Министерство образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Пермский национальный исследовательский политехнический университет»

Кафедра «Информационные технологии и автоматизированные системы»

Отчёт

по лабораторной работе №5

Дисциплина: Объектно-ориентированное программирование.

Тема: функции и массивы.

Выполнил:

Cтудент группы ИВТ-20-2б

Тедеев Александр Зурабович

Проверила:

Доцент кафедры ИТАС

Викентьева Ольга Леонидовна

Пермь 2021

**Постановка задачи**

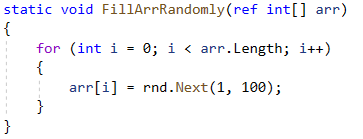
1. Сформировать динамический одномерный массив, заполнить его случайными числами и вывести на печать.
2. Выполнить указанное в варианте задание и вывести полученный массив на печать.
3. Сформировать динамический двумерный массив, заполнить его случайными числами и вывести на печать.
4. Выполнить указанное в варианте задание и вывести полученный массив на печать.
5. Сформировать динамический двумерный массив, заполнить его случайными числами и вывести на печать.
6. Выполнить указанное в варианте задание и вывести полученный массив на печать.
7. При реализации функций необходимо продемонстрировать использование параметров разных типов и различные способы организации функций (параметры по умолчанию, перегрузку функций, и т.д.)



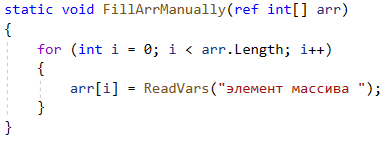
**Анализ**

Для выполнения первого задания необходимо организовать заполнение массива через ДСЧ и ручной ввод:

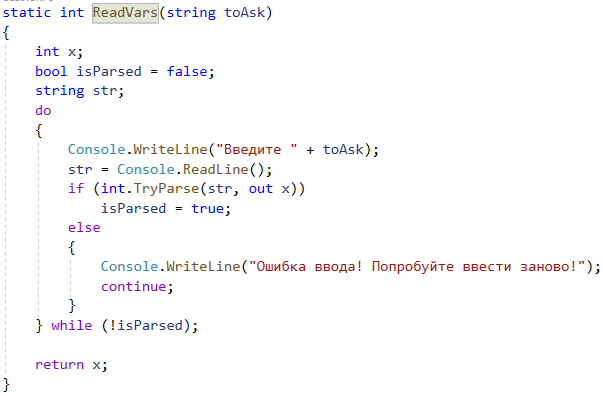
* 1. Функция заполнения одномерного массива с помощью ДСЧ.



* 1. Функция ручного заполнения одномерного массива.

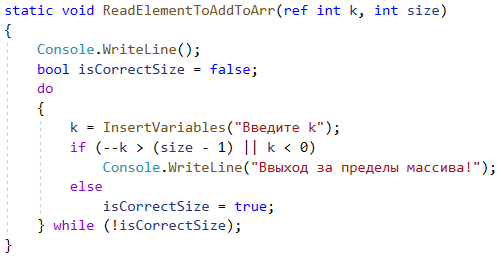


* 1. Функция ReadVars() необходима для проверки ввода элементов массива.

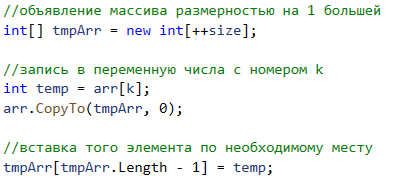


Добавление элемента с номером k в конец массива:

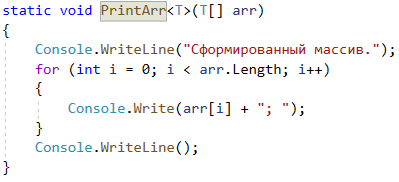
* 1. Ввод элемента с номером k производится через функцию ReadElementToAddToArr(). Данная функция повторяет действие функции ReadVars() за исключением проверки вхождения в диапазон массива.



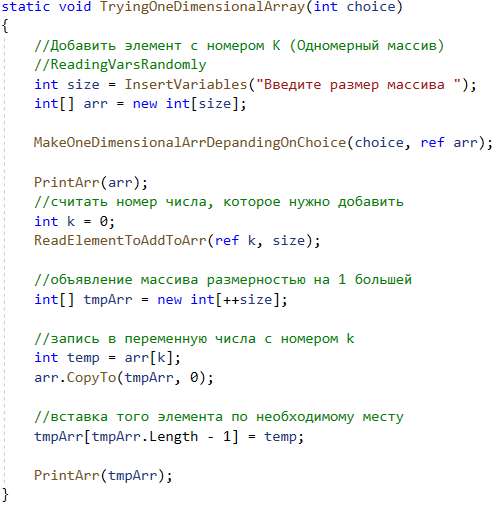
* 1. Чтобы добавить k элемент в массив, создать ещё один массив (tempArr[]), размерность которого больше на 1. Объявить временную переменную (temp), в которую записать элемент по k-индексу. С помощью метода CopyTo() скопировать данные из изначального массива (arr) в новый (tempArr) и записать на последнеее место k-элемент.



* 1. Печать массива – шаблонная функция PrintArr().

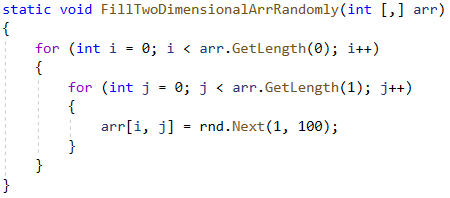


* 1. Полная функция обработки первого задания – TryingOneDimensionalArray().

