с цифрой, строка.

Б) Перемення count. Тип int. Количество столбцов с положительной суммой элементов.

буква, буква с цифрой, строка, вещественное, меньше или равно 0, большие числа (зависит от наличия свободной оперативной памяти компьютера).

Промежуточные данные + ограничения:

А) Переменная key. Тип int. Флаг переменная для сигнализирования наличия ошибок.

Исключения: буква, буква с цифрой, строка, вещественное число, числа не равные 0 или 1.

Б) Переменная n. Тип float. Промежуточная переменная для проверки на буквенное значение размера массива.

Исключения: буква, буква с цифрой, строка, значение меньше 1.

В) Переменная N. Тип float. Промежуточная переменная для проверки на целостность размеров массивов.

Исключения: буква, буква с цифрой, строка.

Г) Переменная a. Тип char. Промежуточная переменная для проверки на отсутствие буквенных значений в размерах массивов

Исключения: нет.

Д) Переменная m. Тип float. Промежуточная переменная для проверки на буквенное значение размера массива.

Исключения: буква, буква с цифрой, строка, значение меньше 1.

Е) Переменная M. Тип float. Промежуточная переменная для проверки на целостность размеров массивов.

Исключения: буква, буква с цифрой, строка.

Ё) Переменная A. Тип char. Промежуточная переменная для проверки на отсутствие буквенных значений в размерах массивов

Исключения: нет.

Ж) Переменная Z. Тип char. Промежуточная переменная для проверки на отсутствие буквенных значений в размерах массивов

Исключения: нет.

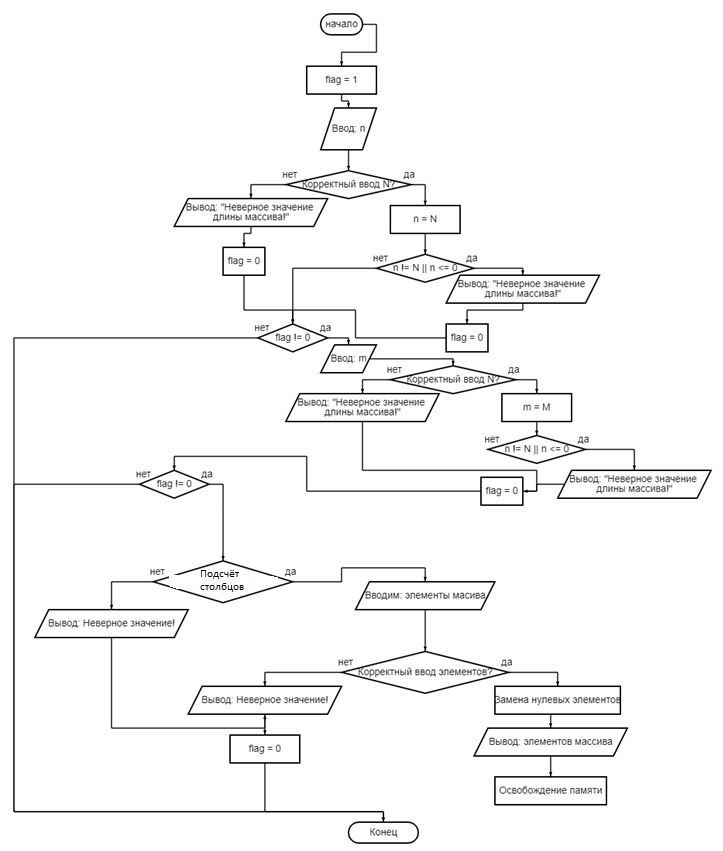
З) Переменная s. Тип float. Промежуточная переменная суммы элементов столбца.

Исключения: буква, буква с цифрой, строка.

Среда разработки: Visual Studio 2022.

2. Структура программы.

2.1 Блок-схема алгоритма



3. Тесты.

|  |  |  |
| --- | --- | --- |
| Входные данные | Результат программы | Смысл |
| n = a | Неверное значение длины матрицы! | Проверка на букву |
| n = 3a | Неверное значение длины матрицы! | Проверка на число c буквой |
| n = 3,2 | Неверное значение длины матрицы!! | Проверка на целое число |
| n = 10000000000000000 | Неверное значение длины матрицы! | Выделение памяти массивам |
| n = 0 | Неверное значение длины матрицы! | Валидность размера массива |
| n = zxcgul | Неверное значение длины матрицы! | Проверка на строку |
| m = b | Неверное значение длины матрицы! | Проверка на букву |
| m = 4a | Неверное значение длины матрицы! | Проверка на число c буквой |
| m = 3,2 | Неверное значение длины матрицы! | Проверка на целое число |
| m = 10000000000000000 | Неверное значение длины матрицы! | Выделение памяти массивам |
| m = 0 | Неверное значение длины матрицы! | Валидность размера массива |
| m = zxcgul | Неверное значение длины матрицы! | Проверка на строку |
| matrix[i][j] = a | Неверное значение элемента матрицы! | Проверка на букву |
| matrix[i][j] = 666rain | Неверное значение элемента матрицы! | Проверка на число c буквой |
| matrix[i][j] = asadsa | Неверное значение элемента матрицы! | Проверка на строку |
| n = 2  m = 2  matrix[0][0] = 0  matrix[0][1] = 0  matrix[1][0] = 0  matrix[1][1] = 0 | Элементы вашей матрицы:  Элемент матрицы [0][0] = 0 Элемент матрицы [0][1] = 0  Элемент матрицы [1][0] = 0 Элемент матрицы [1][1] = 0  Количество подходящих столбцов: 0 | Проверка на корректность работы программы при отсутсвтии положительных и отрицательных чисел |
| n = 2  m = 2  matrix[0][0] = 1  matrix[0][1] = 1  matrix[1][0] = 1  matrix[1][1] = 1 | Элементы вашей матрицы:  Элемент матрицы [0][0] = 1 Элемент матрицы [0][1] = 1  Элемент матрицы [1][0] = 1 Элемент матрицы [1][1] = 1  Количество подходящих столбцов: 2 | Проверка на корректность работы программы cо всеми положительными столбцами |
| n = 2  m = 2  matrix[0][0] = -10  matrix[0][1] = 1  matrix[1][0] = 1  matrix[1][1] = 1 | Элементы вашей матрицы:  Элемент матрицы [0][0] = -10 Элемент матрицы [0][1] = 1  Элемент матрицы [1][0] = 1 Элемент матрицы [1][1] = 1  Количество подходящих столбцов: 1 | Проверка на корректность работы программы c положительными столбцами и с отрицательными |

