Министерство образования и науки 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

Вариант 15

Выполнил работу

студент группы РИС-22-1б

Зырянов Ростислав

Проверила

Доцент кафедры ИТАС

Викентьева О.Л.

Пермь, 2023

**Постановка задачи**

1. Сформировать динамический одномерный массив, заполнить его случайными числами и вывести на печать.
2. Сформировать динамический двумерный массив, заполнить его случайными числами и вывести на печать.
3. Сформировать динамический двумерный массив, заполнить его случайными числами и вывести на печать.
4. Выполнить задания для массивов
   1. Для одномерного - добавить К элементов, начиная с номера N
   2. Для двумерного - удалить все четные строки
   3. Для рваного - добавить строку в конец массива

**Анализ задачи**

Для выполнения лабораторной работы нужно выполнить 3 задания, каждое из которых относится к своему виду массива. Во всех заданиях будет использоваться ввод данных с консоли, который необходимо проверять на корректность.

При реализации 1 и 3 задания, нужно учесть, что работать надо с номерами, а не с индексами, из-за чего в условиях будет присутствовать -1 для номера строки, чтобы работать с индексом.

Все массивы будут иметь ручное и рандомное заполнение, вывод данных, а также выполнение задания, которое к ним относится.

При создании массивов нужно предусмотреть, что длина, введенная пользователем, должна быть положительным целочисленным значением, в противном случае, вывести сообщение об ошибке.

**Описание этапа проектирования**

Методы с названиями RandomFill и ManualFill для всех 3-х классов, каждый из которых представляет реализацию методов для определенного вида массива, будут называться одинаково. Поэтому для вызова их в пункте меню необходимо реализовать базовый класс, от которого они будут наследованы, а в меню передавать их параметрами в метод, в котором объект базового класса будет ссылаться на них, чтобы при помощи полиморфизма воспользоваться соответствующими методами переданного класса.

**Работа программы**

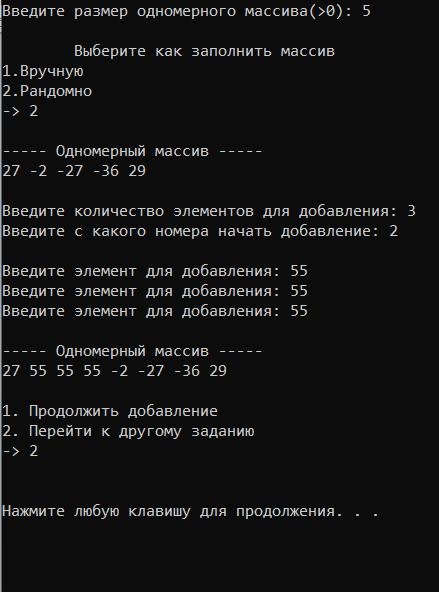


Рисунок 1 – Работа с одномерным массивом.

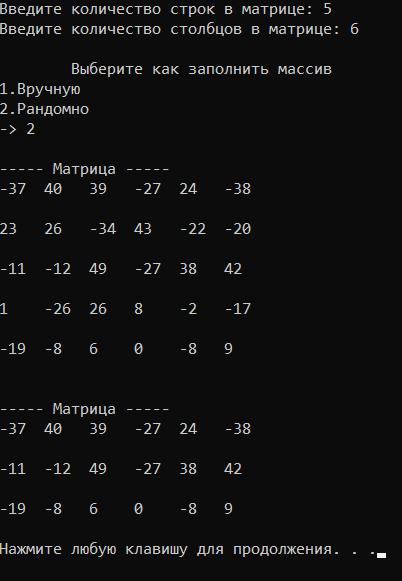


Рисунок 2 – Работа с двумерным массивом.