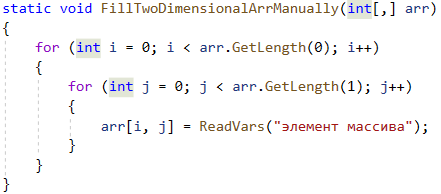


1. Для выполнения второго задания необходимо организовать заполнение двумерного массива через ДСЧ и ручной ввод:

* 1. Заполнение двумерного массива с помощью ДСЧ.

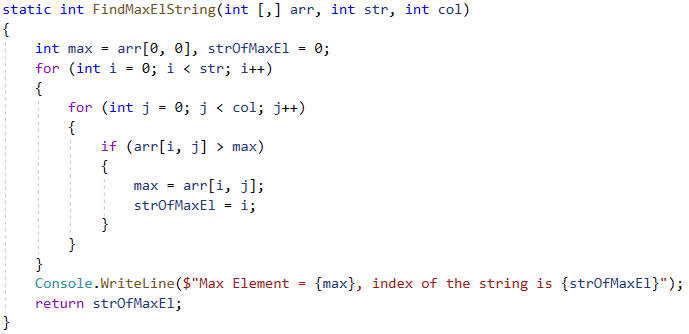


* 1. Ручное заполнение двумерного массива.

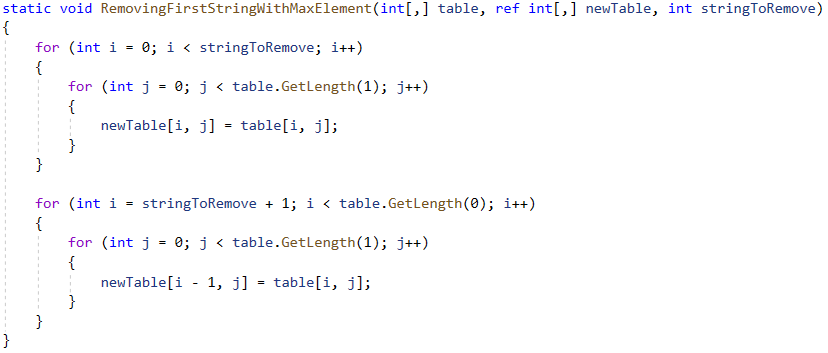


Основное задание разбивается на 2 этапа: найти номер строки с максимальным элементом (если такой элемент встречается несколько раз, то запомнить номер первого вхождения), а также удаление строки с данным номером.

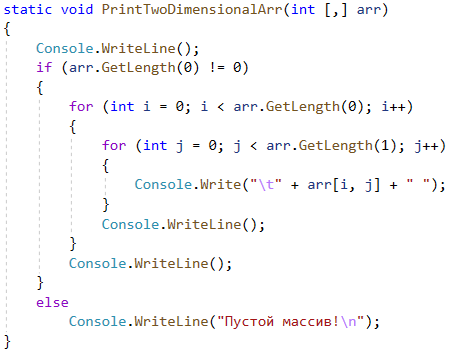
* 1. Функция FindMaxElString() используется для поиска номера строки с максимальным элементом в матрице.



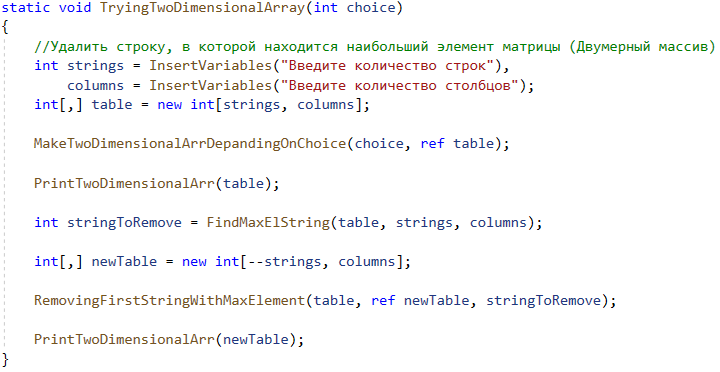
* 1. Функция RemovingFirstStringWithMaxElement() используется для удаления строки с максимальным элементом.



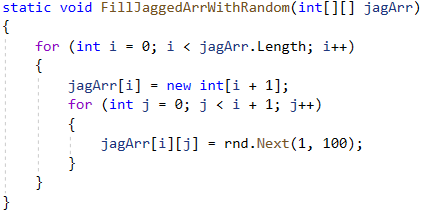
* 1. Функция печати матрицы в консоль – PrintTwoDimensionalArr().



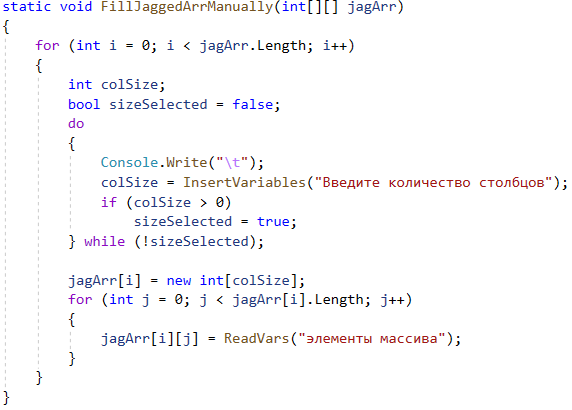
* 1. Полная функция обработки второго задания TryingTwoDimensionalArray().



