**Лабораторная работа № 4**

**Задание 1**. Создайте проект, в котором опишите класс для решения задачи Вашего варианта. Каждый разрабатываемый класс должен содержать следующие элементы: скрытые и открытые поля, конструкторы с параметрами и без параметров, методы, свойства, индексаторы.

Класс должен реализовывать следующие операции над массивами:

* задание произвольной размерности массива при создании объекта;
* доступ к элементу по индексам с контролем выхода за пределы массива;
* вывод на экран элемента массива по заданному индексу и всего массива.

При возникновении ошибок должны выбрасываться исключения.

В программе должна выполняться проверка всех разработанных

элементов класса.

9. Описать класс для работы с двумерным числовым массивом. Обеспечить возможность проверки, является ли элемент массива палиндромом. Палиндром принимает одно и то же значение при чтении его как справа налево, так и слева направо.

using System;

class TwoDimensionalArray

{

private int[,] array;

public TwoDimensionalArray(int[,] inputArray)

{

array = inputArray;

}

// Метод для проверки, является ли элемент массива палиндромом

public bool IsElementPalindrome(int row, int col)

{

int value = array[row, col];

string valueAsString = value.ToString();

// Проверяем, что значение одинаково при чтении справа налево и слева направо

for (int i = 0; i < valueAsString.Length / 2; i++)

{

if (valueAsString[i] != valueAsString[valueAsString.Length - 1 - i])

{

return false;

}

}

return true;

}

// Метод для вывода массива

public void PrintArray()

{

int rows = array.GetLength(0);

int cols = array.GetLength(1);

Console.WriteLine("Двумерный массив:");

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

{

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

{

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

}

Console.WriteLine();

}

}

}

class Program

{

static void Main()

{

// Пример создания двумерного массива

int[,] myArray = {

{ 121, 234, 345 },

{ 456, 787, 123 },

{ 321, 111, 222 }

};

TwoDimensionalArray arrayHandler = new TwoDimensionalArray(myArray);

// Выводим массив

arrayHandler.PrintArray();

// Запрашиваем строку и столбец для проверки

Console.Write("Введите номер строки: ");

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

Console.Write("Введите номер столбца: ");

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

// Проверяем элемент на палиндром

bool isPalindrome = arrayHandler.IsElementPalindrome(rowToCheck, colToCheck);

if (isPalindrome)

{

Console.WriteLine($"Элемент в строке {rowToCheck}, столбце {colToCheck} является палиндромом.");

}

else

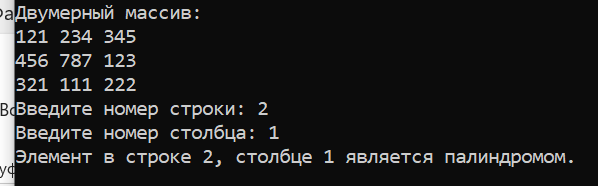
{

Console.WriteLine($"Элемент в строке {rowToCheck}, столбце {colToCheck} не является палиндромом.");

}

}

}



Контрольные вопросы:

1. Что понимается под массивом?

Под массивом понимается структура данных, которая представляет собой упорядоченный набор элементов одного типа, расположенных в памяти компьютера последовательно. Каждый элемент массива имеет свой индекс, который позволяет обращаться к нему.

2. Каковы возможные способы описания массивов (одномерных и многомерных)?

В C# возможны следующие способы описания массивов:

* + Одномерные массивы: объявляются с помощью квадратных скобок [], например int[] numbers = new int[5];.
  + Многомерные массивы: могут быть двумерными, трехмерными и т.д. Объявляются с помощью запятых между квадратными скобками, например int[,] matrix = new int[3, 3];.

3. В каких случаях целесообразно описывать двумерный массив с помощью одномерных?

Целесообразно описывать двумерный массив с помощью одномерных в случаях, когда удобнее работать с данными в одномерной структуре. Например, если требуется представить матрицу в виде одномерного массива для оптимизации работы с памятью или для удобства обработки данных. При этом индексация элементов в одномерном массиве может быть более гибкой и удобной для определенных операций.

4. Какие типы допустимы для описания индексов массивов?

Для описания индексов массивов в C# допустимы следующие типы:

* + - Целочисленные типы: int, uint, long, ulong, short, ushort, byte, sbyte, char.
    - Перечисления (enum), основанные на целочисленных типах.
    - System.Index и System.Range (начиная с C# 8.0).

5. Какие типы могут использоваться в качестве базовых для описания

массивов?

В качестве базовых типов для описания массивов в C# могут использоваться любые типы данных, включая примитивные типы (например, int, double, char), пользовательские классы, структуры, перечисления и другие массивы.

6. Как осуществляется ввод и вывод массивов?

Ввод и вывод массивов в C# можно осуществить с помощью циклов, методов класса Console или других специализированных методов. Например, для ввода массива можно использовать цикл for или foreach, считывая значения с клавиатуры и сохраняя их в элементы массива. Для вывода массива можно также использовать циклы или методы вывода Console.WriteLine.

7. Для чего предназначен цикл foreach?

Цикл foreach в C# предназначен для перебора элементов коллекции или массива без явного использования индексов. Он удобен для случаев, когда не требуется знать текущий индекс элемента, а нужно просто последовательно обработать все элементы коллекции. foreach автоматически перебирает все элементы коллекции от начала до конца.

8. Можно ли использовать цикл foreach для ввода элементов массива?

Нет, цикл foreach не предназначен для ввода элементов массива. Цикл foreach используется для перебора элементов коллекции или массива, но не позволяет изменять значения элементов. Для ввода элементов массива обычно используются циклы for или while, где можно явно указать индексы элементов и их значения.

9. Как определяется базовый тип индексатора?

Базовый тип индексатора определяется возвращаемым типом значения элемента массива или коллекции, к которому происходит доступ с помощью индексатора.

10.Что записывается в качестве имени индексатора?

Имя индексатора записывается как this, за которым в квадратных скобках указываются параметры индексации. Например, public int this[int index].

11.Что содержит список параметров индексатора?

Список параметров индексатора содержит параметры, по которым происходит индексация элементов. В случае одномерного массива параметром может быть один целочисленный индекс, а для многомерного массива список параметров будет содержать несколько индексов для доступа к элементу по соответствующим координатам.