Класс Array (МАССИВЫ)

Класс Array (МАССИВЫ)

Массив (array) — это коллекция переменных одинакового типа, обращение к которым происходит с использованием общего для всех имени. В C# массивы могут быть одномерными или многомерными,

C#-массивы реализованы как объекты. Это позволило получить ряд преимуществ, причем одно из них состоит в том, что неиспользуемые массивы могут автоматически утилизироваться системой сбора мусора.

Класс Array предоставляет методы для создания, изменения, поиска и сортировки массивов, то есть выступает в роли базового класса для всех массивов в среде CLR. Тем не менее, явно наследовать класс Array может только система и компиляторы. Пользователи должны применять конструкции массивов, предоставляемые языком программирования.

Класс Array (МАССИВЫ)

Элементом называется значение, содержащееся в объекте Array.

Длина объекта Array равна общему числу элементов, которые могут в нем содержаться.

Ранг объекта Array равен размерности объекта Array. Нижняя граница измерения объекта Array является начальным индексом этого измерения объекта Array; в многомерном массиве Array у каждого измерения могут быть свои нижние границы.

Некоторые свойства класса Array

Length - Возвращает 32-разрядное целое число, представляющее общее число элементов во всех измерениях объекта Array.

LongLength — Возвращает 64-разрядное целое число, представляющее общее число элементов во всех измерениях объекта Array.

Rank - Возвращает ранг (размерность) объекта Array.

Некоторые методы класса Array

public static void Resize(**ref** T[] array, int newSize)

Параметры:

T - тип элементов массива,

ref T[] array - подлежащий изменению размера одномерный массив, индексация которого начинается с нуля, или пустая ссылка (null) для создания нового массива заданного размера,

newSize - размер нового массива.

Этот метод выделяет новый массив заданного размера, копирует элементы из старого массива в новый, а затем заменяет старый массив новым.

Если массив *array* равен значению null, этот метод создает новый массив указанного размера.

Если значение *newSize* больше значения свойства **Length** старого массива, выделяется новый массив, в который копируются все элементы старого массива. Если значение *newSize* меньше значения свойства **Length** старого массива, выделяется новый массив, в который копируются все элементы старого массива, пока этот массив не окажется заполненным. Оставшиеся элементы старого массива игнорируются. Если значение *newSize* равно значению свойства **Length** старого массива, этот метод не выполняет никаких действий.

Одномерные массивы

Одномерный массив — это список связанных переменных. Для объявления одномерного массива используются следующие различные формы записи:

тип[] имя__массива;

тип[] имя__массива = new тип [размер];

тип[] имя__массива = {список инициализаторов};

тип[] имя__массива = new тип [] {список инициализаторов};

тип[] имя__массива = new тип [размер] {список инициализаторов};

Например:

```
double[] a;
float[] b = new float[3];
byte[] c = { 5, 7, -9, 8, 4 };
int[] d = new int[] { 5, 7, -9, 8, 4, -4 };
 short[] e = new short[6] { 5, 7, -9, 8, 4, -4 };
```

Оператор foreach in

foreach (<тип> <имя пер> ***in*** <группа_данных>***)*** <тело_цикла>***;***

Повторяет группу внедренных операторов для каждого элемента в специально организованной группе данных (массива, объекта). Оператор foreach используется для итерации коллекции с целью получения необходимой информации, однако его не следует использовать для изменения содержимого коллекции во избежание непредвиденных побочных Внедренные операторы продолжают выполняться для каждого элемента массива или коллекции. После завершения итерации всех элементов коллекции управление переходит к следующему оператору после блока foreach.

В любой точке блока foreach можно разорвать цикл с помощью ключевого слова break или перейти к следующей итерации в цикле с помощью ключевого слова continue.

Цикл foreach также может быть разорван при помощи операторов goto, return или throw.

Пример Ex1.cs

Сформировать с помощью датчика случайных чисел целочисленный вектор заданного размера, вывести его на экран. Изменить длину вектора. Найти среднее арифметическое элементов вектора.

Пример Ex2.cs

Сформировать с помощью датчика случайных чисел целочисленный вектор заданного размера, вывести его на экран. Добавить в вектор заданное количество элементов, начиная с указанной позиции.

Пример Ex3.cs

Сформировать с помощью датчика случайных чисел целочисленный вектор заданного размера, вывести его на экран. Удалить из вектора заданное количество элементов, начиная с указанной позиции.

Многомерные массивы

Помимо массивов с одним измерением в C# поддерживаются две основные разновидности многомерных массивов:

- **прямоугольные массивы**

- **ломаные (jagged) (ступенчатые)**.

Прямоугольные массивы

Такой тип массива образуется простым сложением нескольких измерений. При этом все строки и столбцы в данном массиве будут одинаковой длины. ***Объявление*** таких массивов имеет вид:

тип[,] имя__массива;
тип[,] имя__массива = new тип [p_1,p_2];
тип[,] имя__массива = {список инициализаторов};
тип[,] имя__массива = new тип [,] {список инициализаторов};
тип[,] имя__массива = new тип [p_1,p_2] {список инициализаторов};

Например:

```
int [,] a;
int [,] b=new int [2,3];
int [,] c={{5,7},{-9,8,4}};
int [,] d=new int[,] {{5,7},{-9,8,4,-4}};
int [,] e=new int[2,3] {{5,7,-9},{8,4,-4}};
```

Доступ к элементам массива: имя_массива[ном_строки, ном_столб]

Пример Ex.cs

Сформировать с помощью датчика случайных чисел целочисленную матрицу заданного размера, вывести ее на экран. Найти максимальный элемент матрицы.

Ступенчатые (jagged) массивы

Такой массив содержит в качестве внутренних элементов некоторое количество внутренних массивов, каждый из которых может иметь свой внутренний уникальный размер.

Объявление таких массивов имеет вид:

тип[][] имя__массива;

Под каждый из массивов, составляющий ломаный массив память требуется выделять явно.

Например:

```
int [][] a=new int [3][];
a[0]=new int [5];
a[1]=new int [7];
a[2]=new int [2];
```

Доступ к элементам массива: имя_массива [ном_строки][ном_столб].

Пример Ex1.cs

Сформировать с помощью датчика случайных чисел целочисленную матрицу с заданным количеством строк, количество столбцов в каждой строке может быть разным и вводится с экрана. Вывести ее на экран. Найти максимальный элемент матрицы.

Пример Ex2.cs

Сформировать с помощью датчика случайных чисел целочисленную матрицу с заданным количеством строк, количество столбцов в каждой строке может быть разным и вводится с экрана. Вывести ее на экран. Удалить из матрицы все четные элементы.

Пример Ex3.cs

Сформировать с помощью датчика случайных чисел целочисленную матрицу с заданным количеством строк, количество столбцов в каждой строке может быть разным и вводится с экрана. Вывести ее на экран. Продублировать в матрице все четные элементы.