**15 Обобщения**

Задание 1. Используя Visual Studio, создайте проект по шаблону Console Application.

Создайте класс MyList<T>. Реализуйте возможность использования его экземпляра аналогично экземпляру класса MyList<T>. Минимально требуемый интерфейс взаимодействия с экземпляром, должен включать метод добавления элемента, индексатор для получения значения элемента по указанному индексу и свойство только для чтения для получения общего количества элементов.

Листинг программы:

internal class Program

{

static void Main(string[] args)

{

MyList<int> myList = new MyList<int>();

for (int i = 0; i < 10; i++)

{

myList.Add(i);

}

foreach (var item in myList)

{

Console.Write("{0} | ", item);

}

var count = myList.count;

Console.WriteLine("\nКоличество элементов: {0}", count);

Console.WriteLine("\n7-oй элемент: {0}", myList[6]);

Console.ReadLine();

}

}

class MyList<T> : IEnumerable<T>

{

public int count { get { return array.Length; } }

T[] array = new T[0];

public void Add(T item)

{

Array.Resize(ref array, array.Length + 1);

array[array.Length - 1] = item;

}

public IEnumerator<T> GetEnumerator()

{

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

{

yield return array[i];

}

}

IEnumerator IEnumerable.GetEnumerator()

{

return array.GetEnumerator();

}

public T this[int index]

{

get { return array[index]; }

set { array[index] = value; }

}

}

Таблица 1.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |  Количество элементов: 10  7-oй элемент: 6 |

Анализ результатов:

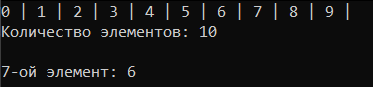


Рисунок 1.1 – Результат работы программы

Задание 2. Используя Visual Studio, создайте проект по шаблону Console Application.

Создайте класс MyDictionary <TKey, TValue>. Реализуйте возможность использования его экземпляра аналогично экземпляру класса Dictionary. Минимально требуемый интерфейс взаимодействия с экземпляром, должен включать метод добавления пар элементов, индексатор для получения значения элемента по указанному индексу и свойство только для чтения для получения общего количества пар элементов.

Листинг программы:

internal class Program

{

static void Main(string[] args)

{

MyDictionary<string, string> myDict = new MyDictionary<string, string>();

myDict.Add("1", "One");

myDict.Add("2", "Two");

myDict.Add("3", "Three");

foreach (var item in myDict)

{

Console.Write("{0} | {1}\n", item.key, item.value);

}

Console.WriteLine("Number of pairs: {0}", myDict.Count);

Console.WriteLine("Choose: ");

var key = Console.ReadLine();

Console.WriteLine("{0}", myDict[key]);

Console.ReadLine();

}

}

class MyDictionary<Tkey, Tvalue> : IEnumerable<KeyValueClass<Tkey, Tvalue>>

{

public int Count { get { return myArray.Length; } }

private KeyValueClass<Tkey, Tvalue>[] myArray = new KeyValueClass<Tkey, Tvalue>[0];

public void Add(Tkey key, Tvalue value)

{

Array.Resize(ref myArray, myArray.Length + 1);

myArray[myArray.Length - 1] = new KeyValueClass<Tkey, Tvalue>() { key = key, value = value };

}

public IEnumerator<KeyValueClass<Tkey, Tvalue>> GetEnumerator()

{

foreach (var item in myArray)

{

yield return item;

}

}

IEnumerator IEnumerable.GetEnumerator()

{

return myArray.GetEnumerator();

}

public Tvalue this[Tkey index]

{

get

{

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

{

if (myArray[i].key.Equals(index))

{

return myArray[i].value;

}

}

throw new DllNotFoundException("Not found");

}

}

}

class KeyValueClass<Tkey, Tvalue>

{

public Tkey key { get; set; }

public Tvalue value { get; set; }

}

Таблица 1.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 3 | 1 | One  2 | Two  3 | Three  Number of pairs: 3  Choose:  Three |

Анализ результатов:

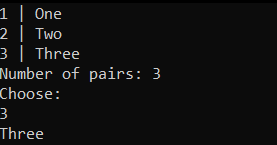


Рисунок 1.2 – Результат работы программы