1. Для выполнения второго задания необходимо организовать заполнение рваного массива через ДСЧ и ручной ввод:
   1. Заполнение рваного массива с помощью ДСЧ.

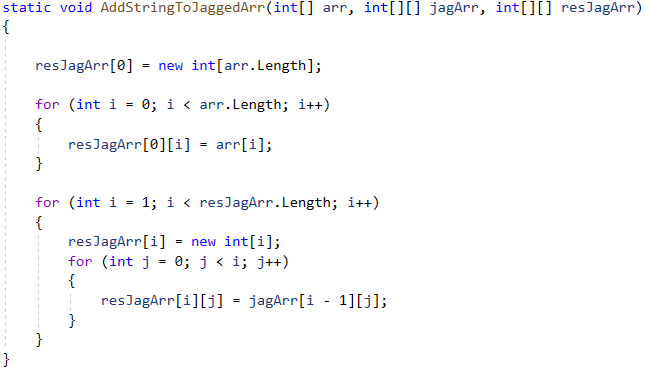


* 1. Ручное заполнение рваного массива. Заполнение происходит лесенкой: в первой строке 1 элемент, во второй строке 2 элемента, в 3 строке 3 элемента и т.п.

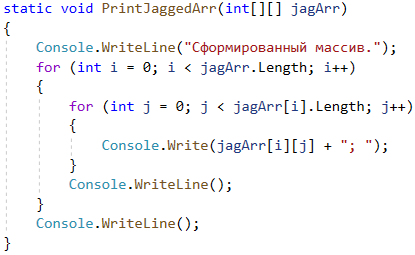


Чтобы выполнить задание необходимо ввести строку, которая является одномерным массивом. Для ввода такой строки используется функция FillArrRandomly(); для добавление данной строки используется функция AddStringToJaggedArr().

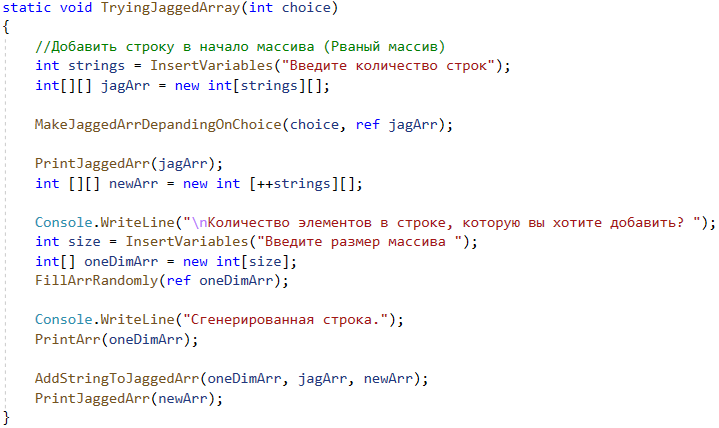
* 1. Функция AddStringToJaggedArr().



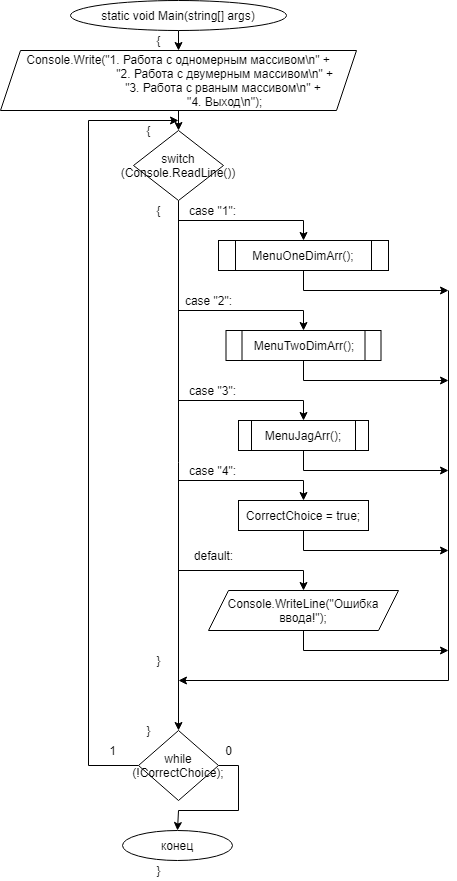
* 1. Печать рваного массива в консоль.