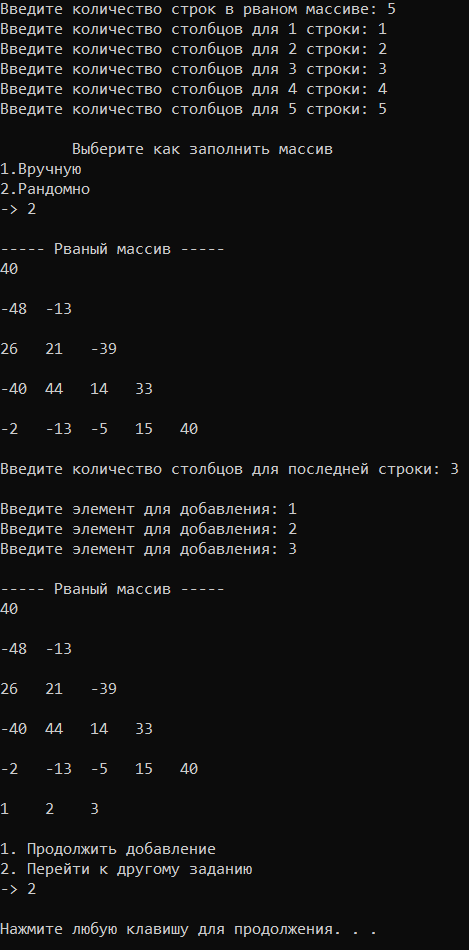


Рисунок 3 – Работа с рваным массивом.

**Тестирование программы**

|  |  |  |  |
| --- | --- | --- | --- |
| Тестирование | | | |
| Номер | Тест | Предполагаемый результат | Полученные результат |
| **Меню** | | | |
| Т1 | choice = 1 | Переход к заполнению одномерного массива | + |
| Т2 | choice = 2 | Переход к заполнению матрицы | + |
| Т3 | choice = 3 | Переход заполнению рваного массива | + |
| Т4 | choice = 4 | Выход из программы | + |
| Т5 | choice = "abc" | Некорректный ввод числа! | + |
| Т6 | choice = 5 | Такого пункта меню не существует! | + |
| Т7 | choice = 1,1 | Некорректный ввод числа! | + |
| **Одномерный массив** | | | |
| Т8 | choice = 1 | Заполнение вручную | + |
| Т9 | choice = 2 | Заполнение рандомно | + |
| Т10 | choice = "abc" | Некорректный ввод числа! | + |
| Т11 | choice = 3 | Такого пункта меню не существует! | + |
| Т12 | choice = 1,1 | Некорректный ввод числа! | + |
| Т13 | length = 3 | Массив успешно создан | + |
| Т14 | length = 0 | Количество элементов должно быть положительным! | + |
| Т15 | length = -1 | Количество элементов должно быть положительным! | + |
| Т16 | length = 1,1 | Некорректный ввод числа! |  |
| Т17 | length = "abc" | Некорректный ввод числа! | + |
| Т18 | el = "abc" | Некорректный ввод числа! | + |
| Т19 | el = -6 | Добавление елемента в массив вручную | + |
| Т20 | countElements = 2 | Переход к вводу номера, с которого начать удаление | + |
| Т21 | countElements = 4 | Количество элементов для удаления не входит в диапазон массива! | + |
| Т22 | countElements = "abc" | Некорректный ввод числа! | + |
| Т23 | countElements = 1,1 | Некорректный ввод числа! | + |
| Т24 | itemNumber = 1 | Останется массив с 1 элементом | + |
| Т25 | itemNumber = 3 | Невозможно удалить 2 элемента(-ов) с этого номера! | + |
| Т26 | itemNumber = "abc" | Некорректный ввод числа! | + |
| Т27 | itemNumber = 1,1 | Некорректный ввод числа! | + |
| **Двумерный массив** | | | |
| Т28 | choice = 1 | Заполнение вручную | + |
| Т29 | choice = 2 | Заполнение рандомно | + |
| Т30 | choice = "abc" | Некорректный ввод числа! | + |
