Министерство транспорта Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования

«Российский университет транспорта»

(ФГАОУ ВО РУТ(МИИТ), РУТ (МИИТ)

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

Лабораторная работа № 11

по дисциплине: «Программирование и основы алгоритмизации»

на тему: «Методы»

Выполнил: ст. гр. ТУУ-111

Богомолов В.Н.

Вариант №1

23.03.2025

(дата выполнения)

Проверил: к.т.н., доц. Сафронов А.И.

26.03.2025

(дата приёмки)

Москва – 2025 г.

1. **Цель**

Составить программное обеспечение для изменения размерности двумерного массива вещественных чисел на единицу с сохранением содержимого. Метод должен быть нетипизированным, принимать массив по ссылке и логический признак направления изменения.

1. **Формулировка задачи**

Необходимо разработать нетипизированный (пустой) метод \_aResizeArray, предназначенный для изменения размерности исходного двумерного массива вещественных чисел на единицу. Метод должен принимать по ссылке исходный массив и логический параметр, определяющий направление изменения размерности (увеличение или уменьшение). При этом содержимое исходного массива должно сохраняться в новом массиве – элементы копируются в пределах возможных индексов, а ячейки, для которых нет исходных значений, остаются инициализированными значением по умолчанию. Кроме того, программа должна обеспечить корректный контроль ввода исходных данных с использованием метода TryParse, выводя побуждающие сообщения до тех пор, пока пользователь не введёт корректные значения.

