**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 List<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:

class MyList<T>

{

private T[] myList = null;

public T this[int index]

{

get { return myList[index]; }

set { myList[index] = value; }

}

public MyList()

{

this.myList = new T[1];

}

public MyList(int count)

{

this.myList = new T[count];

}

public void Add(T item)

{

if (myList == null)

{

myList = new T[1];

myList[0] = item;

}

else

{

int length = myList.Length;

Array.Resize(ref myList, length + 1);

myList[length] = item;

}

}

public int Capacity

{

get

{

return myList.Length;

}

}

public int Count

{

get

{

int count = 0;

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

{

if (myList[i].ToString() != null)

{

count++;

}

}

return count;

}

}

public void Info()

{

foreach (var item in myList)

{

Console.WriteLine(item);

}

}

}

class Program

{

static void Main(string[] args)

{

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

Console.WriteLine("Введите емкость: ");

mylist[0] = int.Parse(Console.ReadLine());

Console.WriteLine("Емкость списка: {0}", mylist.Capacity);

Console.WriteLine("Список фактически содержит: {0}", mylist.Count);

mylist.Add(10);

Console.WriteLine("Список после добавления: {0}", mylist.Capacity);

mylist.Info();

mylist.Add(25);

Console.WriteLine("After");

mylist.Info();

Console.ReadLine();

}}

Таблица 15.1 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| емкость:  45 | Емкость списка: 1  Список фактически содержит: 1  Список после добавления: 2  45  10  After  45  10  25 |

Анализ результатов:

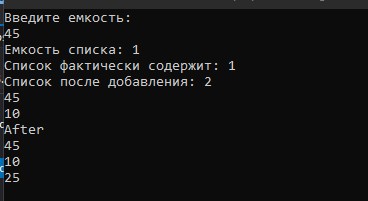


Рисунок 15.1 – Результат работы программы

Источник: собственная разработка

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;

namespace p2

{

class MyDictionary<Tkey, Tval>

{

private int counter = 0;

private Tkey[] keysArray = null;

private Tval[] valsArray = null;

public int Counter

{

get { return this.counter; }

}

public void Add(Tkey key, Tval val)

{

this.counter++;

Array.Resize(ref keysArray, counter);

keysArray[counter - 1] = key;

Array.Resize(ref valsArray, counter);

valsArray[counter - 1] = val;

}

public Tval this[Tkey key]

{

get

{

int ind = 0;

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

{

if (key.Equals(keysArray[i]))

ind = i;

}

return valsArray[ind];

}

}

}

class Program

{

static void Main(string[] args)

{

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

dictionary.Add(0, "Sport");

Console.WriteLine(dictionary[0]);

Console.WriteLine("Количество элементов после добавления = " + dictionary.Counter);

dictionary.Add(1, "Run");

Console.WriteLine(dictionary[1]);

Console.WriteLine("Количество элементов = " + dictionary.Counter);

Console.ReadKey();

}

}

}

Таблица 15.2 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | Sport  Элементов после добавления = 1  Run  Кол-во элементов 2 |

Анализ результатов:

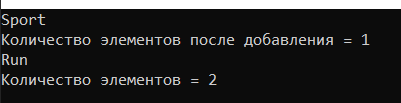


Рисунок 15.2 – Результат работы программы

Источник: собственная разработка

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;

namespace p3

{

public class MyList<T>

{

public static T session;

public List<T> lst = new List<T>();

static int count;

public int Count

{

get

{

return count;

}

set

{

count = lst.Count;

}

}

public void Add(T value)

{

lst.Add(value);

}

public void Remove(T value)

{

lst.Remove(value);

}

public void PrintList()

{

Console.WriteLine("Список:");

foreach (var item in lst)

{

Console.Write(item + " ");

}

Console.WriteLine();

}

public void GetIndex(int index)

{

if (lst.Count >= index)

{

Console.WriteLine("Значение с индеком {0}:{1}", index, lst[index]);

}

else

{

Console.WriteLine("Элемента с индексом {0} не существует!", index);

}

}

}

public static class StaticClass

{

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

{

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

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

{

array[i] = list.lst[i];

}

return array;

}

}

class Program

{

static void Main(string[] args)

{

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

MyList<int>.session = 100;

mylst.Add(9999);

mylst.Add(1);

mylst.Add(123);

mylst.Add(87);

mylst.Add(1);

mylst.Add(321);

mylst.Remove(200);

mylst.PrintList();

mylst.GetIndex(1);

mylst.Count = 0;

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

Console.WriteLine("Массив:");

int[] array = StaticClass.GetArray<int>(mylst);

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

{

Console.Write(array[i] + " ");

}

}

}

}

Таблица 15.3 – Входные и выходные данные

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
|  | Список  9999 1 123 87 1 321  Значение с индексом 1:1  Кол-во элементов: 6  Массив  9999 1 123 87 1 321 |

Анализ результатов:

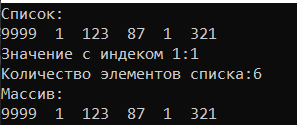


Рисунок 15.3 – Результат работы программы

Источник: собственная разработка