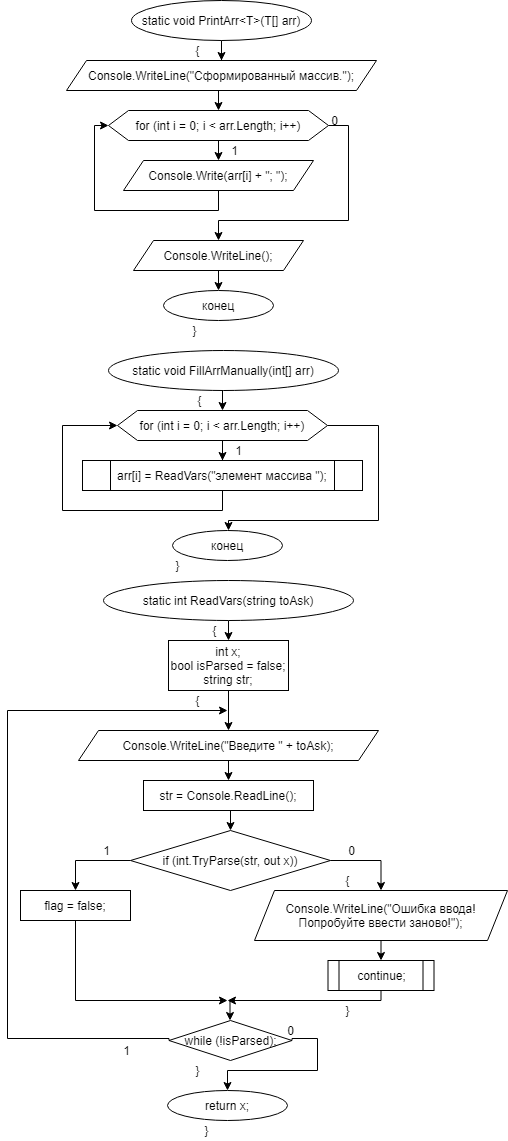


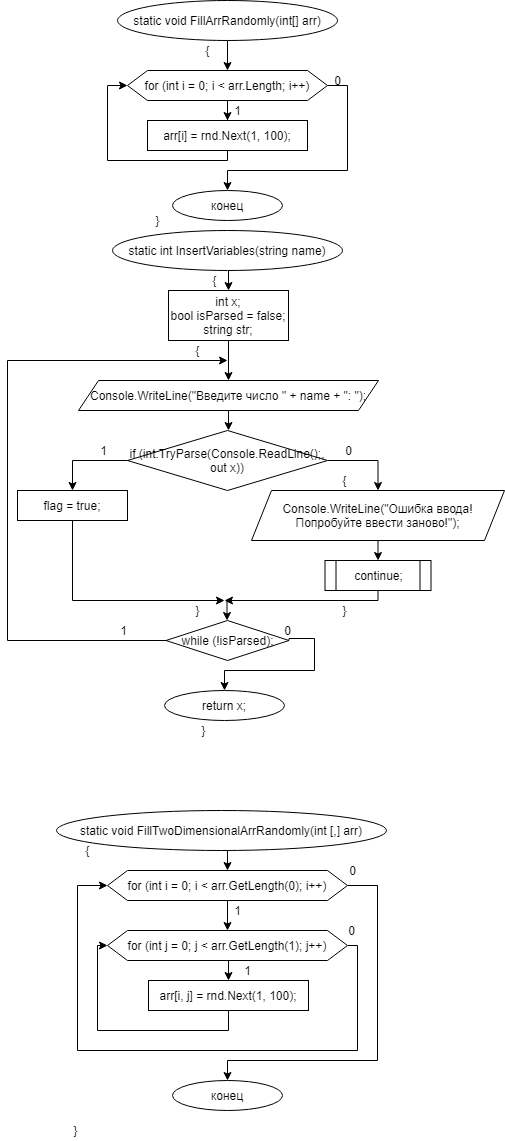
* 1. Полная функция обработки 3 задания TryingJaggedArray().

