Дисциплина «Программирование корпоративных систем»
Рабочая тетрадь 2.2
Массивы

Теоретический материал

Массив представляет набор однотипных данных. Объявление массива похоже на объявление переменной за тем исключением, что после указания типа ставятся квадратные скобки:

тип_переменной[] название_массива;

Определим массив целых чисел:

int[] numbers;

После определения переменной массива можно присвоить ей определенное значение:

int[] nums = new int[4];

Также мы сразу можем указать значения для этих элементов:

int[] nums2 = new int[4] { 1, 2, 3, 5 };

int[] nums3 = new int[] { 1, 2, 3, 5 };

int[] nums4 = new[] { 1, 2, 3, 5 };

int[] nums5 = { 1, 2, 3, 5 };

Все перечисленные выше способы будут равноценны.

Начиная с версии C# 12 для определения массивов можно использовать выражения коллекций, которые предполагают заключение элементов массива в квадратные скобки:

```
int[] nums1 = [ 1, 2, 3, 5 ];

int[] nums2 = [];   // пустой массив
```

Для обращения к элементам массива используются индексы. Индекс представляет номер элемента в массиве, при этом нумерация начинается с нуля, поэтому индекс первого элемента будет равен 0, индекс четвертого элемента - 3.

Используя индексы, можно, как получить элементы массива:

```
int[] numbers = { 1, 2, 3, 5 };

Console.WriteLine(numbers[3]);

//получение эл-та массива 5
```

Так и изменить элемент массива по индексу:

```
numbers[1] = 505;

Console.WriteLine(numbers[1]);  // 505
```

Каждый массив имеет свойство Length, которое хранит длину массива. Например, получим длину массива numbers:

```
int[] numbers = { 1, 2, 3, 5 };

Console.WriteLine(numbers.Length);  // 4
```

Для перебора массивов можно использовать различные типы циклов. Например, цикл **foreach**:

```
int[] numbers = { 1, 2, 3, 4, 5 };

foreach (int i in numbers)
```

```
{
        Console.WriteLine(i);
}
```

Аналогично подобные действия можно сделать и с помощью цикла **for**:

```
int[] numbers = { 1, 2, 3, 4, 5 };
for (int i = 0; i < numbers.Length; i++)
{
        Console.WriteLine(numbers[i]);
}
```

Также можно использовать и другие виды циклов, например, **while**:

```
int[] numbers = { 1, 2, 3, 4, 5 };
int i = 0;
while(i < numbers.Length)
{
        Console.WriteLine(numbers[i]);
        i++;
}
```

Массивы, которые имеют два измерения (ранг равен 2) называют двухмерными. Например, создадим одномерный и двухмерный массивы, которые имеют одинаковые элементы:

```
int[] nums1 = new int[] { 0, 1, 2, 3, 4, 5 };

int[,] nums2 = { { 0, 1, 2 }, { 3, 4, 5 } };
```

Для генерации случайных чисел в программах, написанных на C#, предназначен класс «Random».

```
//Создание объекта для генерации чисел

Random rnd = new Random(245);

//Получить случайное число (в диапазоне от 0 до 10)

int value = rnd.Next(0, 10);

//Вывод числа в консоль

Console.WriteLine(value);
```

Задание 1

Задача:

Калькулятор матриц

Реализуйте программный продукт средствами языка C# со следующим функционалом:

1) Создание двух матриц размерности n*m (значения n и m вводятся с клавиатуры);
2) Заполнение матриц значениями с клавиатуры (по выбору пользователя, с последующим выводом результата на экран);
3) Заполнение матриц рандомными числами в диапазоне [a; b] (значения a и b вводятся с клавиатуры) (по выбору пользователя, с последующим выводом результата на экран);
4) Сложение матриц (предусмотреть проверку на возможность выполнения операции, с последующим выводом результата на экран);
5) Умножение матриц (предусмотреть проверку на возможность выполнения операции, с последующим выводом результата на экран);
6) Нахождение детерминанта (определителя) матрицы (предусмотреть проверку на возможность выполнения операции, с последующим выводом результата на экран);
7) Нахождение обратной матрицы (предусмотреть проверку на возможность выполнения операции, с последующим выводом результата на экран);
8) Транспонирование матриц (с последующим выводом результата на экран);
9) Нахождение корней системы уравнений, заданных матрицей (с последующим выводом результата на экран).

При тестировании продемонстрировать успешное выполнение всех пунктов (положительный сценарий), а также обработку следующих ситуаций (негативный сценарий):

1) Невозможность сложения матриц по причине несоответствия их размерностей;
2) Невозможность умножения матриц в связи с их несовместимостью;
3) Невозможность нахождения детерминанта у не квадратных матриц (n!=m);
4) Невозможность нахождения обратной матрицы в случае, если детерминант равен нулю (d=0);
5) Невозможность нахождения корней систему уравнений, если она не имеет решения или не имеет однозначного решения.

Весь функционал должен быть реализован вами, программы, разработанные с использованием сторонних решений (библиотеки, фреймворки и т.д.) реализующих функционал, приниматься не будут.

Решение:

Ответ: