**15 ОБОБЩЕНИЯ**

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 для получения общего

количества элементов.

Листинг программы:

using System;

public class MyList<T>

{

private T[] \_items;

public MyList()

{

\_items = new T[0];

}

public T this[int index]

{

get

{

if (index < 0 || index >= Count)

{

throw new IndexOutOfRangeException();

}

return \_items[index];

}

}

public int Count

{

get

{

return \_items.Length;

}

}

public void Add(T item)

{

T[] newItems = new T[Count + 1];

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

{

newItems[i] = \_items[i];

}

newItems[Count] = item;

\_items = newItems;

}

}

class Program

{

static void Main(string[] args)

{

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

myList.Add(1);

myList.Add(2);

myList.Add(3);

Console.WriteLine("Количество элементов: " + myList.Count);

Console.WriteLine("Элемент по индексу 1: " + myList[1]);

}

}

Таблица 15.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | 3  2 |

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;

public class MyDictionary<TKey, TValue>

{

private List<TKey> keys;

private List<TValue> values;

public MyDictionary()

{

keys = new List<TKey>();

values = new List<TValue>();

}

public TValue this[TKey key]

{

get

{

int index = keys.IndexOf(key);

if (index == -1)

{

throw new KeyNotFoundException();

}

return values[index];

}

}

public int Count

{

get

{

return keys.Count;

}

}

public void Add(TKey key, TValue value)

{

if (keys.Contains(key))

{

throw new ArgumentException("An element with the same key already exists in the MyDictionary");

}

keys.Add(key);

values.Add(value);

}

}

class Program

{

static void Main(string[] args)

{

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

myDictionary.Add("one", 1);

myDictionary.Add("two", 2);

myDictionary.Add("three", 3);

Console.WriteLine("Количество пар элементов: " + myDictionary.Count);

Console.WriteLine("Значение элемента по ключу 'two': " + myDictionary["two"]);

}

}

Таблица 15.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| two | 3  2 |

Анализ результатов:

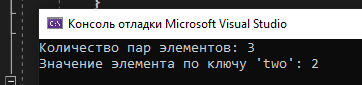


Рисунок 15.2 – Результат работы программы

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;

public class MyList<T>

{

private List<T> \_items = new List<T>();

public T this[int index] => \_items[index];

public int Count => \_items.Count;

public void Add(T item)

{

\_items.Add(item);

}

}

public 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(string[] args)

{

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

myList.Add(1);

myList.Add(2);

myList.Add(3);

int[] array = myList.GetArray();

Console.Write("Элементы массива: ");

foreach (var item in array)

{

Console.Write(item + ", ");

}

}

}

Таблица 15.3 – Входные и выходные данные

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

Анализ результатов:

****

Рисунок 15.3 – Результат работы программы