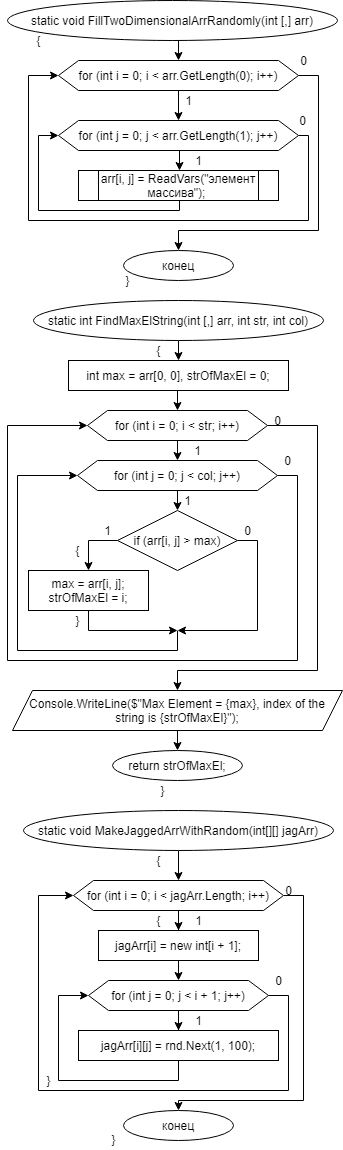


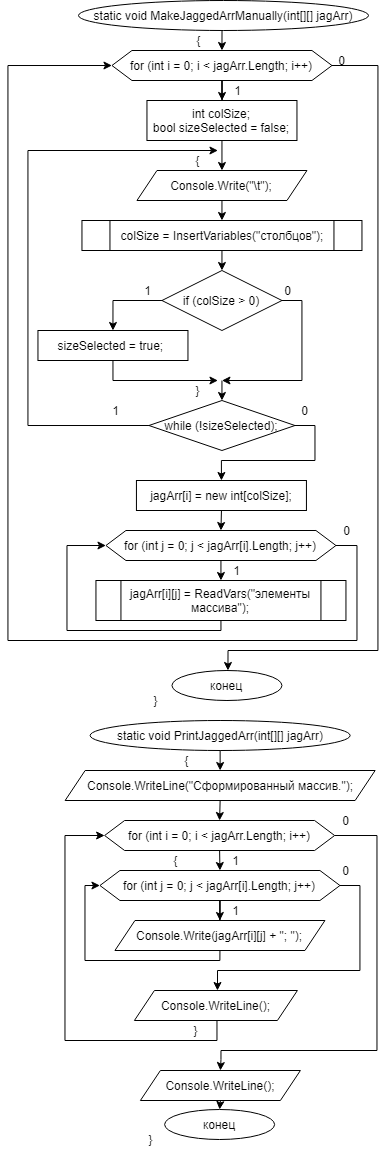
**Блок-схема**

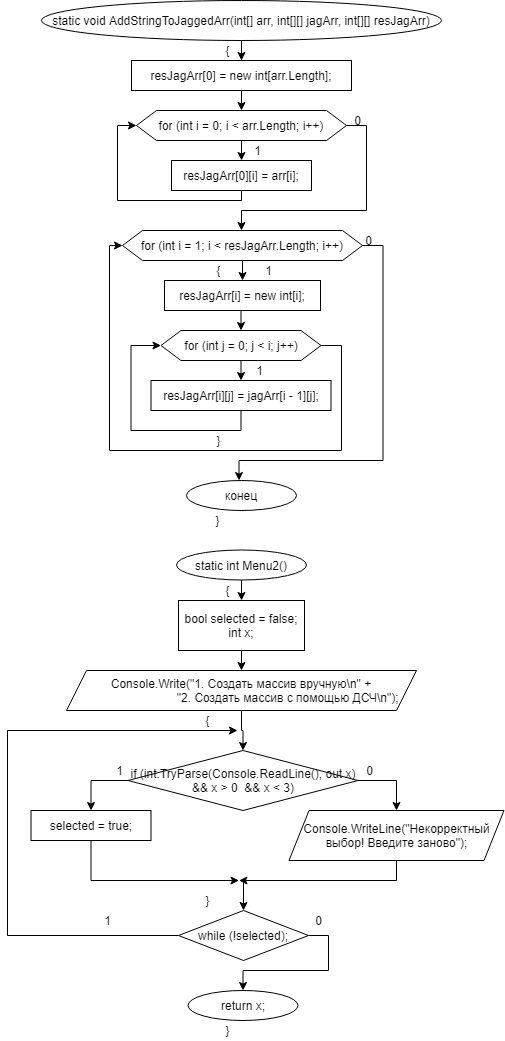


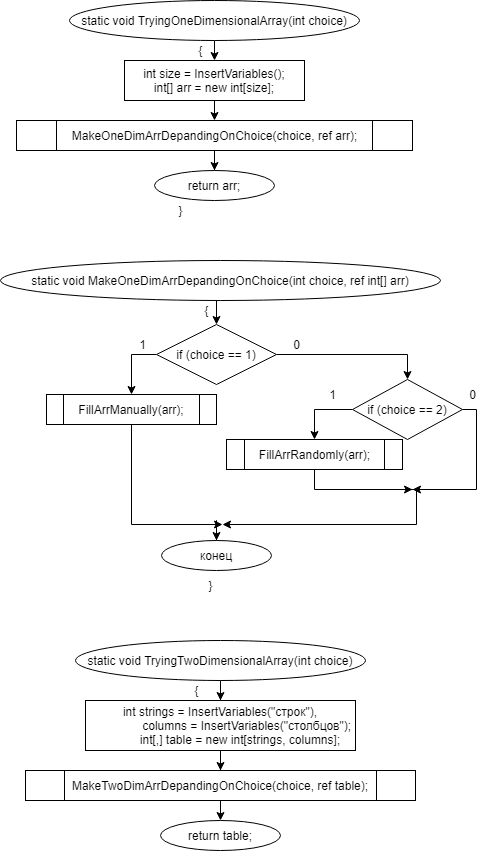


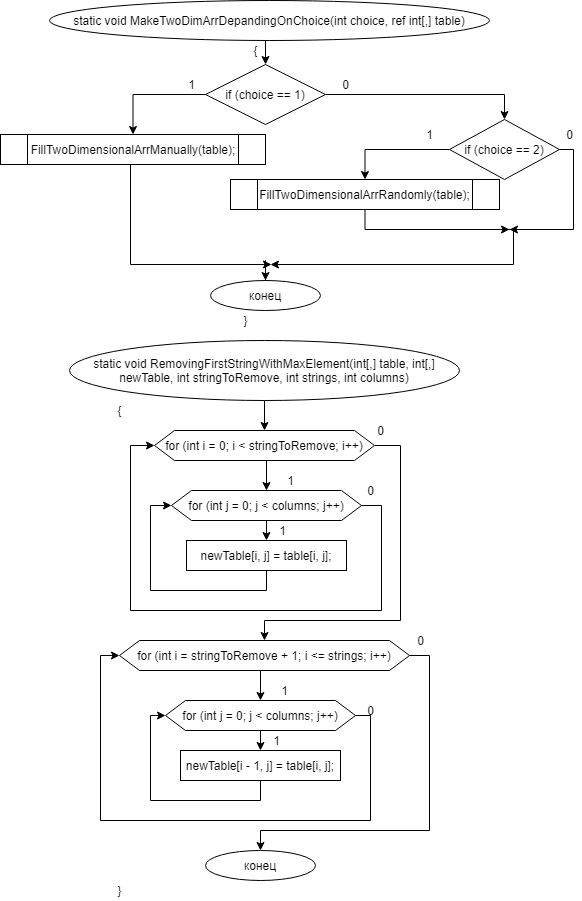


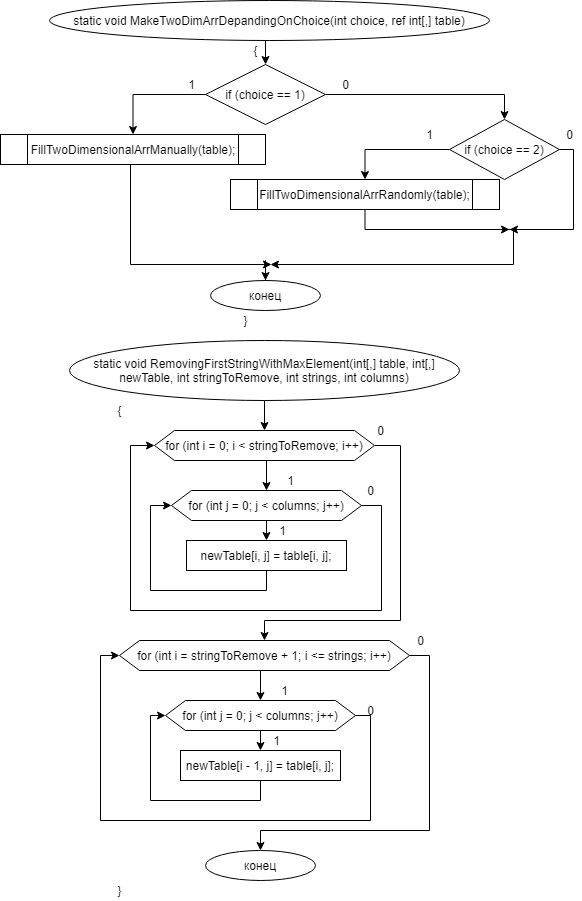












**Код**

using System;

using System.Collections.Generic;

using System.Collections.ObjectModel;

using System.Linq;

using System.Net;

using System.Runtime.ExceptionServices;

using System.Security.Cryptography.X509Certificates;

using System.Text;

using System.Threading.Tasks;

namespace Lab5

{

class Program

{

static Random rnd = new Random();

static void LgjH2m5c8emE66pjdExmgep47BAdKTrCJ7oice(int choice, ref int[] arr)

{

if (choice == 1)

FillArrManually(ref arr);

else if (choice == 2)

FillArrRandomly(ref arr);

}

static void TryingOneDimensionalArray(int choice)

{

//Добавить элемент с номером К (Одномерный массив)

//ReadingVarsRandomly

int size = InsertVariables("Введите размер массива ");

int[] arr = new int[size];

LgjH2m5c8emE66pjdExmgep47BAdKTrCJ7oice(choice, ref arr);

PrintArr(arr);

//считать номер числа, которое нужно добавить

int k = 0;

ReadElementToAddToArr(ref k, size);

//объявление массива размерностью на 1 большей

int[] tmpArr = new int[++size];

//запись в переменную числа с номером k

int temp = arr[k];

arr.CopyTo(tmpArr, 0);

//вставка того элемента по необходимому месту

tmpArr[tmpArr.Length - 1] = temp;

PrintArr(tmpArr);

}

static void LgjH2m5c8emE66pjdExmgep47BAdKTrCJ7oice(int choice, ref int[,] table)

{

if (choice == 1)

FillTwoDimensionalArrManually(table);

else if (choice == 2)

FillTwoDimensionalArrRandomly(table);

}

