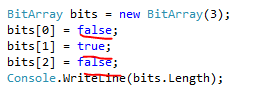
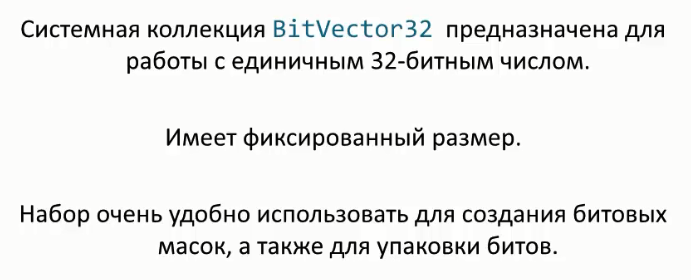


Создаем массив на три бита (0,1,0)

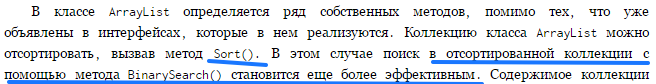






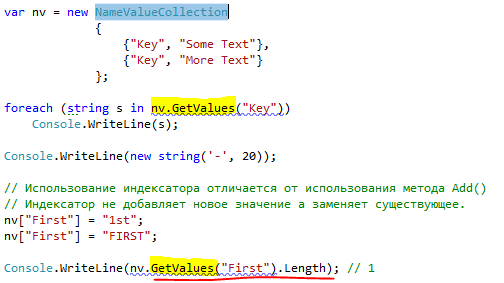


ArrayList тоже что и List<T>



NameValueCollection

Работает по принципу словаря, но позволяет под одним ключом хранить несколько значений

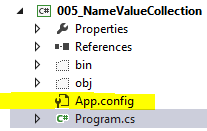


Метод Add() – добавляет

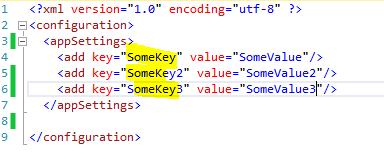
[ ] – индексатор заменяет существующий элемент

App.Config (settings)

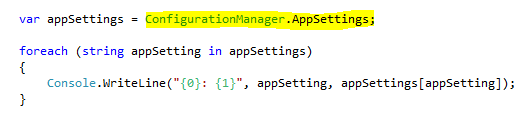
Файл конфигурации, находятся настройки для приложения



Можно добавлять сетингс



using System.Configuration;





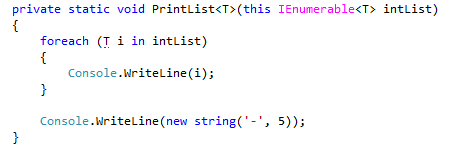
Generic Collection

Класс List<T> является generic версией класса ArrayList. Если вам нужно будет выбирать между этими двумя вариантами, то стоит помнить, что List<T> обычно более эффективен и является типобезопасным (type safe). Однако, если вы используете в качестве типа Т используете значимый тип, то стоит учитывать нагрузку из-за проблем с упаковкой и распаковкой.

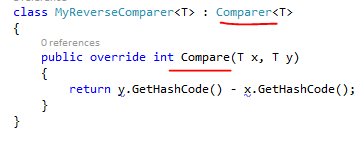
List<T> (на подобие ArrayList)

Сортировки списка

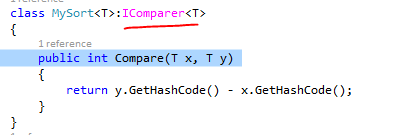
Расширяющий метод



Необходимо создать класс наследующийся от класса Comparer<T> или реализовывающий интерфейс