| Т31 | choice = 3 | Такого пункта меню не существует! | + |
| Т32 | choice = 1,1 | Некорректный ввод числа! | + |
| Т33 | countRows = 3 | Переход к заполнению количества колонок | + |
| Т34 | countRows = 0 | Количество строк должно быть положительным числом! | + |
| Т35 | countRows = -1 | Количество строк должно быть положительным числом! | + |
| Т36 | countRows = 1,1 | Некорректный ввод числа! | + |
| Т37 | countRows = "abc" | Некорректный ввод числа! | + |
| Т38 | countCols = 3 | Матрица успешно создана | + |
| Т39 | countCols = 0 | Количество строк должно быть положительным числом! | + |
| Т40 | countCols = -1 | Количество строк должно быть положительным числом! | + |
| Т41 | countCols = 1,1 | Некорректный ввод числа! | + |
| Т42 | countCols = "abc" | Некорректный ввод числа! | + |
| Т43 | el = "abc" | Некорректный ввод числа! | + |
| Т44 | el = -6 | Добавление елемента в матрицу вручную | + |
| **Рванный массив** | | | |
| Т45 | choice = 1 | Заполнение вручную | + |
| Т46 | choice = 2 | Заполнение рандомно | + |
| Т47 | choice = "abc" | Некорректный ввод числа! | + |
| Т48 | choice = 3 | Такого пункта меню не существует! | + |
| Т49 | choice = 1,1 | Некорректный ввод числа! | + |
| Т50 | countRows = 3 | Переход к заполнению количества колонок | + |
| Т51 | countRows = 0 | Количество строк должно быть положительным числом! | + |
| Т52 | countRows = -1 | Количество строк должно быть положительным числом! | + |
| Т53 | countRows = 1,1 | Некорректный ввод числа! | + |
| Т54 | countRows = "abc" | Некорректный ввод числа! | + |
| Т55 | countCols = 3 | Количество колонок для 1й строки | + |
| Т56 | countCols = 4 | Количество колонок для 2й строки | + |
| Т57 | countCols = 5 | Количество колонок для 3й строки | + |
| Т58 | countCols = 0 | Количество строк должно быть положительным числом! | + |
| Т59 | countCols = -1 | Количество строк должно быть положительным числом! | + |
| Т60 | countCols = 1,1 | Некорректный ввод числа! | + |
| Т61 | countCols = "abc" | Некорректный ввод числа! | + |
| Т62 | el = "abc" | Некорректный ввод числа! | + |
| Т63 | el = -6 | Добавление елемента в матрицу вручную | + |

