**Collection** overview:

* Dynamic in size(growable).
* Heterogeneous (mixed data types).
* Underlying data-structure (pre-defined methods available to use).

**Collection framework:** contains several classes and interfaces which can be used to represent a group of individual objects as single entity.

**9** key **interfaces** of **collection framework:**

1. **Collection(I)**
2. **List(I)**
3. **Set(I)**
4. **SortedSet(I)**
5. **NavigableSet(I)**
6. **Queue(I)**
7. **Map(I)**
8. **SortedMap(I)**
9. **NavigableMap(I)**

**Collection(I)**:

* If we want to represent a group of individual objects as single entity, then we go for Collection.
* defines the most common methods which are applicable for any Collection object. In general Collection(I) is considered as root interface of Collection framework.
* The is no concrete class which implements Collection(I) interface directly.

**List(I)** - implementation classes – **ArrayList**(C)(1.2v), **LinkedList**(C) (1.2v), **Vector**(C), **Stack**(C) (1.0v – legacy classes).

* Duplicate are allowed.
* Insertion order is preserved.
* It is child interface of Collection(I).

**Set(I)**- implementation classes- **HashSet**(1.2v), **LinkedHashSet**(1.4v)

* Duplicate are not allowed.
* Insertion order is not preserved.
* It is child interface of Collection(I).

**SortedSet(I)** is child interface of Set(I)

* + Can be used to preserve insertion order.

**NavigableSet(I) -** is child interface of **SortedSet(I)** implementation **TreeSet**(C)

* + - It provides several methods for navigation purpose.

**Queue(I)** – implementation class **PriortQueue**(C),**PriorityBlockingQueue**(C),**LinkedBlockingQueue**(C)

* It is the child interface of Collection(I).
* Prior to processing (**FIFO**).

**Map(i)**- implementation classes **HashMap(C)**, **LinkedHashMap(C)**, **WeakHashMap(C)**, IdentityHaspMap(C), Hashtable(c), Properties(c), Dictionary (AC).

* If we want to represent a group of objects as key - value pairs, then we go for Map.
* Both key and values are objects.
* Duplicate keys are not allowed.
* Map(I) is not child interface of Collection(I)

**SortedMap(I)** is child interface of Map(I)

* + Can be used to preserve insertion order.
  + Sorting order of keys

**NavigableMap(I) -** is child interface of **SortedMap(I)** implementation **TreeMap**(C)

* + - It provides several methods for navigation purpose.

Sorting – for sorting objects

**Comparable(I)** if we want default natural sorting.

**Comparator(I)** if we want customized sorting.

Cursors – to get the objects one by one.

**Enumeration (i)**

**Iterator(I)**

**ListIterator(I)**

Utility classes – present in Collection framework.

**Collections(C)**

**Arrays(C)**

**\*\* Note**

|  |  |
| --- | --- |
| **Collection** | **Collections** |
| Is an **interface**. | Is a **Utility class** present in java.util pkg |
| If we want to represent a group of individual objects as single entity. | contains several **static utility methods** for collection object. |