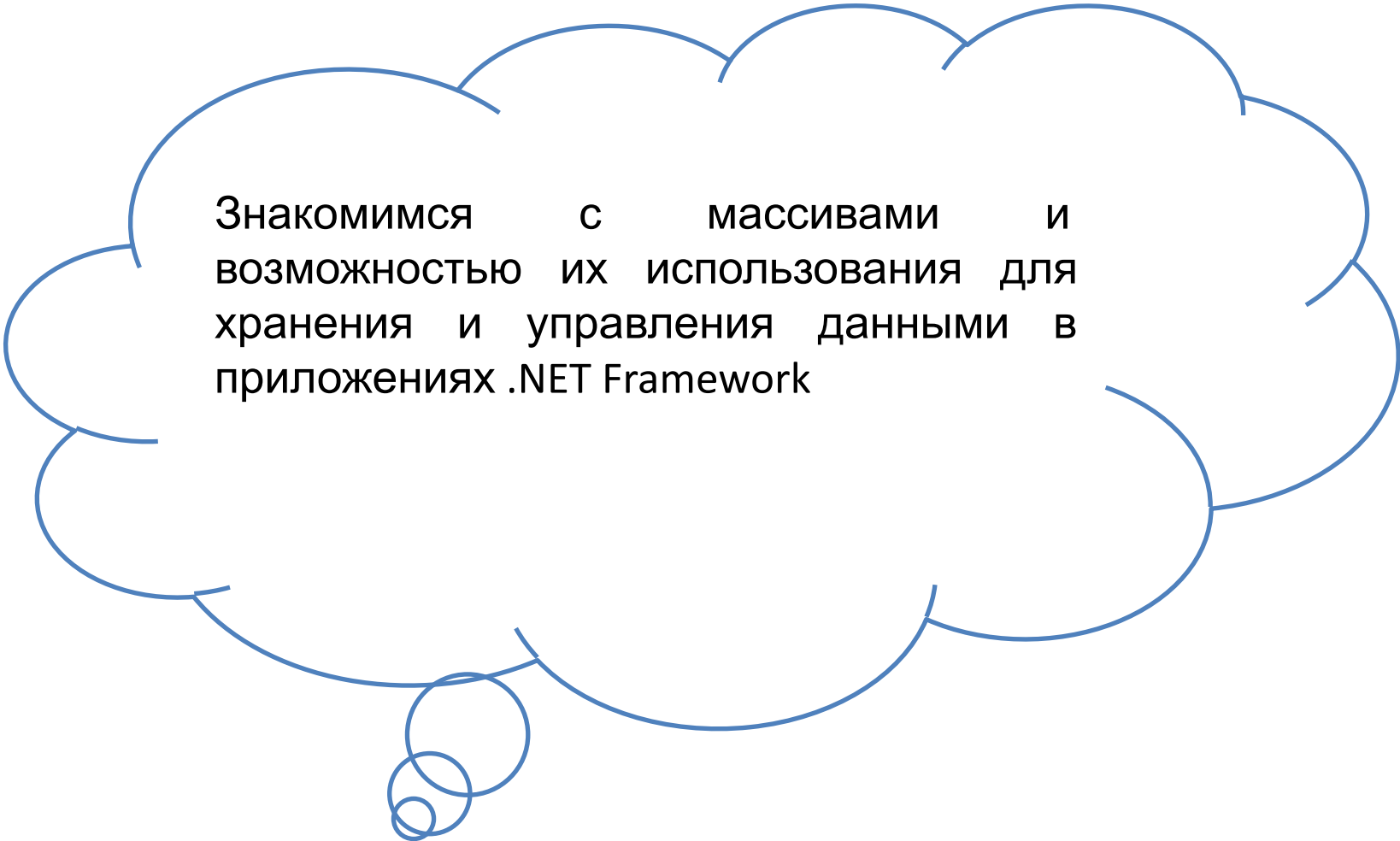


Курс “Языки программирования”

Лекция 05. Массивы в C#.

Создание и использование массивов



Знакомимся с массивами и
возможностью их использования для
хранения и управления данными в
приложениях .NET Framework

Что такое массив?

Массив представляет собой набор объектов, которые сгруппированы вместе и управляются как единое целое

Массивы имеют следующие характеристики:

Каждый элемент в массиве содержит значение

Индексируются с нуля

Длина массива это общее число элементов, которое он может содержать

Нижняя граница массива индекс его первого элемента

Могут быть одномерными, многомерными или непрямоугольные

Ранг массива это число измерений в массиве

Создание и инициализация массивов

Одномерные массивы

```
int[] arrayName;  
.  
.  
.  
int[] list;  
list = new int[20];  
.  
.  
.  
int[] list = new int[20];  
.  
.  
.  
int[] list = new int[5] { 1, 2, 3, 4, 5 };  
int[] list = new int[] { 1, 2, 3, 4, 5 };  
int[] list = new[] { 1, 2, 3, 4, 5 };  
int[] list = { 1, 2, 3, 4, 5 };
```

Если не инициализировать элементы массива, компилятор C# инициализирует их автоматически при его создании с помощью ключевого слова new значениями по умолчанию для его базового типа

Создание и инициализация массивов

Массивы массивов

```
Type [][] jaggedArray = new Type[10][];  
jaggedArray[0] = new Type[5]; // Can specify different sizes  
jaggedArray[1] = new Type[7];  
...  
jaggedArray[9] = new Type[21];
```

```
int[][] jaggedArray = new int[3][],  
    {  
        new int[] {{1, 3}, {5, 7}},  
        new int[] {{0, 2}, {4, 6}, {8, 10}},  
        new int[] {{11, 22}, {99, 88}, {0,  
9}}  
    };
```

Создание и инициализация массивов

Неявно типизированные массивы

```
var mixed = new[] { 1, DateTime.Now, true, false, 1.2 }
```

CTE

```
// int[]  
var a = new[] { 1, 10, 100, 1000 };  
// string[]  
var b = new[] { "hello", null, "world" };
```

```
// single-dimension jagged array  
var c = new[]  
{  
    new[] {1, 2, 3, 4},  
    new[] {5, 6, 7, 8}  
};
```

```
// jagged array of strings  
var d = new[]  
{  
    new[] {"Luca", "Mads", "Luke",  
"Dinesh"},  
    new[] {"Karen", "Suma", "Frances"}  
};
```

Общие свойства и методы, предоставляемые массивами

BinarySearch

```
int[] numbers = { 1, 2, 3, 4, 5 };  
object searchTerm = 3;  
int result = Array.BinarySearch(numbers, searchTerm);
```

Clone

```
int[] numbers = { 1, 2, 3, 4, 5 };  
object numbersClone = numbers.Clone();
```

CopyTo

```
int[] oldNumbers = { 1, 2, 3, 4, 5 };  
int[] newNumbers = new  
int[oldNumbers.Length];  
oldNumbers.CopyTo(newNumbers, 0);
```

GetLength

```
int[] oldNumbers = { 1, 2, 3, 4, 5 };  
int count = oldNumbers.GetLength(0);
```


Общие свойства и методы, предоставляемые массивами

GetValue

```
int[] oldNumbers = { 1, 2, 3, 4, 5 };  
object number = oldNumbers.GetValue(2);  
// returns the value 3
```

SetValue

```
int[] oldNumbers = { 1, 2, 3, 4, 5 };  
oldNumbers.SetValue(5000, 4);  
// Changes the value 5 to 5000
```

Rank

```
int[] oldNumbers = { 1, 2, 3, 4, 5 };  
int rank = oldNumbers.Rank;  
// Returns the value 1
```

Sort

```
int[] oldNumbers = { 5, 2, 1, 3, 4 };  
Array.Sort(oldNumbers);  
// Sorted values: 1 2 3 4 5
```

Length

```
int[] oldNumbers = { 1, 2, 3, 4, 5 };  
int numberCount = oldNumbers.Length;  
// returns the value 5
```

Доступ к данным в массиве

Доступ к конкретным элементам

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

Перебор всех элементов

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

```
int[] oldNumbers = { 1, 2, 3, 4, 5 };  
foreach(int number in oldNumbers)  
{  
    ...  
}
```

Спасибо за внимание