1. **Блок-схема алгоритма**

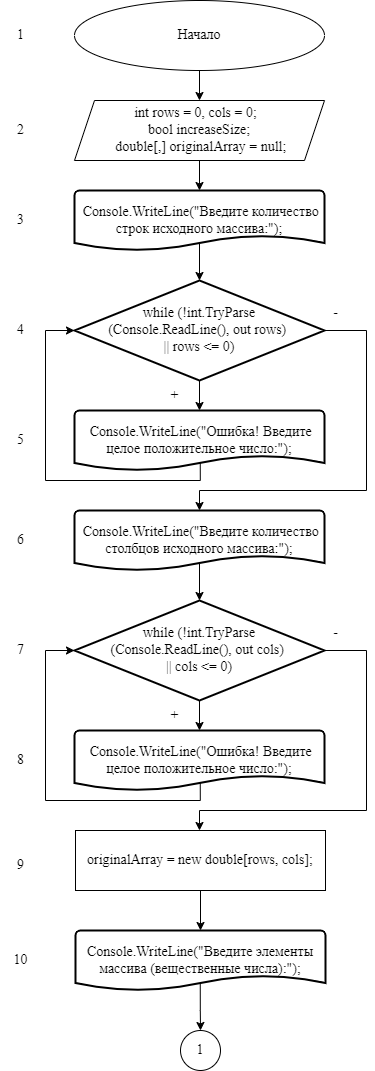
****

Рисунок 3.1.1 – Блок-схема алгоритма

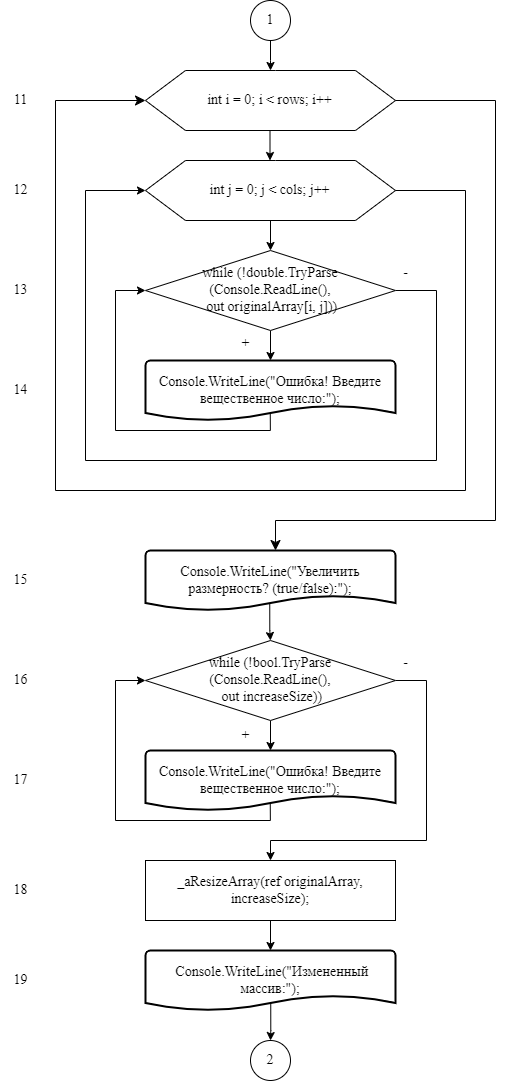
****

Рисунок 3.1.2 – Блок-схема алгоритма

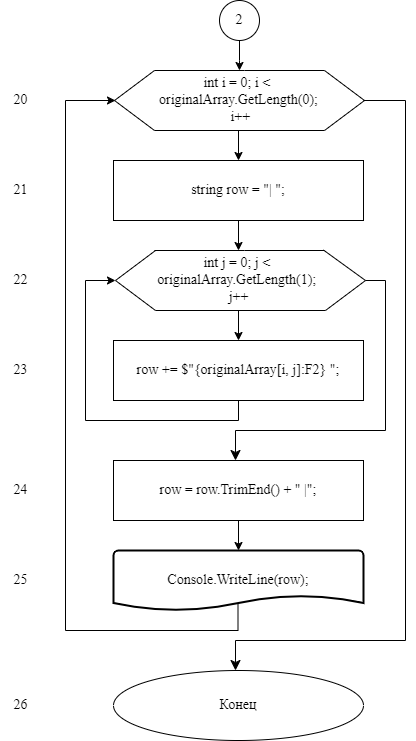
****

Рисунок 3.1.3 – Блок-схема алгоритма

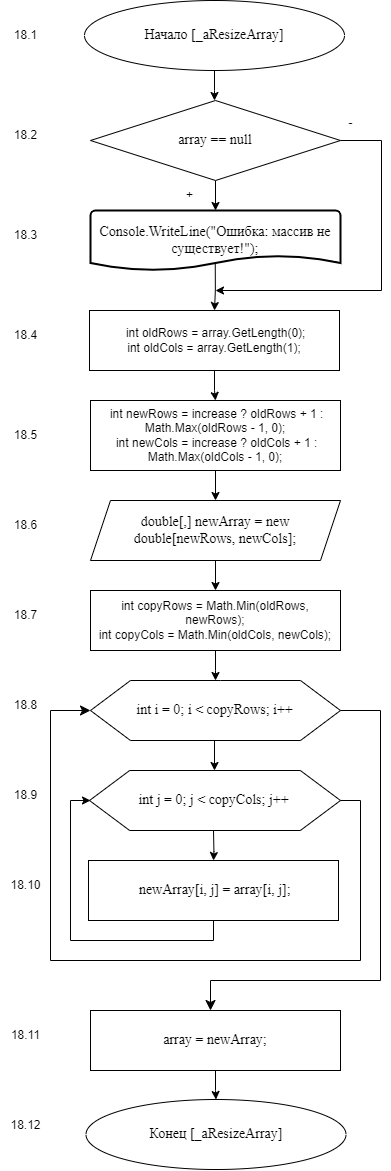
****

Рисунок 3.1.4 – Блок-схема алгоритма

1. **Подбор тестовых примеров**
   1. Тестовый пример №1

Цель: Проверка увеличения размерности.

Ввод:

Строки: 2, Столбцы: 2

1-й элемент массива = 1,1

2-й элемент массива = 2,2

3-й элемент массива = 3,3

4-й элемент массива = 4,4

Увеличение: true

Ожидаемый результат:

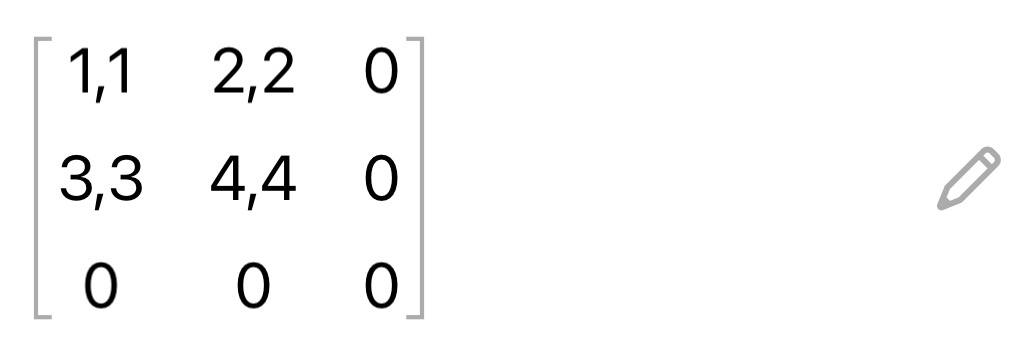


Рисунок 4.1.1 – Тестовый пример №1

* 1. Тестовый пример №2

Цель: Проверка уменьшения до нуля.

