**5 МАССИВЫ. КЛАСС ARRAY**

Задание №1. Найти количество отрицательных элементов массива А.

Листинг программы:

static int massParser(double[] array)

{

return array.Where(x => x < 0).Count();

}

static void Main(String[] args)

{

Random random = new Random();

double[] array = new double[20];

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

{

array[i] = random.Next(-10, 10);

Console.WriteLine($"{i} : {array[i]}");

}

int numberOfNegative = massParser(array);

Console.WriteLine($"Negativ - {numberOfNegative}");

}

Таблица 5.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Array | -4, 2, -8, -1, 8, -9, 1, -4, 9, -3, -8 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 5.1.

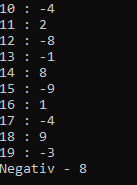


Рисунок 5.1 – Результат работы программы

Источник: собственная разработка

Задание №2 Cформировать и вывести на экран одномерный массив согласно варианту. Произвести его сортировку и бинарный поиск числа k (k - вводится с клавиатуры).

Пусть даны целые числа а1, ..., а99. Получите новую последовательность, выбросив из исходной максимальный и минимальный члены.

Листинг программы:

public static object BinarySearchDisplay(int[] arr, int key)

{

int minNum = 0;

int maxNum = arr.Length - 1;

while (minNum <= maxNum)

{

int mid = (minNum + maxNum) / 2;

if (key == arr[mid]) {

return ++mid;

}

else if (key < arr[mid]) {

maxNum = mid - 1;

}

else

{

minNum = mid + 1;

}

}

return "None";

}

static void Main(String[] args)

{

int firstNum = 0, secondNum = 99;

int[] array = new int[100];

for (firstNum = 1; firstNum < secondNum; firstNum++)

{

array[firstNum] = firstNum;

Console.WriteLine(array[firstNum]);

}

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

Console.WriteLine(BinarySearchDisplay(array, k)); }

Таблица 5.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 97 | 98 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 5.2.

C:\Users\Leo\Downloads\photo1652859535.jpeg

Рисунок 5.2 – Результат работы программы

Источник: собственная разработка

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 элементов, больших заданного G. Вычислить количество чётных элементов k-той строки (значение k вводится с клавиатуры).

Листинг программы:

Console.Write("Enter n: ");

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

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

int a, b;

Console.Write("Enter a: ");

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

Console.Write("Enter b: ");

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

}

Console.WriteLine();

}

Console.WriteLine($"SR = {Arithmetic(matrix, 5)}");

Console.Write("Enter k: ");

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

Console.WriteLine(matrixParser(matrix, k));

static int matrixParser(int[,] matrix, int k)

{

int sum = 0;

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

{

sum += matrix[i, k];

}

return sum;

}

static double Arithmetic(int[,] matrix, int G)

{

int sr = 0;

int indexer = 0;

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

{

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

{

if (matrix[i, j] > G)

{

sr += matrix[i, j];

indexer++;

}

}

}

return sr / indexer;

}

Таблица 5.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 9, 1, 10, 8 | int[,], 7, 58 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 5.3.

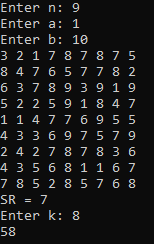


Рисунок 5.3 – Результат работы программы

Источник: собственная разработка

Задание №4. Решить задачу на проверку условий после выполнения расчетов. Дан двумерный массив. Определить в какой строке массива сумма элементов больше: в первой или в предпоследней.

Листинг программы:

Console.Write("Enter n: ");

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

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

int a, b;

Console.Write("Enter a: ");

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

Console.Write("Enter b: ");

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

Random random = new Random();

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

{

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

{

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

Console.Write($"{array[i, j]} ");

}

Console.WriteLine();

}

if (arrayParser(array)) { Console.WriteLine("First"); }

else { Console.WriteLine("Penultimate"); }

static bool arrayParser(int[,] array)

{

int firstSum = 0, penultimateSum = 0;

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

{

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

{

if (i == 0) { firstSum += array[i, j]; }

if (i == array.GetLength(1) - 1) { penultimateSum += array[i, j]; }

}

}

return firstSum > penultimateSum;

}

Таблица 5.4 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 10, 1, 10 | int[,], Penultimate |

Источник: собственная разработка

Анализ результатов представлен на рисунке 5.4.

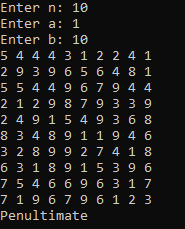


Рисунок 5.4 – Результат работы программы

Источник: собственная разработка

Задание №5. Разработать рекурсивный алгоритм и программу решения задачи, в которой вычислить: f(n)=(n+2)!/(n+4)!. Исходные данные вводятся с клавиатуры.

Листинг программы:

Console.WriteLine("Enter n: ");

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

Console.WriteLine(Formula(n));

static double Formula(int n)

{

return Factorial(n + 2) / Factorial(n + 1);

}

static int Factorial(int f)

{

if (f == 0)

return 1;

else

return f \* Factorial(f - 1);

}

Таблица 5.5 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 7 | 9 |

Источник: собственная разработка

Анализ результатов представлен на рисунке 5.5.



Рисунок 5.5 – Результат работы программы

Источник: собственная разработка

Задание №6. Заполнить двумерный массив размером 7х7 числами 1, 2, …, 49, расположенными в нем по спирали.

Листинг программы:

const int n = 7;

const int m = 7;

int[,] matrix = new int[n, m];

int row = 0;

int col = 0;

int dx = 1;

int dy = 0;

int dirChanges = 0;

int visits = m;

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

{

matrix[row, col] = i + 1;

if (--visits == 0)

{

visits = m \* (dirChanges % 2) + n \* ((dirChanges + 1) % 2) - (dirChanges / 2 - 1) - 2;

int temp = dx;

dx = -dy;

dy = temp;

dirChanges++;

}

col += dx;

row += dy;

}

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

{

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

{

Console.Write($"{matrix[i, j],3}");

}

Console.WriteLine();

}

Таблица 5.6 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 7, 7 | int[,] |

Источник: собственная разработка

Анализ результатов представлен на рисунке 5.6.

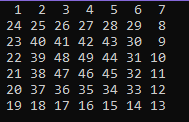


Рисунок 5.6 – Результат работы программы

Источник: собственная разработка