2. Реализовать метод расширения для поиска кол-во символов в строке.

public static class StringExtension

{

    public static int CharCount(this string str)

    {

        int counter = str.Length + 1;

        return counter;

    }

}

3. Дана коллекция List<T>. Требуется подсчитать, сколько раз каждый элемент встречается в данной коллекции:

a. для целых чисел;

List<int> intList = new List<int> { 21, 46, 46, 55, 17, 21, 55, 55 };

Dictionary<int, int> dictionary = new Dictionary<int, int>();

foreach (int i in intList)

{

if(dictionary.ContainsKey(i))

{

dictionary[i]++;

}

else

{

dictionary[i] = 1;

}

}

foreach (KeyValuePair<int, int> p in dictionary)

{

Debug.Log($”Элемент {p.Key} встречается {p.Value} раз”);

}

b. для обобщенной коллекции;

List<T> list = new List<T>;

Dictionary<T, int> dictionary = new Dictionary<T, int>();

foreach (var l in list)

{

if(dictionary.ContainsKey(l))

{

dictionary[l]++;

}

else

{

Dictionary.Add(l, 1);

}

}

foreach (KeyValuePair<T, int> p in dictionary)

{

Debug.Log($”Элемент {p.Key} встречается {p.Value} раз”);

}

c. используя Linq

List<T> list = new List<T>;

var arrg = list.GroupBy(l => 1)

.Select(g => new {Value = g.Key, Count = g.Count()});

foreach (var el in arrg)

{

Debug.Log($”Элемент {el.Key} встречается {el.Value} раз”);

}

4. В методичке дан фрагмент программы, необходимо:

Dictionary<string, int> dict = new Dictionary<string, int>()

{

{"four",4 },

{"two",2 },

{ "one",1 },

{"three",3 },

};

a.Свернуть обращение к OrderBy с использованием лямбда-выражения =>.

var d = dict.OrderBy(pair => pair.Value);

foreach (var pair in d)

Debug.Log($”{0} – {1}”, pair.Key, pair.Value);

b. Развернуть обращение к OrderBy с использованием делегата

Func<KeyValuePAir<string, int>, int> delegate =

delegate (KeyValuePAir<string, int> pair) {return pair.Value};

var d = dict.OrderBy(deleagte);

foreach (var pair in d)

Debug.Log($”{0} – {1}”, pair.Key, pair.Value);