**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:

static void Main(string[] args)

{

MyList<int> myList = new();

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

{

myList.Add(i);

}

foreach (var item in myList)

{

Write($"{item} | ");

}

var count = myList.count;

WriteLine($"\nКоличество элементов: {count}");

WriteLine($"\n3й элемент: {myList[2]}");

ReadLine();

}

Анализ результатов:

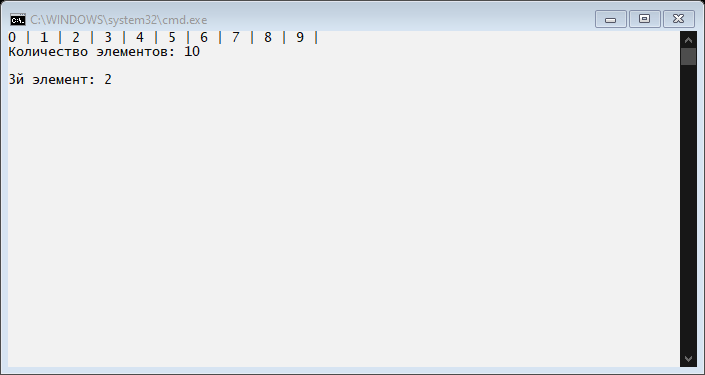


Рисунок 1.1 – Результат работы программы

**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:

static void Main(string[] args)

{

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

for (int i = 0; i < 5; i++)

{myDict.Add(Faker.Address.City(), Faker.Company.Name());}

foreach (var item in myDict)

{Write($"{item.key} | {item.value}\n");}

WriteLine($"Количество пар: {myDict.Count}");

WriteLine("Выберите: ");

var key = ReadLine();

WriteLine($"{myDict[key]}");

ReadLine();

}

Анализ результатов:

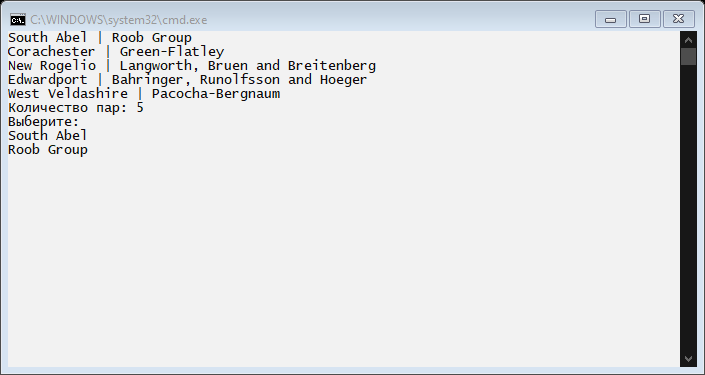


Рисунок 1.2 – Результат работы программы