Ввод:

Строки: 1, Столбцы: 3

1-й элемент массива = 5,5

2-й элемент массива = 6,6

3-й элемент массива = 7,7

Увеличение: false

Ожидаемый результат: Массив 0x0 (пустой).

* 1. Тестовый пример №3

Цель: Проверка уменьшения размерности.

Ввод:

Строки: 3, Столбцы: 3

1-й элемент массива = 1

2-й элемент массива = 2

3-й элемент массива = 3

4-й элемент массива = 4

5-й элемент массива = 5

6-й элемент массива = 6

7-й элемент массива = 7

8-й элемент массива = 8

9-й элемент массива = 9

Увеличение: false

Ожидаемый результат:



Рисунок 4.3.1 – Тестовый пример №3

* 1. Тестовый пример №4

Цель: Проверка обработки некорректного ввода.

Ввод:

Строки: abc => затем 2 (исправление).

Столбцы: -5 => затем 3 (исправление).

Увеличение: yes => затем true (исправление).

Ожидаемый результат: Программа требует корректные данные до их ввода.

1. **Листинг (код) программы**

using System;

class Program

{

static void Main()

{

// Инициализация переменных для хранения размеров массива

int rows = 0, cols = 0; // Количество строк и столбцов исходного массива

bool increaseSize; // Флаг направления изменения размерности (true - увеличение, false - уменьшение)

double[,] originalArray = null;// Инициализация массива заданного размера (изначально null)

// Ввод количества строк с проверкой корректности

Console.WriteLine("Введите количество строк исходного массива:");

// Цикл продолжается до тех пор, пока пользователь не введет целое положительное число

while (!int.TryParse(Console.ReadLine(), out rows) || rows <= 0)

{

Console.WriteLine("Ошибка! Введите целое положительное число:");

}

// Ввод количества столбцов с аналогичной проверкой

Console.WriteLine("Введите количество столбцов исходного массива:");

while (!int.TryParse(Console.ReadLine(), out cols) || cols <= 0)

{

Console.WriteLine("Ошибка! Введите целое положительное число:");

}

// Создание двумерного массива и его заполнение

originalArray = new double[rows, cols]; // Выделение памяти для массива после получения корректных размеров

Console.WriteLine("Введите элементы массива (вещественные числа):");

// Внешний цикл для перебора строк

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

{

// Внутренний цикл для перебора столбцов

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

{

// Проверка ввода: цикл продолжается, пока пользователь не введет вещественное число

while (!double.TryParse(Console.ReadLine(), out originalArray[i, j]))

{

Console.WriteLine("Ошибка! Введите вещественное число:");

}

}

}

// Ввод направления изменения размерности массива

Console.WriteLine("Увеличить размерность? (true/false):");

// Проверка ввода: только "true" или "false"

while (!bool.TryParse(Console.ReadLine(), out increaseSize))

{

Console.WriteLine("Ошибка! Введите 'true' или 'false':");

}

// Вызов метода изменения размерности массива

// Передача массива по ссылке (ref) позволяет изменить его внутри метода

\_aResizeArray(ref originalArray, increaseSize);

// Вывод измененного массива с форматированием

Console.WriteLine("Измененный массив:");

// Цикл для перебора строк измененного массива

for (int i = 0; i < originalArray.GetLength(0); i++)

{

string row = "| "; // Начало строки с левым разделителем

// Цикл для перебора столбцов

for (int j = 0; j < originalArray.GetLength(1); j++)

{

// Форматирование элемента: два знака после запятой (например, 5 → 5.00)

row += $"{originalArray[i, j]:F2} ";

}

row = row.TrimEnd() + " |"; // Удаление последнего пробела и добавление правого разделителя

Console.WriteLine(row); // Вывод строки

}

Console.ReadKey(); // Ожидание нажатия клавиши для завершения

}

/// <summary>

/// Изменяет размерность двумерного массива на +1 или -1 по каждому измерению.