Таблица 1 – Тестирование программы

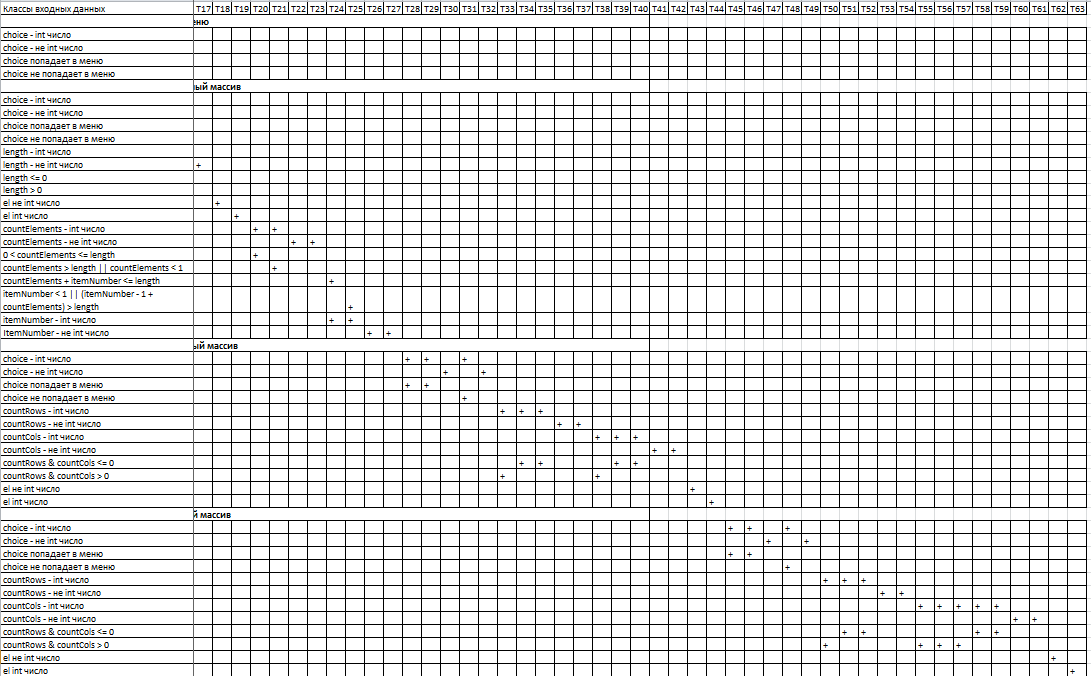


Таблица 2 – Тестирование программы черный ящик.

**Код программы**

**Файл - Univariate.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Laba\_5

{

internal class Univariate : ArrayBase

{

int[] array;

public Univariate()

{

string msg = "Введите размер одномерного массива(>0): ";

string errorMsg = "\nКоличество элементов должно быть положительным!\n";

array = new int[CheckAndInput.InputPositiveNumber(msg, errorMsg)];

Console.WriteLine();

}

public override void RandomFill()

{

Random random = new Random();

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

array[i] = random.Next(-50, 50);

}

public override void ManualFill()

{

string? msg;

Console.WriteLine();

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

{

msg = $"Введите {i + 1}-й элемент: ";

array[i] = CheckAndInput.InputIntNumber(msg);

}

}

public void ShowArrayObj()

{

if (array.Length == 0)

{

Console.WriteLine("Массив: <пустой>\n");

return;

}

Console.WriteLine("----- Одномерный массив -----");

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

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

Console.WriteLine('\n');

}

public void ShowArray(int[] array)

{

if (array.Length == 0)

{

Console.WriteLine("Массив: <пустой>\n");

return;

}

Console.WriteLine("----- Одномерный массив -----");

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

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

Console.WriteLine('\n');

}

public void AddElements()

{

int countElements = 0, itemNumber = 0;

string msg = "";

string errorMsg = "";

msg = "Введите количество элементов для добавления: ";

errorMsg = "\nКоличество элементов для добавление должно быть положительным числом!\n";

countElements = CheckAndInput.InputPositiveNumber(msg, errorMsg);

msg = "Введите с какого номера начать добавление: ";

errorMsg = "\nНевозможно начать добавление элементов с этого номера!\n";

itemNumber = CheckAndInput.InputWithinBoundariesArray(array, msg, errorMsg);

msg = "1. Продолжить добавление\n2. Перейти к другому заданию\n-> ";

errorMsg = "\nТакого пункта меню не существует!\n";

int[] tempArray = new int[array.Length + countElements];

AddToTemp(itemNumber, countElements, tempArray);

array = tempArray;

Console.WriteLine();

ShowArray(array);

do

{

int choice = CheckAndInput.InputIntNumber(msg);

Console.WriteLine();

if (choice == 1)

{

AddElements();

break;

}

else if (choice == 2)

break;

else

Console.WriteLine(errorMsg);

} while (true);

}

public void AddToTemp(int itemNumber, int countElements, int[] tempArray)

{

Console.WriteLine();

string msg = "Введите элемент для добавления: ";

for (int i = 0, j = 0; i < tempArray.Length; i++)

{

if (i < itemNumber - 1)

tempArray[i] = array[j++];

else if (i < itemNumber + countElements - 1)

tempArray[i] = CheckAndInput.InputIntNumber(msg);

else

tempArray[i] = array[j++];

}

}

}

}

**Файл – Bivariate.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Laba\_5

{

internal class Bivariate : ArrayBase

{

int[,] matrix;

public Bivariate()

{

int countRows = 0, countCols = 0;

string msg = "";

string errorMsg = "";

msg = "Введите количество строк в матрице: ";

errorMsg = "\nКоличество строк должно быть положительным числом!\n";

countRows = CheckAndInput.InputPositiveNumber(msg, errorMsg);

msg = "Введите количество столбцов в матрице: ";

errorMsg = "\nКоличество колонок должно быть положительным числом!\n";

countCols = CheckAndInput.InputPositiveNumber(msg, errorMsg);

matrix = new int[countRows, countCols];