static void TryingTwoDimensionalArray(int choice)

{

//Удалить строку, в которой находится наибольший элемент матрицы (Двумерный массив)

int strings = InsertVariables("Введите количество строк"),

columns = InsertVariables("Введите количество столбцов");

int[,] table = new int[strings, columns];

LgjH2m5c8emE66pjdExmgep47BAdKTrCJ7oice(choice, ref table);

PrintTwoDimensionalArr(table);

int stringToRemove = FindMaxElString(table, strings, columns);

int[,] newTable = new int[--strings, columns];

RemovingFirstStringWithMaxElement(table, ref newTable, stringToRemove);

PrintTwoDimensionalArr(newTable);

}

static void TryingJaggedArray(int choice)

{

//Добавить строку в начало массива (Рваный массив)

int strings = InsertVariables("Введите количество строк");

int[][] jagArr = new int[strings][];

MakeJaggedArrDepandingOnChoice(choice, ref jagArr);

PrintJaggedArr(jagArr);

int [][] newArr = new int [++strings][];

Console.WriteLine("\nКоличество элементов в строке, которую вы хотите добавить? ");

int size = InsertVariables("Введите размер массива ");

int[] oneDimArr = new int[size];

FillArrRandomly(ref oneDimArr);

Console.WriteLine("Сгенерированная строка.");

PrintArr(oneDimArr);

AddStringToJaggedArr(oneDimArr, jagArr, newArr);

PrintJaggedArr(newArr);

}

static void Main(string[] args)

{

Menu();

}

static void Menu()

{

bool CorrectChoice = false;

int n;

do

{

Console.Write("1. Работа с одномерным массивом\n" +

"2. Работа с двумерным массивом\n" +

"3. Работа с рваным массивом\n" +

"4. Выход\n");

if (int.TryParse(Console.ReadLine(), out n))

switch (n)

{

case 1:

{

int choice = DecideRandomOrManualInput();

TryingOneDimensionalArray(choice);

break;

}

case 2:

