Collections

- The **Collection in Java** is a framework that provides an architecture to store and manipulate the group of objects.
- Java Collections can achieve all the operations that you perform on a data such as searching, sorting, insertion, manipulation, and deletion.
- Java Collection means a single unit of objects. Java Collection framework provides many interfaces (Set, List, Queue, Deque) and classes (ArrayList, Vector, LinkedList,PriorityQueue, HashSet, LinkedHashSet, TreeSet).

Why do we need a Collection framework in java?

- To store multiple objects or group of objects together we can generally use arrays, but arrays has some limitations.
- Two important hierarchies of collection framework is

1.Collection hierarchy

2.Map Hierarchy

Limitations of array

- The size of the array is fixed, we cannot reduce or increase dynamically during the execution of the program.
- Array is collection of homogenous elements.
- Array manipulation such as,

1.Removing an element from array

2.Adding the element in between the array etc requires complex logic

- To avoid the limitations of array we can store the group of objects or elements using different data structures such as
- List
- Set
- Queue
- Map

Collection Interface

- Collection is an interface defined in java.util.
- Collection interface provides the mechanism to store group of Objects(elements)together.
- All the elements in the collections are stored in the form of objects(i.e only non primitive is allowed).
- Which helps the programmer to perform the following task

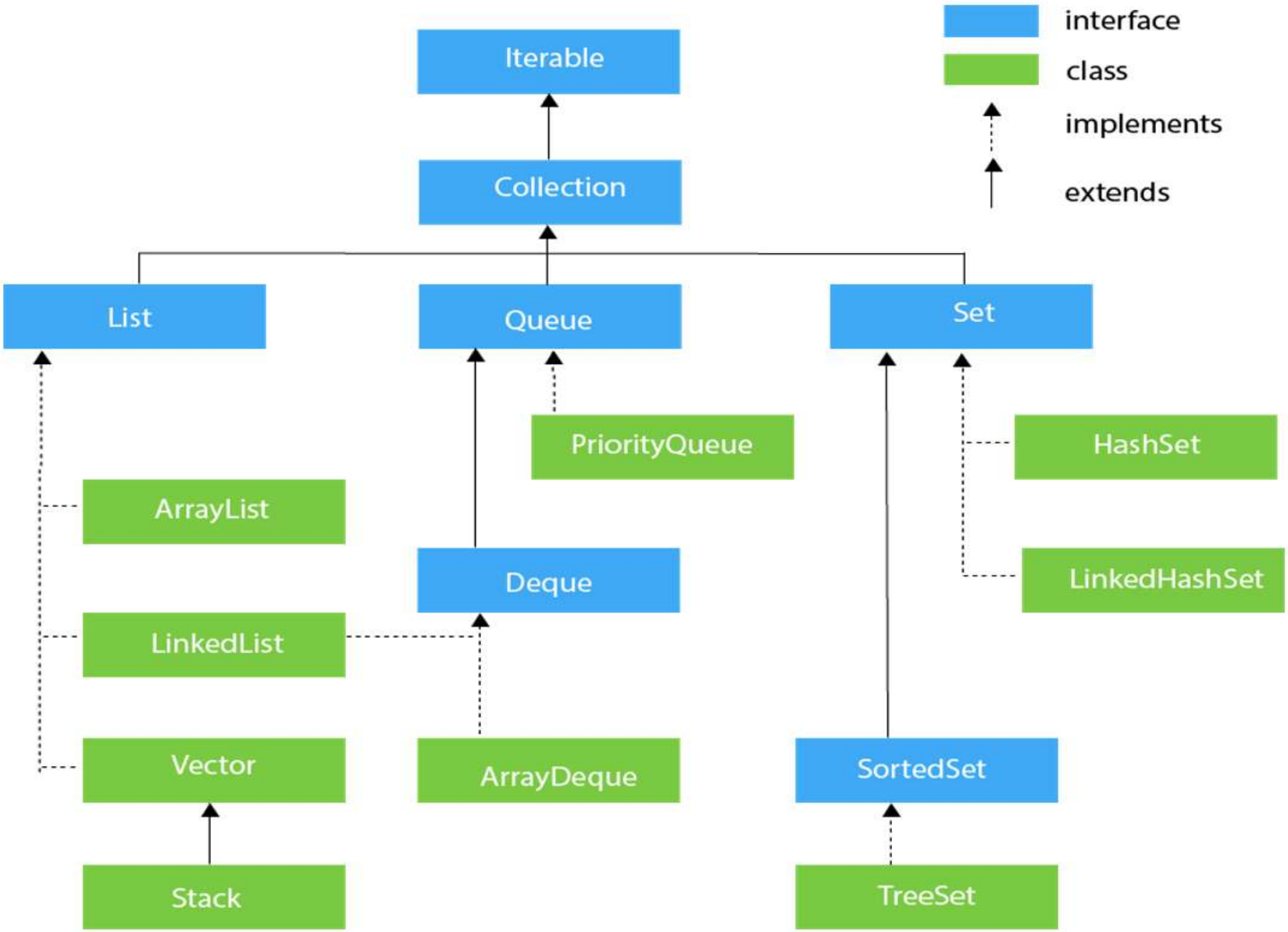
1.Add element in the collection

2.Search element in the collection

3.Remove the element from the collection

4.Access the element from the collection

Collection Hierarchy



Methods in Collection Interface

Purpose	Return type	Method
To add element	boolean	add(Object o)
		add(Collection)
To remove element	boolean	remove(Object o)
		removeAll(Collection)
		retainAll(Collection)
	void	clear()
To search element	boolean	contains(Object)
		containsAll(Collection)
To access element	Iterator	iterator()
		for each loop
miscellanous		size()
		isEmpty()
		toArray()
		hashCode()
		equals()

List Interface

- List interface is the child interface of Collection interface. It inhibits a list type data structure in which we can store the ordered collection of objects. It can have duplicate values.