Console.WriteLine();

}

public override void RandomFill()

{

Random random = new Random();

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

{

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

matrix[i, j] = random.Next(-50, 50);

}

}

public override void ManualFill()

{

string? msg;

Console.WriteLine();

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

{

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

{

msg = $"Введите элемент для {i + 1}-й строки {j + 1}-го столбца: ";

matrix[i, j] = CheckAndInput.InputIntNumber(msg);

}

}

}

public void ShowMatrix()

{

if (matrix.Length == 0)

{

Console.WriteLine("Матрица: <пустая>\n");

return;

}

Console.WriteLine("----- Матрица -----");

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

{

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

Console.Write($"{matrix[i, j], -4} ");

Console.WriteLine('\n');

}

}

public void DeleteEvenRows()

{

int count = 0;

if (matrix.GetLength(0) % 2 == 0)

count = matrix.GetLength(0) / 2;

else

count = matrix.GetLength(0) / 2 + 1

;

int[,] tempArray = new int[count, matrix.GetLength(1)];

for (int i = 0, z = 0; i < matrix.GetLength(0); i++)

{

if ((i + 1) % 2 != 0)

{

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

tempArray[z, j] = matrix[i, j];

z++;

}

}

matrix = tempArray;

Console.WriteLine();

}

}

}

**Файл – Ragged.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Laba\_5

{

internal class Ragged : ArrayBase

{

int[][] ragged;

public Ragged()

{

int countRows = 0, countCols = 0;

string msg = "";

string errorMsg = "";

msg = "Введите количество строк в матрице: ";

errorMsg = "\nКоличество строк должно быть положительным числом!\n";

countRows = CheckAndInput.InputPositiveNumber(msg, errorMsg);

ragged = new int[countRows][];

errorMsg = "\nКоличество колонок должно быть положительным числом!\n";

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

{

msg = $"Введите количество столбцов для {i + 1} строки: ";

countCols = CheckAndInput.InputPositiveNumber(msg, errorMsg);

ragged[i] = new int[countCols];

}

Console.WriteLine();

}

public override void RandomFill()

{

Random random = new Random();

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

{

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

ragged[i][j] = random.Next(-50, 50);

}

}

public override void ManualFill()

{

string? msg;

Console.WriteLine();

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

{

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

{

msg = $"Введите элемент для {i + 1}-й строки {j + 1}-го столбца: ";

ragged[i][j] = CheckAndInput.InputIntNumber(msg);

}

}

}

public void ShowRaggedObj()

{

if (ragged.Length == 0)

{

Console.WriteLine("Рваный массив: <пустой>\n");

return;

}

Console.WriteLine("----- Рваный массив -----");

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

{

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

Console.Write($"{ragged[i][j],-4} ");

Console.WriteLine('\n');

}

}

public void ShowRagged(int[][] ragged)

{

if (ragged.Length == 0)

{

Console.WriteLine("Рваный массив: <пустой>\n");

return;

}

Console.WriteLine("----- Рваный массив -----");

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

{

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

Console.Write($"{ragged[i][j],-4} ");

Console.WriteLine('\n');

}

}

public void AddRowToEnd()

{

int[][] tempArray = new int[ragged.Length + 1][];

for (int i = 0, j = 0; i < ragged.Length; i++)

tempArray[j++] = new int[ragged[i].Length];

string msg = "Введите количество столбцов для последней строки: ";

string errorMsg = "\nКоличество столбцов должно быть положительным числом!\n";

int countCols = CheckAndInput.InputPositiveNumber(msg, errorMsg);

tempArray[tempArray.GetLength(0) - 1] = new int[countCols];

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

{

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

tempArray[i][j] = ragged[i][j];

}

Console.WriteLine();

msg = "Введите элемент для добавления: ";

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

tempArray[tempArray.GetLength(0) - 1][j] = CheckAndInput.InputIntNumber(msg);

ragged = tempArray;

Console.WriteLine();

