**5 МАССИВЫ. КЛАСС ARRAY**

Задание 1. В массиве вещественных чисел найти минимальный элемент среди отрицательных элементов.

Листинг программы:

namespace task1

{

using System;

class Program

{

static void Main()

{

Console.WriteLine("Введите колличество элементов в массиве: ");

int arr\_len = Convert.ToInt32(Console.ReadLine());

double[] numbers = new double[arr\_len];

for (int i = 0; i < arr\_len; i++)

{

Console.WriteLine($"Введите элемент массива N{i + 1}: ");

numbers[i] = Convert.ToDouble(Console.ReadLine());

}

double minNegative = double.MaxValue;

foreach (double number in numbers)

{

if (number < 0 && number < minNegative)

{

minNegative = number;

}

}

if (minNegative < 0)

{

Console.WriteLine("Минимальный отрицательный элемент: " + minNegative);

}

else

{

Console.WriteLine("В массиве нет отрицательных элементов.");

}

}

}

}  
Таблица 5.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Кол-во элементов: 5  N1 = -10  N2 = 2  N3 = -4  N4 = -3  N5 = 1 | Минимальный отрицательный элемент: -10 |

Анализ результатов:



Рисунок 5.1 – Результат работы программы

Задание 2 Во всех задачах сформировать и вывести на экран

одномерный массив согласно варианту. Произвести его сортировку и

бинарный поиск числа k (k - вводится с клавиатуры). Напишите программу

для решения задачи: Пусть даны натуральные числа п, а 1 , ..., а п . Определите количество членов а к последовательности а 1 , ..., а п кратных 3 и не кратных 5.

Листинг программы:

namespace task2

{

using System;

class Program

{

static void Main()

{

Console.WriteLine("Введите колличество(n) элементов в последовательности: ");

int n = Convert.ToInt32(Console.ReadLine());

int[] numbers = new int[n];

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

{

Console.WriteLine($"Введите элемент N{i+1} в последовательности: ");

numbers[i] = Convert.ToInt32(Console.ReadLine());

}

Array.Sort(numbers);

Console.Write("Введите число k для бинарного поиска: ");

int k = int.Parse(Console.ReadLine());

int index = Array.BinarySearch(numbers, k);

if (index >= 0)

{

Console.WriteLine($"Число {k} найдено в массиве на позиции {index}");

}

else

{

Console.WriteLine($"Число {k} не найдено в массиве");

}

int count = 0;

foreach (int number in numbers)

{

if (number % 3 == 0 && number % 5 != 0)

{

count++;

}

}

Console.WriteLine($"Количество членов последовательности кратных 3 и не кратных 5: {count}");

}

}

}

Таблица 5.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Кол-во элементов: 5  N1 = -10  N2 = 2  N3 = -4  K = 1 | Число 1 найдено в массиве на позиции 0  Количество членов последовательности кратных 3 и не кратных 5: 1 |

Анализ результатов:



Рисунок 5.2 – Результат работы программы

Задание 3. В каждой задаче задается квадратная целочисленная матрица NxN

(значение N вводится с клавиатуры, N&lt;10). Программа должна заполнять

матрицу случайными числами из диапазона [a, b] (a, b вводятся с

клавиатуры) и осуществлять вывод на экран исходной матрицы. Затем

необходимо произвести необходимые действия и напечатать результаты.

Вычислить среднее арифметическое отрицательных чисел.

Вычислить сумму элементов каждой строки.

Листинг программы:

namespace task3

{

using System;

class Program

{

static void Main()

{

Console.Write("Введите размер N квадратной матрицы (N < 10): ");

int N = int.Parse(Console.ReadLine());

int[,] matrix = new int[N, N];

Console.WriteLine("Введите диапазон случайных чисел [a, b] (a < b): ");

Console.Write("a: ");

int a = Convert.ToInt32(Console.ReadLine());

Console.Write("b: ");

int b = int.Parse(Console.ReadLine());

Random random = new Random();

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

{

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

{

matrix[i, j] = random.Next(a, b);

}

}

Console.WriteLine("Исходная матрица:");

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

{

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

{

Console.Write(matrix[i, j] + " ");

}

Console.WriteLine();

}

int countNegative = 0;

int sumNegative = 0;

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

{

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

{

if (matrix[i, j] < 0)

{

countNegative++;

sumNegative += matrix[i, j];

}

}

}

if (countNegative == 0)

{

Console.WriteLine("В массиве отсутствуют отрицательные элементы!");

}

else

{

double averageNegative = sumNegative / countNegative;

Console.WriteLine($"Среднее арифметическое отрицательных чисел: {averageNegative}"); }

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

{

int sumRow = 0;

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

{

sumRow += matrix[i, j];

}

Console.WriteLine($"Сумма элементов в строке {i + 1}: {sumRow}");

}

}

}

}

Таблица 5.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| N = 4  A = -10  B = 2 | Исходная матрица:  0 -6 -9 -3  1 -2 -5 -7  0 -2 -10 -7  -6 -1 -2 -3  Среднее арифметическое отрицательных чисел: -4  Сумма элементов в строке 1: -18  Сумма элементов в строке 2: -13  Сумма элементов в строке 3: -19  Сумма элементов в строке 4: -12 |

Анализ результатов:

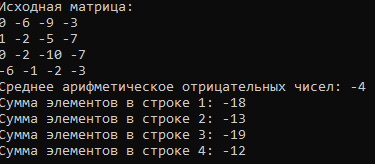


Рисунок 5.3 – Результат работы программы

Задание 4. Решить задачу на проверку условий после выполнения

расчетов: В поезде 20 вагонов, в каждом по 36 мест. Информация о

проданных на поезд билетах хранится в двумерном массиве, номера строк

которого соответствуют номерам вагонов, а номера столбцов – номерам

мест. Если билет на то или иное место продан, то соответствующий элемент

массива имеет значение 1, в противном случае – 0. Определить имеются ли в

поезде свободные места.

Листинг программы:

namespace task4

{

using System;

class Program

{

static void Main()

{

int[,] tickets = new int[20, 36];

Random random = new Random();

for (int i = 0; i < 20; i++)

{

for (int j = 0; j < 36; j++)

{

tickets[i, j] = random.Next(0, 2);

}

}

bool hasFreeSeats = false;

for (int i = 0; i < 20; i++)

{

for (int j = 0; j < 36; j++)

{

if (tickets[i, j] == 0)

{

hasFreeSeats = true;

break;

}

}

if (hasFreeSeats)

{

break;

}

}

if (hasFreeSeats)

{

Console.WriteLine("В поезде имеются свободные места.");

}

else

{

Console.WriteLine("В поезде нет свободных мест.");

}

}

}

}

Таблица 5.4 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| random | В поезде имеются свободные места. |

Анализ результатов:



Рисунок 5.4 – Результат работы программы

Задание 5. Разработать рекурсивный алгоритм и программу решения задачи, в которой вычислить: f(n)=(n-2)!/n!. Исходные данные вводятся с клавиатуры.

Листинг программы:

namespace task5

{

using System;

class Program

{

static double Factorial(int n)

{

if (n == 0)

{

return 1;

}

else

{

return n \* Factorial(n - 1);

}

}

static double CalculateF(int n)

{

if (n <= 2)

{

return 0;

}

else

{

return Factorial(n - 2) / Factorial(n);

}

}

static void Main()

{

Console.WriteLine("Введите значение n:");

int n = int.Parse(Console.ReadLine());

double result = CalculateF(n);

Console.WriteLine($"Значение f({n}) = {result}");

}

}

}

Таблица 5.5 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| n:3 | f(3) = 0,16666666666666666 |

Анализ результатов:



Рисунок 5.5 – Результат работы программы

Задание 6. Заполнить двумерный массив размером 7х7

числами 1, 2, …, 49, расположенными в нем по спирали

Листинг программы:

namespace dopZadanie

{

using System;

class Program

{

static void Main()

{

int n = 7;

int[,] array = new int[n, n];

int num = 1;

int row = 0;

int col = 0;

int maxRow = n - 1;

int maxCol = n - 1;

while (num <= n \* n)

{

for (int i = col; i <= maxCol; i++)

{

array[row, i] = num++;

}

for (int i = row + 1; i <= maxRow; i++)

{

array[i, maxCol] = num++;

}

for (int i = maxCol - 1; i >= col; i--)

{

array[maxRow, i] = num++;

}

for (int i = maxRow - 1; i > row; i--)

{

array[i, col] = num++;

}

row++;

col++;

maxRow--;

maxCol--;

}

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

{

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

{

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

}

Console.WriteLine();

}

}

}

}

Таблица 5.6 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | 1 2 3 4 5 6 7  24 25 26 27 28 29 8  23 40 41 42 43 30 9  22 39 48 49 44 31 10  21 38 47 46 45 32 11  20 37 36 35 34 33 12  19 18 17 16 15 14 13 |

Анализ результатов:

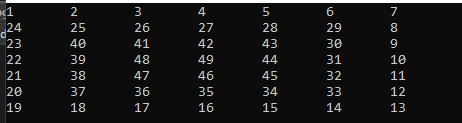


Рисунок 5.6 – Результат работы программы