- List interface is implemented by the classes ArrayList, LinkedList, Vector, and Stack.

To instantiate the List interface, we must use :

List <data-type> list1= **new** ArrayList();

List <data-type> list2 = **new** LinkedList();

List <data-type> list3 = **new** Vector();

List <data-type> list4 = **new** Stack();

There are various methods in List interface that can be used to insert, delete, and access the elements from the list.

Characteristics of List Interface

- List interface defined in java.util package.
- List is a sub interface of collection interface. Therefore it has all the methods inherited from collection interface.
- List is a ordered collection of objects(It maintains insertion order of elements).
- List has indexing, therefore we can insert,remove,or access the element with the help of index.
- List can have duplicate objects.

- The elements in the ArrayList can be accessed in the following ways

1.get()

2.Iterator

3.for each loop

1)get()

- Is used to access the elements present in the arraylist
- It accepts index as an argument
- It returns the object type

2)To access the elements of arraylist using iterator

- iterator() belongs to iterable interface.
- iterator() creates an iterator type object and returns reference of iterator type.

Iterator interface has two methods:

1.hasNext(): It checks whether there is an element to be accessed from collection, if present it returns true, else it returns false.

2.next():it is used to access the element and moves the cursor to the next element. The return type of next() is object.

for each loop:

Syntax:

```
for(data_type variable : array | collection)
{
        //body of for-each loop
}
```

Example

```
class ForEachExample1{
  public static void main(String args[]){
   //declaring an array
   int arr[]={12,13,14,44};
   //traversing the array with for-each loop
   for(int i:arr){
     System.out.println(i);
   }  }}
```

ArrayList

- The ArrayList class implements the List interface. It uses a dynamic array to store the duplicate element of different data types. The ArrayList class maintains the insertion order.

```
class ArrayListDemo{
public static void main(String args[]){
ArrayList list=new ArrayList();//Creating arraylist
list.add(“Karthik");//Adding object in arraylist
list.add("Vinay");
list.add(20);
list.add("Alok");
//Traversing list through Iterator
Iterator itr=list.iterator();
while(itr.hasNext()){
System.out.println(itr.next());
}
}
}
```