ShowRagged(ragged);

msg = "1. Продолжить добавление\n2. Перейти к другому заданию\n-> ";

errorMsg = "Такого пункта меню не существует!\n";

do

{

int choice = CheckAndInput.InputIntNumber(msg);

Console.WriteLine();

if (choice == 1)

{

AddRowToEnd();

break;

}

else if (choice == 2)

break;

else

Console.WriteLine(errorMsg);

} while (true);

}

}

}

**Файл – CheckAndInput.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Laba\_5

{

internal static class CheckAndInput

{

static public int InputIntNumber(string msg, string errorMsg = "\nНекорректный ввод числа!\n")

{

string? strEl;

int el = 0;

do

{

Console.Write(msg);

strEl = Console.ReadLine();

if (!int.TryParse(strEl, out el))

Console.WriteLine(errorMsg);

} while (!int.TryParse(strEl, out el));

return el;

}

static public int InputPositiveNumber(string msg, string errorMsg)

{

int el = 0;

do

{

el = CheckAndInput.InputIntNumber(msg);

if (el < 1)

Console.WriteLine(errorMsg);

} while (el < 1);

return el;

}

static public int InputWithinBoundariesArray(int[] array, string msg, string errorMsg) //ввод в границах массива

{

int el = 0;

do

{

el = CheckAndInput.InputIntNumber(msg);

if (el < 1 || el > array.Length + 1)

Console.WriteLine(errorMsg);

} while (el < 1 || el > array.Length + 1);

return el;

}

}

}

**Файл – Menu.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Laba\_5

{

internal class Menu

{

public Menu()

{

bool isNonExit = true;

int choice = 0;

string msg = "", errorMsg = "\nТакого пункта меню не существует!\n";

do

{

Console.Clear();

Console.WriteLine("----- Добро пожаловать в МЕНЮ программы! -----\n");

msg = "1. Одномерный массив. Добавить К элементов, начиная с номера N.\n" +

"2. Двумерный массив. Удалить все четные строки.\n" +

"3. Рваный массив. Добавить строку в конец массива.\n" +

"4. Завершить работу программы.\n-> ";

choice = CheckAndInput.InputIntNumber(msg);

switch (choice)

{

case 1:

Console.Clear(); Univariate array = new Univariate(); choiceFill(array); MakeAndShowChanges(array); break;

case 2:

Console.Clear(); Bivariate matrix = new Bivariate(); choiceFill(matrix); MakeAndShowChanges(matrix); break;

case 3:

Console.Clear(); Ragged ragged = new Ragged(); choiceFill(ragged); MakeAndShowChanges(ragged); break;

case 4:

isNonExit = false; break;

default:

Console.WriteLine(errorMsg);

break;

}

if (isNonExit)

{

Console.Write("Нажмите любую клавишу для продолжения. . .");

Console.ReadKey();

}

} while (isNonExit);

}

public void choiceFill(ArrayBase obj)

{

string msg = "\tВыберите как заполнить массив\n1.Вручную\n2.Рандомно\n-> ", errorMsg = "\nТакого пункта меню не существует!\n";

do

{

int choice = CheckAndInput.InputIntNumber(msg);

if (choice == 1)

{

obj.ManualFill();

break;

}

else if (choice == 2)

{

obj.RandomFill();

break;

}

else

Console.WriteLine(errorMsg);

} while (true);

Console.WriteLine();

}

public void MakeAndShowChanges(ArrayBase obj)

{

if (obj is Univariate)

{

Univariate univariate = (Univariate)obj;

univariate.ShowArrayObj();

univariate.AddElements();

}

else if (obj is Bivariate)

{

Bivariate bivariate = (Bivariate)obj;

bivariate.ShowMatrix();

bivariate.DeleteEvenRows();

bivariate.ShowMatrix();

}

else if (obj is Ragged)

{

Ragged ragged = (Ragged)obj;

ragged.ShowRaggedObj();

ragged.AddRowToEnd();

}

}

}

}

**Файл – Program.cs**

using Laba\_5;

Menu menu = new Menu();