4. Код:

// ConsoleApplication20.cpp : Этот файл содержит функцию "main". Здесь начинается и заканчивается выполнение программы.

#define \_CRT\_SECURE\_NO\_WARNINGS // Директива препроцесса для устранения предупреждений

#include <iostream>

#include <stdio.h>

using namespace std; // Использование стандартного пространства имён

int main()

{

setlocale(LC\_ALL, "RUS"); // Установка для консоли русского языка

cout << "\t\tОпределить число столбцов матрицы, сумма элементов которых положительна.\n\n";

cout << "Введите количество строк матрицы:\n";

int key = 1; // Промежуточная переменная для проверки на валидность элемента

int n; // Размер массивов

float N; // Промежуточная переменная для проверки на целостность размеров массивов

char a; // Промежуточная переменная для проверки на отсутствие буквенных значений в размерах массивов

if (scanf("%f%c", &N, &a) != 2 || a != '\n') // Проверка размеров матрицы на буквенное значение

{

cout << "Неверное значение длины матрицы!";

key = 0; // Переключение флага для избежания дальнейших действий

}

else

{

n = N;

if (n != N || n <= 0) // Проверка на целостность и положительность размера матрицы

{

cout << "Неверное значение длины матрицы!";

key = 0; // Переключение флага для избежания дальнейших действий

}

int m; // Размерность матрицы (количество столбцов)

float M; // Промежуточная переменная для проверки на целостность размеров матрицы

char A; // Промежуточная переменная для проверки на отсутствие буквенных значений в размерах матрицы

if (key != 0) // Проверка флага на наличие ошибок

{

cout << "Введите количество столбцов матрицы:\n";

if (scanf("%f%c", &M, &A) != 2 || A != '\n') // Проверка значения размерности матрицы на буквенное значение

{

cout << "Неверное значение длины матрицы!";

key = 0; // Переключение флага для избежания дальнейших действий

}

else

{

m = M;

if (m != M || m <= 0) // Проверка валидности ввода размерности матрицы

{

cout << "Неверное значение длины матрицы!";

key = 0; // Переключение флага для избежания дальнейших действий

}

}

}

int count = 0; // Обнуляем счётчик подходящих столбцов

float\*\* matrix;

if (key != 0) // Проверка флага на наличие ошибок

{

matrix = new float\* [N]; // Выделяем память под строки матрицы

for (int i = 0; i < N; i++) // Цикл для прогона по массиву по столбцам

{

matrix[i] = new float[M]; // Выделяем память под столбцы матрицы

}

char Z; // Промежуточная переменная для проверки на отсутствие буквенных значений в элементах матрицы

cout << "Введите элементы матрицы:\n";

for (int i = 0; i < N; i++) // Цикл для прогона по матрице по строкам

{

for (int j = 0; j < M; j++) // Цикл для прогона по матрице по столбцам

{

if (key != 0) // Проверка флага на наличие ошибок

{

cout << "Элемент" << " [" << i << "][" << j << "]\n"; // Вывод элемента матрицы

}

if (key != 0) // Проверка флага на наличие ошибок

{

if (scanf("%f%c", &matrix[i][j], &Z) != 2 || Z != '\n') // Проверка значения элемента матрицы на буквенное значение

{

cout << "Неверное значение элемента матрицы!";

key = 0; // Переключение флага для избежания дальнейших действий

break; // Преждевременный выход из цикла из-за некорректности ввода координат элемента матрицы

}

}

}

}

int s = 0; // Сумма элементов столбца

if (key != 0) // Проверка флага на наличие ошибок

{

for (int j = 0; j < M; j++) // Цикл для прогона по матрицы по строкам

{

for (int i = 0; i < N; i++) // Цикл для прогона по матрицы по столбцам

{

s += matrix[i][j]; // Прибавляем элемент столбца к сумме

}

if (s > 0) // Проверяем положительная ли сумма элементов столбца

{

++count; // Индексируем счётчик

}

s = 0; // Обнуляем сумму для столбцов

}

}

if (key != 0) // Проверка флага на наличие ошибок

{

cout << endl;

cout << "Элементы вашей матрицы:" << endl;

for (int i = 0; i < N; i++) // Цикл для прогона по матрице по строкам

{

for (int j = 0; j < M; j++) // Цикл для прогона по матрице по столбцам

{

cout << "Элемент матрицы " << "[" << i << "][" << j << "] = " << matrix[i][j] << " "; // Вывод элемента матрицы

}

cout << endl;

}

cout << "Количество подходящих столбцов: " << count << endl;

}

if (key != 0) // Проверка флага на наличие ошибок

{

for (int i = 0; i < N; i++) // Цикл для прогона по матрице

{

free(matrix[i]); // Очищение памяти из под столбцов матрице

}

free(matrix); // Очищение памяти из под строки матрицы

}

cout << endl;

}

}

return 0; // Сигнализирование процессору о удачном выполнение работы программы

}