Государственное ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ   
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«Донецкий национальный технический университет»

Д09.03.04-ПОИС.20-20/5847.ЛР7

***Кафедра*** искусственного интеллекта  
 и системного анализа

Лабораторная работа №2

по дисциплине "Основы программирования"

на тему: "Работа с массивами"

Проверили:

\_\_\_\_\_\_\_\_\_\_\_\_ асс. А.П. Семёнова

(дата, подпись)

\_\_\_\_\_\_\_\_\_\_\_\_ асс. И.В. Савицкая

(дата, подпись)

Выполнил:

\_\_\_\_\_\_\_\_\_\_\_\_ ст.гр.ПИ-20г М. А. Евсеев

(дата, подпись)

Донецк – 2020

Задание

1) Составить метод решения задачи и алгоритм обработки массива, в соответствии с заданием.

2) Написать программу на языке Си.

Вариант 20

Найти сумму двух разреженных матриц.

Ход работы:

1. Постановка задачи

Постоянные данные:

x = 100, y = 100, n = 100, m = 100 – значения, которые регулируют размер максимально возможной для ввода матрицы.

Входные данные:

q – целочисленный, количество строк первой матрицы;

w – целочисленный, количество столбцов первой матрицы;

q1 – целочисленный, количество строк второй матрицы;

w1 – целочисленный, количество столбцов второй матрицы;

k1 – счётчик ненулевых элементов первой матрицы;

k2 – счётчик ненулевых элементов второй матрицы;

arrxy[q][w] – первая вещественная матрица, также в этой матрице находится конечная матрица;

arrnm[q1][w1] – вторая вещественная матрица;

Промежуточные данные:

*i* – целочисленный, счетчик цикла;

*j* – целочисленный, счетчик цикла;

Выходные данные:

arrxy[q][w] – матрица вещественных чисел, полученная из суммы введённых матриц.

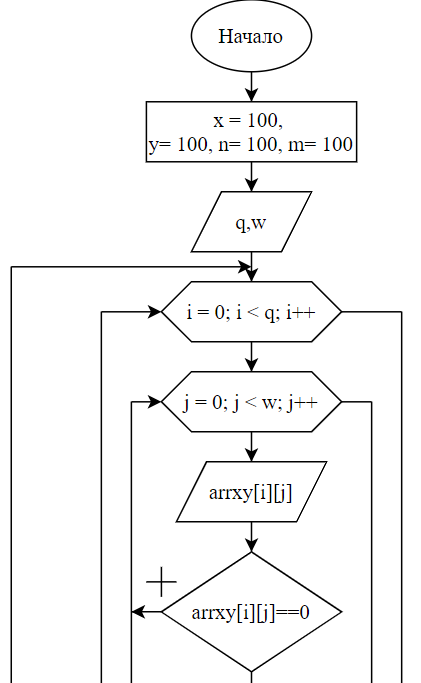
Ограничение на входные данные:

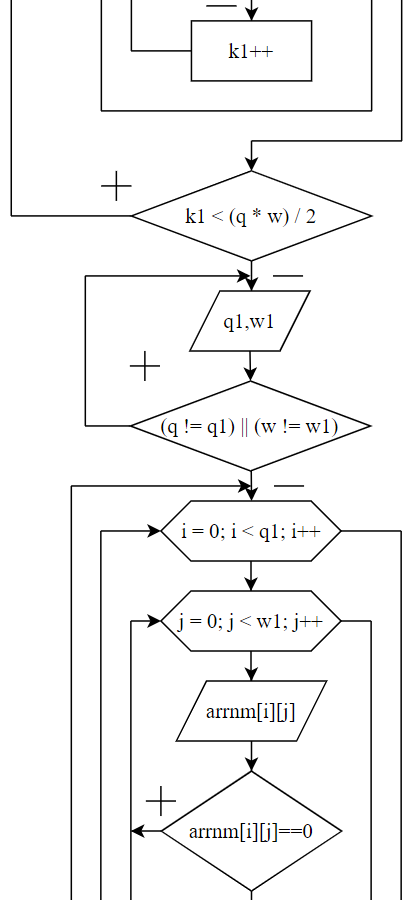
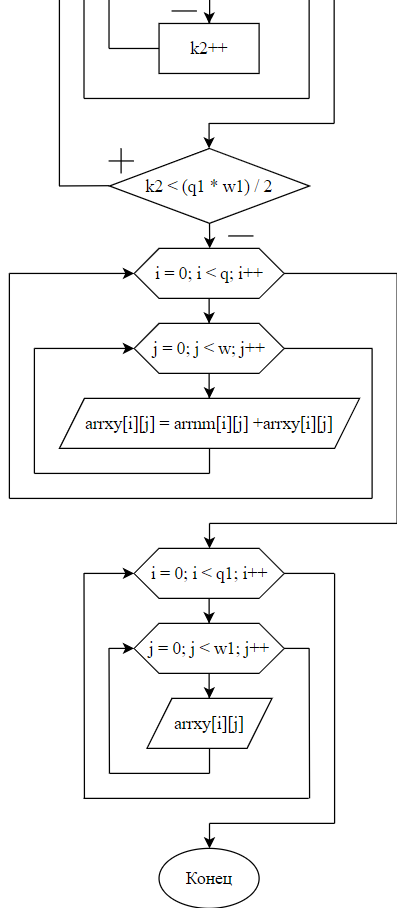
q != q1;

w != w1;

1. Метод решения задачи

Вводим размеры первой матрицы. Считываем саму матрицу, проверяя её на разреженность. Вводим размеры второй матрицы, проверяя на равность размеров с первой матрицей. Считываем матрицу, проверяя её на разреженность. Поэлементно суммируем матрицы и выводим её.