{

int choice = DecideRandomOrManualInput();

TryingTwoDimensionalArray(choice);

break;

}

case 3:

{

int choice = DecideRandomOrManualInput();

TryingJaggedArray(choice);

break;

}

case 4:

{

CorrectChoice = true;

break;

}

default:

{

Console.WriteLine("Ошибка ввода!");

continue;

}

}

else

{

Console.WriteLine("Ошибка ввода!");

continue;

}

} while (!CorrectChoice);

}

static int DecideRandomOrManualInput()

{

bool selected = false;

int x;

Console.Write("1. Создать массив вручную\n" +

"2. Создать массив с помощью ДСЧ\n");

do

{

if (int.TryParse(Console.ReadLine(), out x) && x > 0 && x < 3)

selected = true;

else

Console.WriteLine("Некорректный выбор! Введите заново");

} while (!selected);

return x;

}

static void RemovingFirstStringWithMaxElement(int[,] table, ref int[,] newTable, int stringToRemove)

{

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

{

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

{

newTable[i, j] = table[i, j];

}

}

for (int i = stringToRemove + 1; i < table.GetLength(0); i++)

{

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

{

newTable[i - 1, j] = table[i, j];

}

}

}

static void AddStringToJaggedArr(int[] arr, int[][] jagArr, int[][] resJagArr)

{

resJagArr[0] = new int[arr.Length];

for (int i = 0; i < arr.Length; i++)

{

resJagArr[0][i] = arr[i];

}

for (int i = 1; i < resJagArr.Length; i++)

{

resJagArr[i] = new int[i];

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

{

resJagArr[i][j] = jagArr[i - 1][j];

}

}

}

static void PrintJaggedArr(int[][] jagArr)

{

Console.WriteLine("Сформированный массив.");

for (int i = 0; i < jagArr.Length; i++)

{

for (int j = 0; j < jagArr[i].Length; j++)

{

Console.Write("\t" + jagArr[i][j] + "; ");

}

Console.WriteLine();

}

Console.WriteLine();

}

static void MakeJaggedArrDepandingOnChoice(int choice, ref int[][] jagArr)

{

if (choice == 1)

FillJaggedArrManually(jagArr);

else if (choice == 2)

FillJaggedArrWithRandom(jagArr);

}

static void FillJaggedArrManually(int[][] jagArr)

{

for (int i = 0; i < jagArr.Length; i++)

{

int colSize = InsertVariables("Введите количество столбцов");

jagArr[i] = new int[colSize];

for (int j = 0; j < jagArr[i].Length; j++)

{

jagArr[i][j] = ReadVars("элементы массива");

}

}

}

static void FillJaggedArrWithRandom(int[][] jagArr)

{

for (int i = 0; i < jagArr.Length; i++)

{

jagArr[i] = new int[i + 1];

for (int j = 0; j < i + 1; j++)

{

jagArr[i][j] = rnd.Next(1, 100);

}

}

}

static void PrintTwoDimensionalArr(int [,] arr)

{

Console.WriteLine();

if (arr.GetLength(0) != 0)

{

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

{

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

{

Console.Write("\t" + arr[i, j] + " ");

}

Console.WriteLine();

}

Console.WriteLine();

}

else

Console.WriteLine("Пустой массив!\n");

}

static int FindMaxElString(int [,] arr, int str, int col)

{

int max = arr[0, 0], strOfMaxEl = 0;

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

{

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

{

if (arr[i, j] > max)

{

max = arr[i, j];

strOfMaxEl = i;

}

}

}

Console.WriteLine($"Max Element = {max}, index of the string is {strOfMaxEl}");

return strOfMaxEl;

}

static void FillTwoDimensionalArrManually(int[,] arr)

{

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

{

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

{

arr[i, j] = ReadVars("элемент массива");

}

}

}

static void FillTwoDimensionalArrRandomly(int [,] arr)

{

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

{

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

{

arr[i, j] = rnd.Next(1, 100);

}

}

}

static void ReadElementToAddToArr(ref int k, int size)

{

Console.WriteLine();

bool isCorrectSize = false;

do

{

k = InsertVariables("Введите k");

if (--k > (size - 1) || k < 0)

Console.WriteLine("Ввыход за пределы массива!");

else

isCorrectSize = true;

} while (!isCorrectSize);

}

static int ReadVars(string toAsk)

{

int x;

bool isParsed = false;

string str;

do

{

Console.WriteLine("Введите " + toAsk);

str = Console.ReadLine();

if (int.TryParse(str, out x))

isParsed = true;

else

{

Console.WriteLine("Ошибка ввода! Попробуйте ввести заново!");

continue;

}

} while (!isParsed);

return x;

}

static int InsertVariables(string toAsk)

{

int x;

bool isParsed = false;

do

{

Console.WriteLine(toAsk);

if (int.TryParse(Console.ReadLine(), out x))

if (x > 0)

isParsed = true;

else

continue;

else

{

Console.WriteLine("Ошибка ввода! Попробуйте ввести заново!");

continue;

}

} while (!isParsed);

return x;

}

static void FillArrRandomly(ref int[] arr)

{

for (int i = 0; i < arr.Length; i++)

{

arr[i] = rnd.Next(1, 100);

}

}

static void FillArrManually(ref int[] arr)

{

for (int i = 0; i < arr.Length; i++)

{

arr[i] = ReadVars("элемент массива ");

}

}

static void PrintArr<T>(T[] arr)

