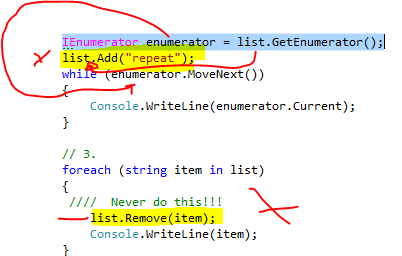


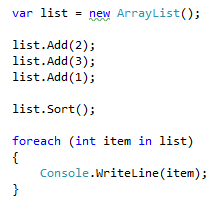
Не в коем случае нельзя удалять и добавлять элементы, после получения

IEnumerator enumerator = list.GetEnumerator();

Если планируется перебор элементов foreach или MoveNext() при помощи полученного enumerator

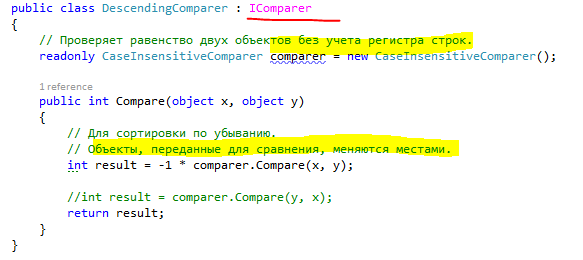


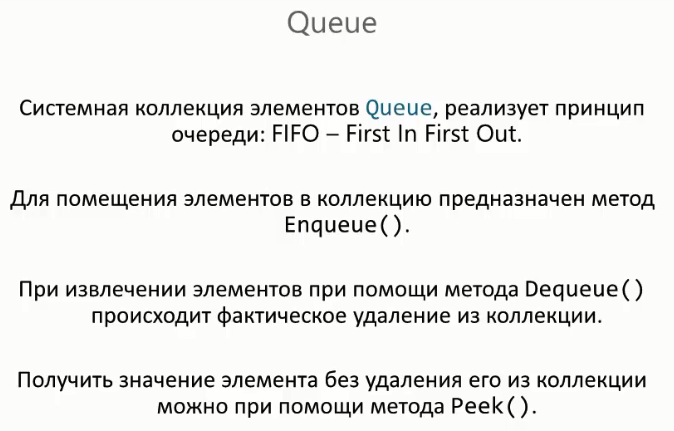
Сортировка



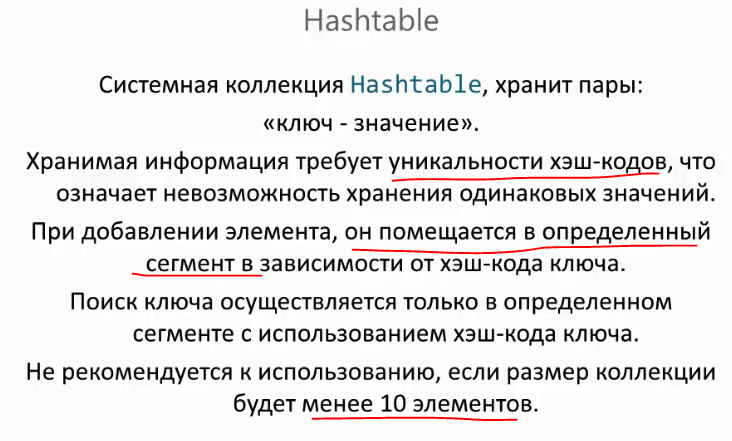
Сортировка по убыванию, создается класс реализовывающий интерфейс IComparer



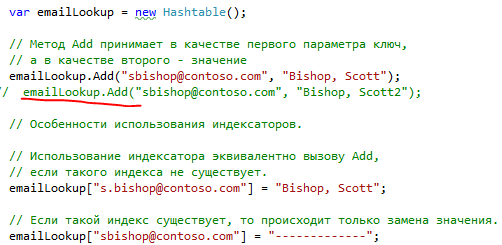








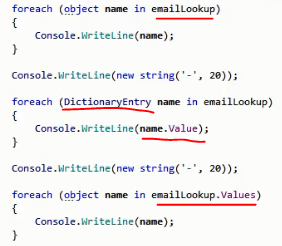
При добавлении существующего ключа вызывается Exception



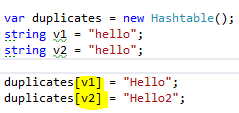
Перебор элементов HashTable

Каждым элементом коллекции является экземпляр

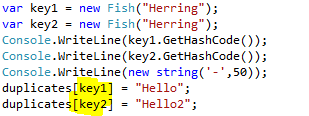




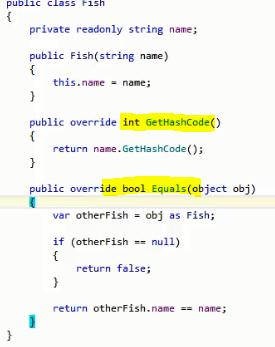
Будет добавлена один элемент в таблицу не смотря на то что переменные разные (происходит Интермирование строки при компиляции (это когда имеются две одинаковые строки) и переменные ссылаются на одну и ту же строку) поэтому элемент будет перезаписан



А в данном случае будет добавлено два элемента не смотря на то что внутри классов объекты одинаковы (НО ХЭШ КОДЫ РАЗНЫЕ)



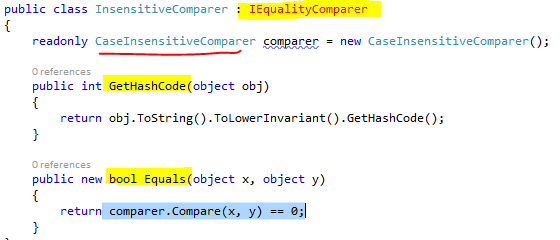
По методам GetHashCode() и Equals()HashTable определяет уникальность ключей (классов)



HashTable при добавлении ключа, берет GetHashCode() ключа и проверяет есть ли в таблице такой же код, если нет то добавляет, а если имеется то проверяет ключи через метод Equals(), т.к. бывают случаи когда два разных объекта имеют одинаковые ХЭШ-КОДЫ но разные значения

Для того чтобы создать коллекция не чувствительную к регистру необходимо реализовать интерфейс IEqualityComparer в классе и после передать его в конструктор HashTable

Класс - readonly CaseInsensitiveComparer comparer = new CaseInsensitiveComparer(); позволяет сравнивать две строки без учета регистра comparer.Compare(x, y) == 0;



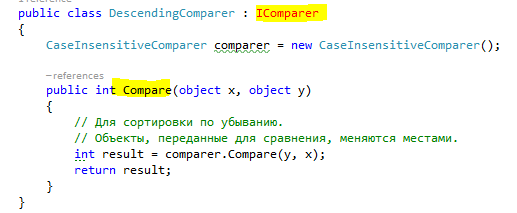


SortedList

требует уникальности ключей, и все элементы храняться по ключу в отсортированном порядке

Для пользоваьельской сортировки необходимо создать класс реализовывающий интерфейс IComparer , и передать созданный класс в конструктор SortedList

var sort = new SortedList(new DescendingComparer());



ListDictionary

– такая же как и HashTable, но использовать ее следует если количество элементов не будет превышать 10 элементов

HybridDictionary

- Рекомендуется к использованию в тех случаях, когда невозможно определить размер коллекции заранее.

OrderedDictionary

Гарантирует что элементы в коллекции будут распологаться в том же порядке в котором их добавляли (HashTable и ListDictionary этотого не грантируют )

Чтобы автоматически сортировалась коллекция

