**The Collection Interface**

Specifies a subset of methods in the List interface, specifically excluding:

* add(int, E)
* get(int)
* remove(int)
* set(int, E)

but including:

* add(E)
* remove(Object)
* Diagram

  Description automatically generatedthe iterator method to return an iterator

int index is for the List only

Common Features of Collections

Collections

* grow as needed
* hold references to objects
* have at least 2 constructors
  + one to create an empty collection and
  + one to make a copy of another collection
    - you should use iterator to traverse through 2nd collection and copy everything

Table

Description automatically generated

In a general Collection, the order of elements is not specified

For collections implementing the List interface, the order of elements is determined by the index

In a general Collection, the position where an object is inserted is not specified

In ArrayList and LinkedList, add(E) always inserts at the end and always returns true

AbstractCollection, AbstractList, and AbstractSequentialList …

The Java API includes several “helper” abstract classes to help build implementations of their corresponding interfaces

By providing implementations for interface methods not used, the helper classes require the programmer to extend the AbstractCollection class and implement only the desired methods

Implementing a Subclass of Collection<E>

Extend AbstractCollection<E>, which implements most operations

You need to implement only:

* add(E)
* size()
* iterator()
* an inner class that implements Iterator<E>

There is a remove method in Collection interface and it is implemented in AbstractCollection<E> so you don’t need to implement it.

How remove method is implemented in AbstractCollection?

You have to implement inner class Iterator and it has remove method. Iterator’s remove method is used to remove an element.

Implementing a Subclass of List<E>

Extend AbstractList<E>

Iterator class is implemented in AbstractList too. How it is implemented because we don’t know if it is linked list or array list or etc.?

* remove, hasNext, next implement edilmeli
* remove dedin mi neyi remove edecek? remove metodu için integer lazım. Sonuç olarak bir index tutman lazım. Sadece bir integer, nerede olduğun bilgisini tutman yeterli. remove dedin mi en son üzerinden geçtiğini remove edeceksin.
* next’i get ile implement edeceksin, yine tuttuğun indexi kullanacaksın.
* hasNext için sonuna geldik mi gelmedik mi diye bakacaksın, index ve size ile kontrol edersin.

Yani AbstractList seviyesinde Iterator implement edebilirsin ama performansı kötüdür. Örneğin LinkedList’in var ve next ile sonuna kadar gidiyorsun, performans quadratic olur. Çünkü her seferinde baştan gitmen lazım. Çünkü index ile gidiyorsun. Çünkü tek bildiğimiz index. Performansının yüksek olmasını istiyorsan LinkedList’te tekrar implement etmen lazım.

AbstractList seviyesinde Iterator performansı ArrayList için yeterlidir.

You need to implement only:

* add(int, E)
* get(int)
* remove(int)
* set(int, E)
* size()

AbstractList implements Iterator<E> using the index

… AbstractCollection, AbstractList, and AbstractSequentialList

Another more complete way to declare KWArrayList is:

public class KWArrayList<E> extends AbstractList<E> implements List<E>

Another more complete way to declare KWLinkedList is:

public class KWLinkedList<E> extends AbstractSequentialList<E>

implements List <E>