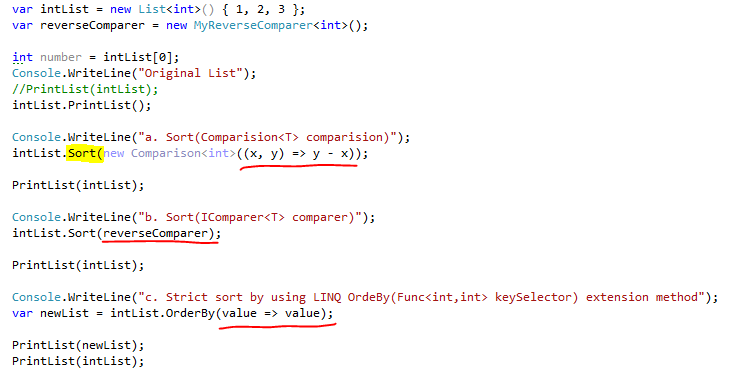


Или

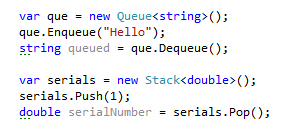


Также можно лямда выражение (принимает класс делегат Sort(Comparision<T> comparision))

А также через расширяющий метод OrderBy который упорядочивает коллекцию

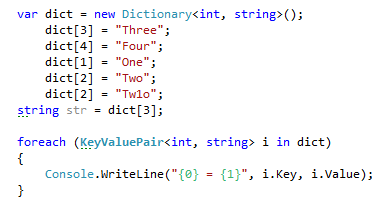


Queue – очередь

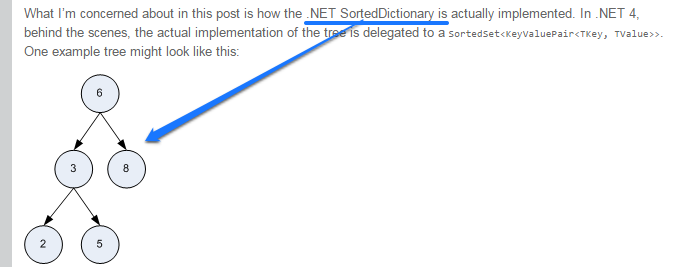


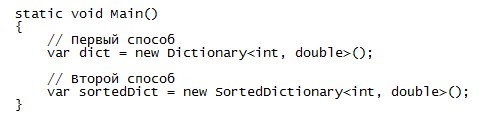
Для очередей есть и потокобезопасные варианты –ConcurrentQueue<T> и ConcurrentStack<T>.

Dictionary тот же HashTable

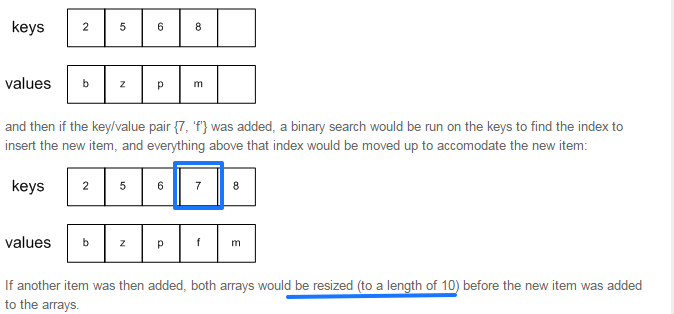


SortedDictionary

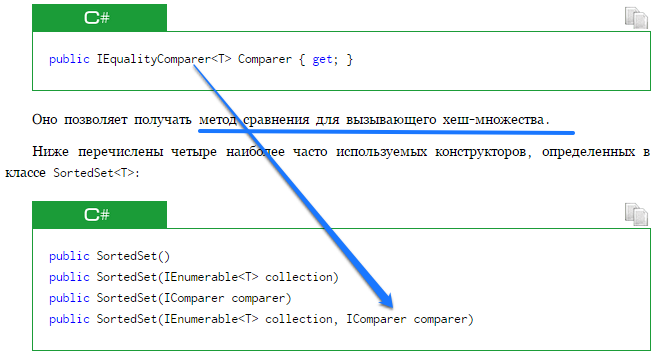


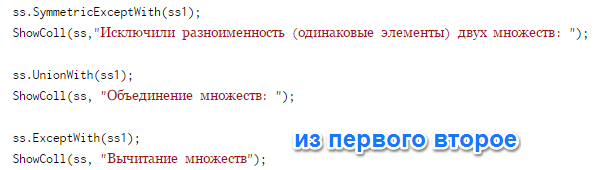


SortedList



**Множества: классы HashSet<T> и SortedSet<T>**





LinkedList