1. Алгоритм решения задачи 

1. Контрольный пример

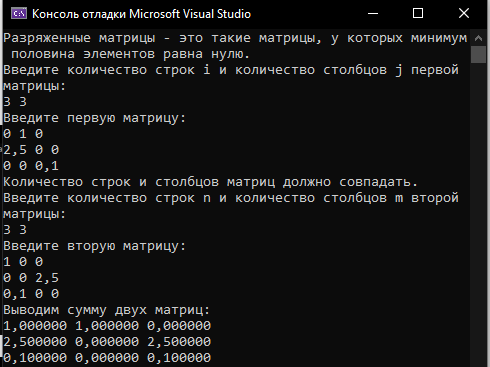


Рисунок 1.1 – Правильный ввод

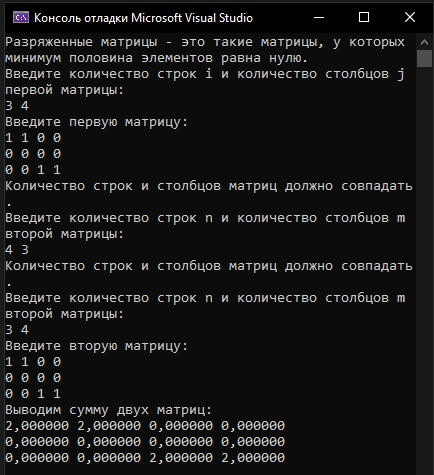


Рисунок 1.2 – Размеры матриц введены неправильно

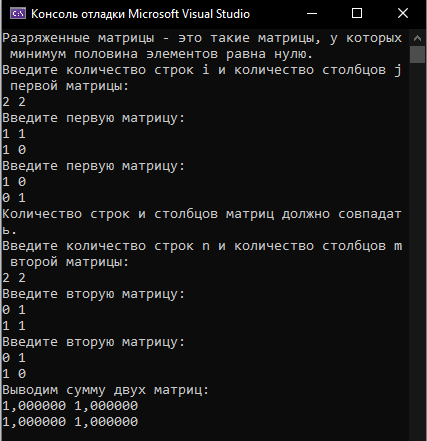


Рисунок 1.3 – Матрицы введены неправильно

1. Листинг программы

#include <stdio.h>

#include <locale.h>

int main()

{

setlocale(LC\_ALL, "Rus");

const int x = 100, y = 100, n = 100, m = 100;

float arrxy[x][y], arrnm[n][m];

int q, w, q1, w1, k1 = 0, k2 = 0;

printf\_s("Разряженные матрицы - это такие матрицы, у которых минимум половина элементов равна нулю.\n");

printf\_s("Введите количество строк i и количество столбцов j первой матрицы:\n");

scanf\_s("%d %d", &q , &w);

do

{

printf\_s("Введите первую матрицу: \n");

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

{

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

{

scanf\_s("%f", &arrxy[i][j]);

if (arrxy[i][j] == 0) k1++;

}

}

} while ((k1 < (q \* w) / 2));

do

{

printf\_s("Количество строк и столбцов матриц должно совпадать.\nВведите количество строк n и количество столбцов m второй матрицы:\n");

scanf\_s("%d %d", &q1, &w1);

} while ((q != q1) || (w != w1));

do

{

printf\_s("Введите вторую матрицу:\n");

for (int i = 0; i < q1; i++)

{

for (int j = 0; j < w1; j++)

{

scanf\_s("%f", &arrnm[i][j]);

if (arrnm[i][j] == 0) k2++;

}

}

} while (k2 < (q1\*w1)/2);

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

{

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

{

arrxy[i][j] = arrxy[i][j] + arrnm[i][j];

}

}

printf\_s("Выводим сумму двух матриц:\n");

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

{

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

{

printf\_s("%f ", arrxy[i][j]);

}

printf\_s("\n");

}

return(0);

}

Вывод: создал программу вычисления суммы разреженных матриц на языке программирования С.

Среднее арифметическое соседей элемента матрицы – сумма элементов, расположенных рядом с данным элементом в строке и столбце, деленная на количество слагаемых суммы.

Чтобы найти среднее арифметическое соседей элемента матрица, мы должны:

1. Убедиться в том, что предполагаемый сосед элемента существует в матрице;
2. Получить сумму существующих соседей;
3. Разделить сумму на количество существующих соседей.

Результат - элемент новой матрицы.

Введем исходную матрицу размерностью arr[N][M], после чего будем следовать методу решения задачи.

Для того, чтобы определить, не является ли элемент матрицы крайним, достаточно знать размерность матрицы и индексы элемента.

Если индекс равен 0, то есть элемент первый в строке или столбце, то соседа с индексом -1 не существует.

Если разность количества строк или столбцов и индекса строк или столбцов равно 1, то соседнего элемента с индексом превышающего количество строк или столбцов не существует.

Поэтапно, проверяя существование каждого соседа элемента матрицы, мы получим их сумму и количество.

В случае, когда мы нашли соседа элемента матрицы, мы складываем соседа с элементом нового массива, который изначально равняется 0 и прибавляем к счетчику соседей единицу k++.

Пройдя 4 проверки на существования элементов и складывая их, мы делим сумму на количество соседей k.

Запишем алгоритм в виде блок-схемы: