**ЗАДАНИЕ 15**

Задание 1. Используя Visual Studio, создайте проект по шаблону Console Application. Создайте класс MyList&lt;T&gt;. Реализуйте возможность использования его экземпляра аналогично экземпляру класса List&lt;T&gt;. Минимально требуемый интерфейс взаимодействия с экземпляром, должен включать метод добавления элемента, индексатор для получения значения элемента по указанному индексу и свойство только для чтения до получения общего количества элементов.

Листинг программы:

using System;

using System.Collections.Generic;

class MyList<T>

{

private List<T> items = new List<T>();

public void Add(T item)

{

items.Add(item);

}

public T this[int index]

{

get { return items[index]; }

}

public int Count

{

get { return items.Count; }

}

}

class Program

{

static void Main()

{

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

myNumbers.Add(10);

myNumbers.Add(20);

myNumbers.Add(30);

for (int i = 0; i < myNumbers.Count; i++)

{

Console.WriteLine(myNumbers[i]);

}

}

}

Таблица 15.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| - | 10, 20, 30 |

Анализ результатов:

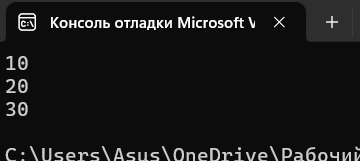


Рисунок 15.1 – Результат работы программы

Задание 2. Используя Visual Studio, создайте проект по шаблону Console Application. Создайте класс MyDictionary &lt;TKey, TValue&gt;. 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:

using System;

using System.Collections.Generic;

class MyDictionary<TKey, TValue>

{

private Dictionary<TKey, TValue> dictionary = new Dictionary<TKey, TValue>();

public void Add(TKey key, TValue value)

{

dictionary.Add(key, value);

}

public TValue this[TKey key]

{

get { return dictionary[key]; }

}

public int Count

{

get { return dictionary.Count; }

}

}

class Program

{

static void Main()

{

MyDictionary<string, int> myNumbers = new MyDictionary<string, int>();

myNumbers.Add("один", 1);

myNumbers.Add("два", 2);

myNumbers.Add("три", 3);

Console.WriteLine("Общее количество пар ключ-значение: " + myNumbers.Count);

Console.WriteLine("Значение для ключа 'два': " + myNumbers["два"]);

}

}

Таблица 15.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 1, 2, 3 | 3, 2 |

Анализ результатов:

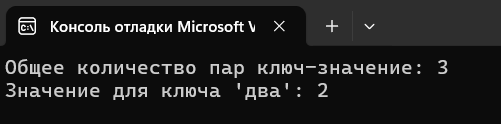


Рисунок 15.2 – Результат работы программы

Задание 3. Используя Visual Studio, создайте проект по шаблону Console Application. Создайте расширяющий метод: public static T[] GetArray&lt;T&gt;(this MyList&lt;T&gt; list) Примените расширяющий метод к экземпляру типа MyList&lt;T&gt;, разработанному в задании 2 для данного урока. Выведите на экран значения элементов массива, который вернул расширяющий метод GetArray().

Листинг программы:

using System;

using System.Collections.Generic;

class MyList<T>

{

private List<T> items = new List<T>();

public void Add(T item)

{

items.Add(item);

}

public T this[int index]

{

get { return items[index]; }

}

public int Count

{

get { return items.Count; }

}

}

static class MyListExtensions

{

public static T[] GetArray<T>(this MyList<T> list)

{

T[] array = new T[list.Count];

for (int i = 0; i < list.Count; i++)

{

array[i] = list[i];

}

return array;

}

}

class Program

{

static void Main()

{

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

myNumbers.Add(10);

myNumbers.Add(20);

myNumbers.Add(30);

int[] array = myNumbers.GetArray();

Console.WriteLine("Значения в массиве:");

foreach (var item in array)

{

Console.WriteLine(item);

}

}

}

Таблица 15.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| 10, 20, 30 | 10, 20, 30 |

Анализ результатов:

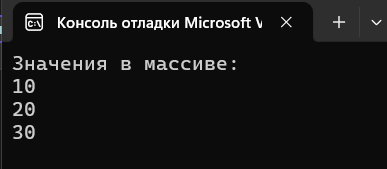


Рисунок 15.3 – Результат работы программы