/// </summary>

/// <param name="array">Исходный массив (передается по ссылке)</param>

/// <param name="increase">true — увеличение размерности, false — уменьшение</param>

static void \_aResizeArray(ref double[,] array, bool increase)

{

// Проверка на null: если массив не существует, выводим ошибку и завершаем метод

if (array == null)

{

Console.WriteLine("Ошибка: массив не существует!");

return;

}

// Получение текущих размеров массива

int oldRows = array.GetLength(0); // Количество строк

int oldCols = array.GetLength(1); // Количество столбцов

// Расчет новых размеров:

// - При увеличении: добавляем 1 к каждому измерению

// - При уменьшении: вычитаем 1, но не меньше 0 (защита от отрицательных значений)

int newRows = increase ? oldRows + 1 : Math.Max(oldRows - 1, 0);

int newCols = increase ? oldCols + 1 : Math.Max(oldCols - 1, 0);

// Создание нового массива с вычисленными размерами

double[,] newArray = new double[newRows, newCols];

// Определение границ копирования:

// Копируем столько строк и столбцов, сколько возможно (минимум из старого и нового размеров)

int copyRows = Math.Min(oldRows, newRows);

int copyCols = Math.Min(oldCols, newCols);

// Копирование данных из старого массива в новый

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

{

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

{

newArray[i, j] = array[i, j]; // Перенос элемента

}

}

// Замена исходного массива новым (работает благодаря передаче по ссылке)

array = newArray;

}

}

1. **Расчет тестовых примеров на ПК**

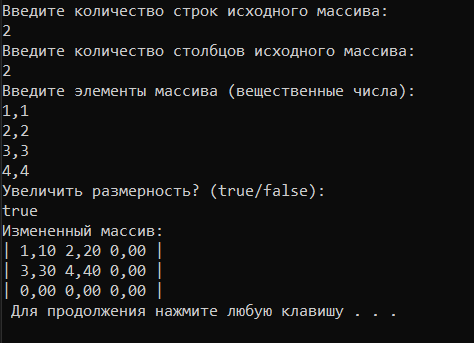
****

Рисунок 6.1.1 – Тестовый пример №1

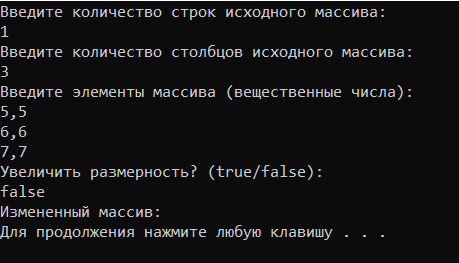
****

Рисунок 6.1.2 – Тестовый пример №2

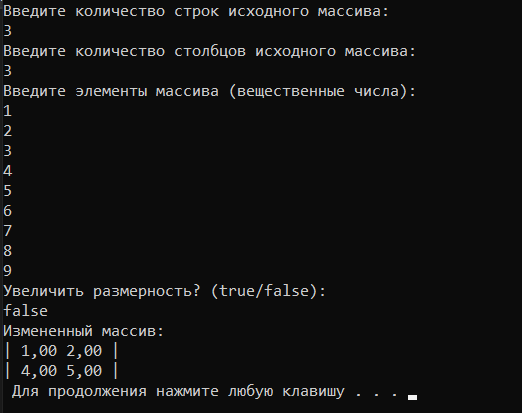


Рисунок 6.1.3 – Тестовый пример №3

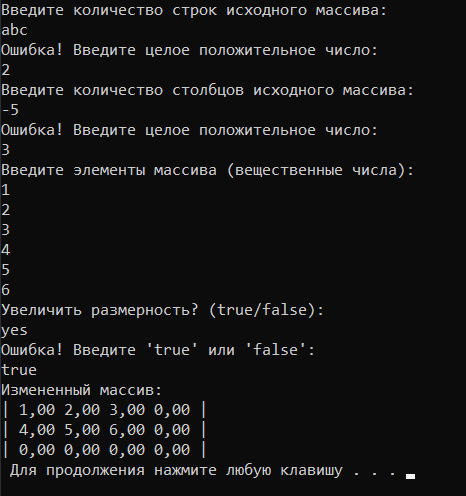


Рисунок 6.1.4 – Тестовый пример №4

1. **Вывод**

Разработано программное обеспечение, удовлетворяющее требованиям задания. Метод \_aResizeArray корректно изменяет размерность массива, сохраняя данные. Реализован контроль ввода и обработка граничных случаев. Тестовые примеры подтверждают работоспособность кода.