{

Console.WriteLine("Сформированный массив.");

for (int i = 0; i < arr.Length; i++)

{

Console.Write(arr[i] + "; ");

}

Console.WriteLine();

}

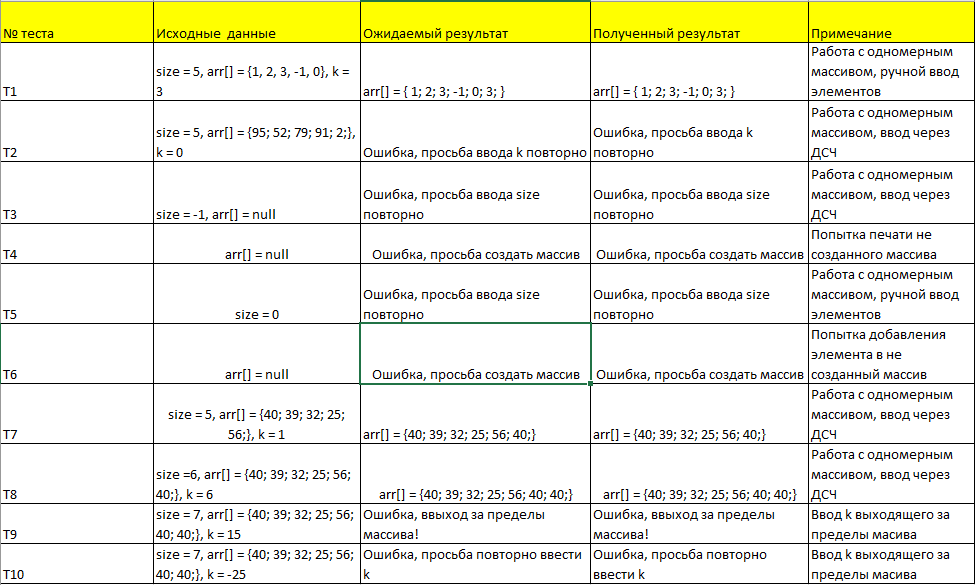
}

}

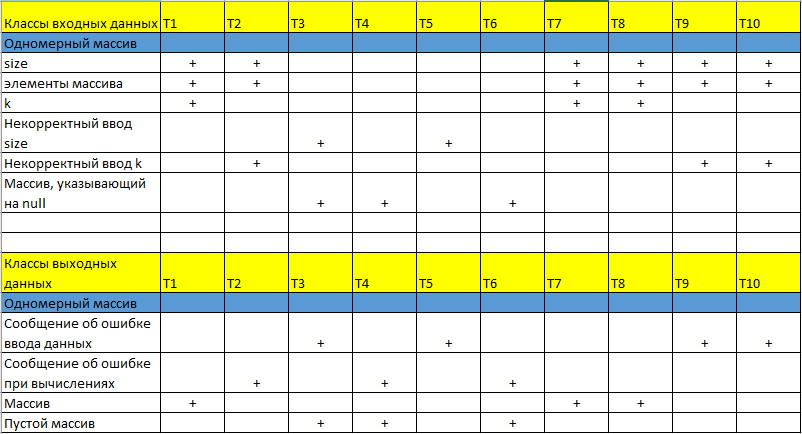
**Тесты**

**Задача 1**

**Тесты**



**Чёрный ящик**

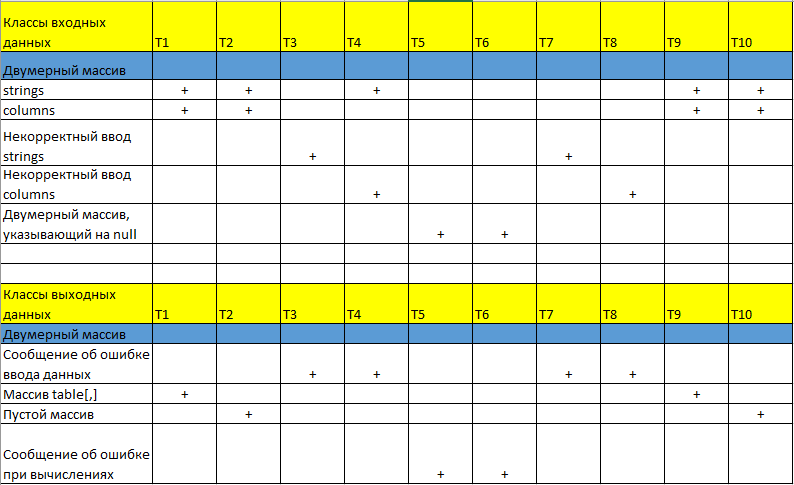


**Задача 2**

**Тесты**



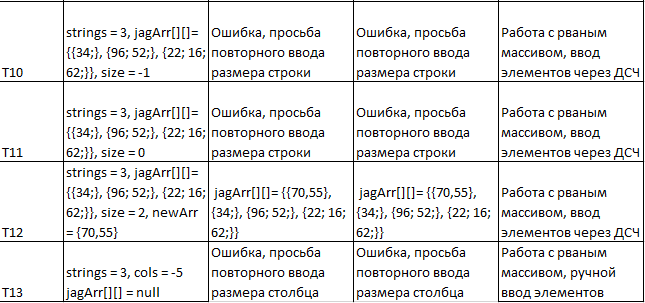
**Чёрный ящик**



**Задача 3**

**Тесты**





**Чёрный ящик**

