Что такое коллекции, какие пространства имен нужны для использования в шарпе, задачи которые проще решать с помощью коллекций (без кода)

Необобщенные коллекции, перечислить их (ArrayList, BitArray, Hashtable, Queue, SortedList, Stack)

ArrayList, класс простой коллекции объектов. Реализует интерфейсы IList, ICollection, IEnumerable, тот же массив только без верхней границы

BitArray, класс коллекции, содержащей массив битовых значений. Реализует интерфейсы ICollection, IEnumerable

Hashtable, класс коллекции, представляющей хэш-таблицу и храняющий набор пар "ключ-значение"

Queue, класс очереди объектов, работающей по алгоритму FIFO ("первый вошел -первый вышел"). Реализует интерфейсы ICollection, IEnumerable, пример из реальности очередь в магазине

SortedList, класс коллекции, хранящей наборы пар "ключ-значение", отсортированных по ключу. Реализует интерфейсы ICollection, IDictionary, IEnumerable, ну тут все сказано

Stack, класс стека, каждый последующий эл-нт кладется поверх предыдущего, берутся соответственно в обратном порядке, пример из реальности стопка книг, тарелок, да чего угодно

Обобщенные коллекции, отличие от необобщенных (типизация)

У необобщенных есть аналоги обобщенных:

у ArrayList List<T>: класс, представляющий последовательный список. Реализует интерфейсы IList<T>, ICollection<T>, IEnumerable<T>

BitArray особый случай аналогов не имеет так как хранит не объекты

у Hashtable Dictionary<TKey, TValue>: класс коллекции, хранящей наборы пар "ключ-значение". Реализует интерфейсы ICollection<T>, IEnumerable<T>, IDictionary<TKey, TValue>, товары в магазине

остальные имеют одноименные аналоги

LinkedList надо дать картинку принципа работы рассказать о некоторых преимуществах перед List



бывают задачи когда необходимо создать свою коллекцию, в таких задачах чтобы унифицировать ее надо использовать те интерфейсы что лежат в основе базовых коллекций: рассказать про интерфейсы стоящими за коллекциями IEnumerable<T> IEnumerator ICollection<T> IList<T> readonly IList<T> ICollection<T> IKeyValueCollection (IDictionary<TKey, TValue>)

IEnumerable<T> наследуется от своего необобщенного аналога, который в свою очередь имеет след методы: IEnumerator GetEnumerator() который возвращает нумератор позволяющий проходить по коллекции

IEnumerator: свой-во object Current { get; }, которое указывает на текущий рассматриваемый эл-нт

Методы bool MoveNext(); - перейти к следующему, возвращает true при успешном и false в противном случае

void Reset(); возвращает Current в начало коллекции

ICollection<T>: наследуется от IEnumerable<T> и IEnumerable

int Count { get; } кол-во эл-тов в коллекции на данный момент

bool IsReadOnly { get; } только на чтение или нет?

void Add(T item); добавить эл-т

void Clear(); отчистить коллекцию

bool Contains(T item); содержит ли item?

void CopyTo(T[] array, int arrayIndex); копирует из коллекции в массив с arrayIndex индекса

bool Remove(T item); удаляет эл-т item

IList<T>: наследуется от ICollection<T>, IEnumerable<T>, IEnumerable

T this[int index] { get; set; } тип эл-тов коллекции

int IndexOf(T item); индекс эл-та item

void Insert(int index, T item); вставить в коллекцию на позицию index эл-т item

void RemoveAt(int index); удалить эл-т с позиции index

IDictionary<TKey, TValue>: наследуется от ICollection<KeyValuePair<TKey, TValue>>, IEnumerable<KeyValuePair<TKey, TValue>>, IEnumerable

void Add(TKey key, TValue value); добавить эл-т в словарь

bool ContainsKey(TKey key); содержит ли ключ key?

bool Remove(TKey key); удалить эл-т, ключ которого совпадает с key

bool TryGetValue(TKey key, out TValue value); найти эл-т с ключом key

ICollection<TKey> Keys { get; } коллекция ключей

ICollection<TValue> Values { get; } коллекция значений

TValue this[TKey key] { get; set; } возвращает или изменяет эл-т с соответствующим ключом