Collection Framework

Q. What is the Collection?

Collection is ready made implementation of data structure provided by java to us.

Q. Why use the Collection or what is the benefit of Collection?

1) Ability to store the different type of data

2) Collection can change its dynamic size at run time as well as shrink its size at run time.

3) Collection can provide the readymade data structure algorithm like as inserting element, deleting element, searching element etc

Note: if we use the Object class array in java then we can store different type of data in it also.

```
public class ObjArrApp
{
    public static void main(String x[])
    {
        Object obj[]=new Object[5];
        obj[0]=100;
        obj[1]="good";
        obj[2]=5.4f;
        obj[3]=false;
        obj[4]=new java.util.Date();
       for(int i=0; i<obj.length; i++)
      {
         System.out.println(obj[i]);
      }
    }
}
```

```
C:\Program Files\Java\jdk1.8.0_291\bin>javac ObjArrApp.java

C:\Program Files\Java\jdk1.8.0_291\bin>java ObjArrApp
100
good
5.4
false
Mon Aug 16 10:04:41 IST 2021

C:\Program Files\Java\jdk1.8.0_291\bin>
```

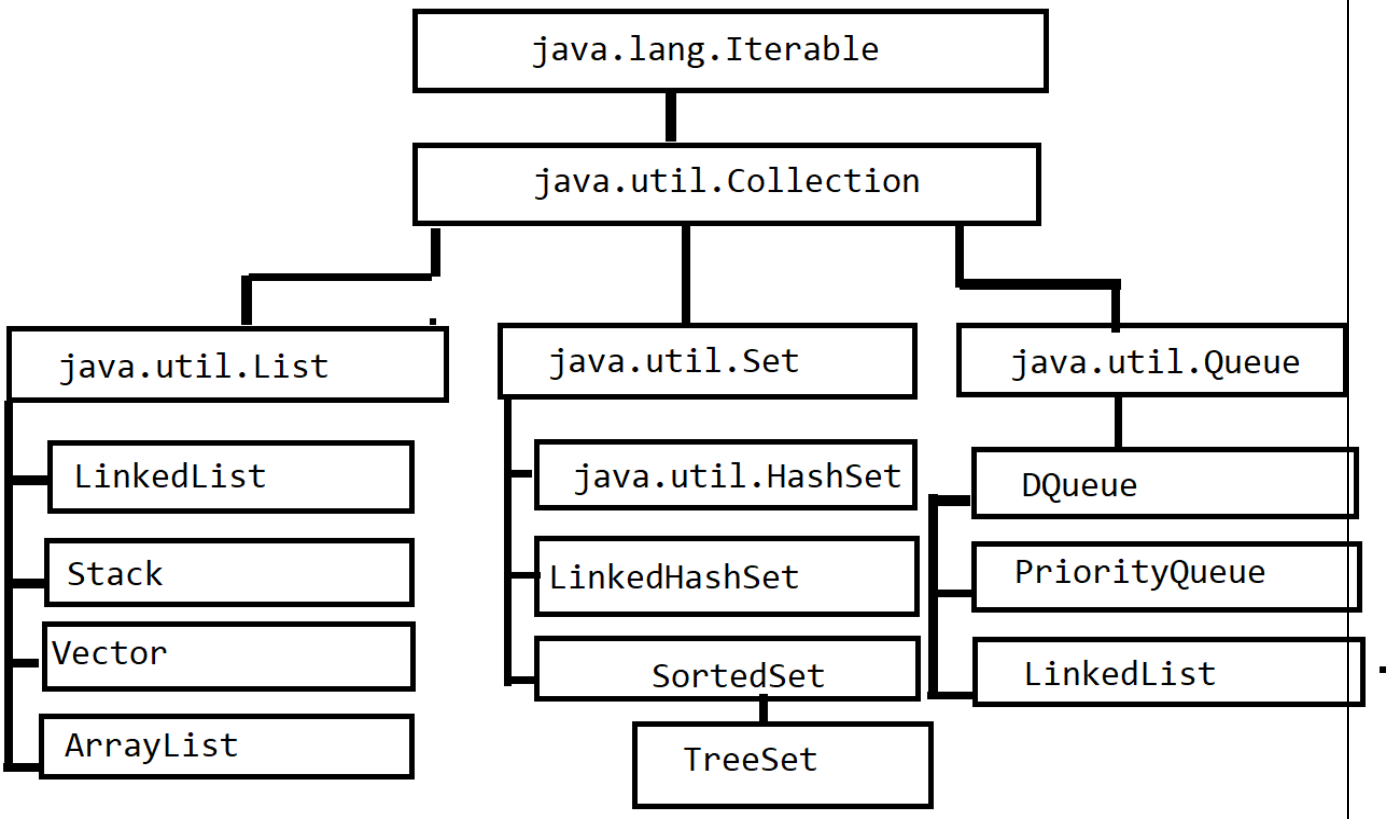
If we think about above diagram then we can store the different type of data in Object class array.

But there is some limitation

1) If we want to perform any data structure operation we need to perform manually like as inserting data on specified position, deleting data from specified position, searching data etc

If we want to avoid this problem in java we have the Collection Framework.

If we want to work with Collection Framework in java we have the Following Class Hierarchy



If we think about above diagram we have the top most interface name as java.lang.Iterable

What is the Role of Iterable interface in Collection API?

The Major goal of Iterable interface is to fetch data from collection framework. Because data fetching is common operation in Collection API so they design the Iterable as parent to all and provide the methods to us for data fetching purpose.Iterable is interface from java.lang package and which provide the iterator() method to us for data fetching purpose and iterator method return the reference of Iterator() interface.

Internally Iterable Look Like as

Syntax

```
public interface java.lang.Iterable<T> {
  public abstract java.util.Iterator<T> iterator();
  public void forEach(java.util.function.Consumer<? super T>);
  public java.util.Spliterator<T> spliterator();
}
   Note: forEach() and splitIterator() we will discuss later
```

Iterator interface

Iterator interface return by iterator () method declared within Iterable interface and Iterator interface provide the following method to us for fetching data from any type of Collection

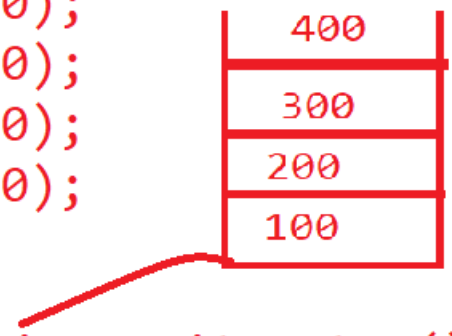
boolean hasNext(): this method check wheather element present in collection or not if present return true otherwise return false.

Object next(): this method is used for fetch data from collection and move cursor on next element.

void remove(): this method can remove the data from collection at the time of data fetching.

Following Code Example Describe The Working of Iterator interface

```
Stack s = new Stack();
s.add(100);
s.add(200);
s.add(300);
s.add(400);
```




```
Iterator i = s.iterator();

while(i.hasNext())
{
   Object obj = i.next();

 System.out.println(obj);
}
```

Output

```
100
200
300
400
```

Working of Above Code

In above code we create the object of Stack class and store the four element in 100,200 ,300 and 400 using add method() of Stack. When we call the statement Iterator i = s.iterator() means your reference of Iterator i.e i in our example points to Stack.

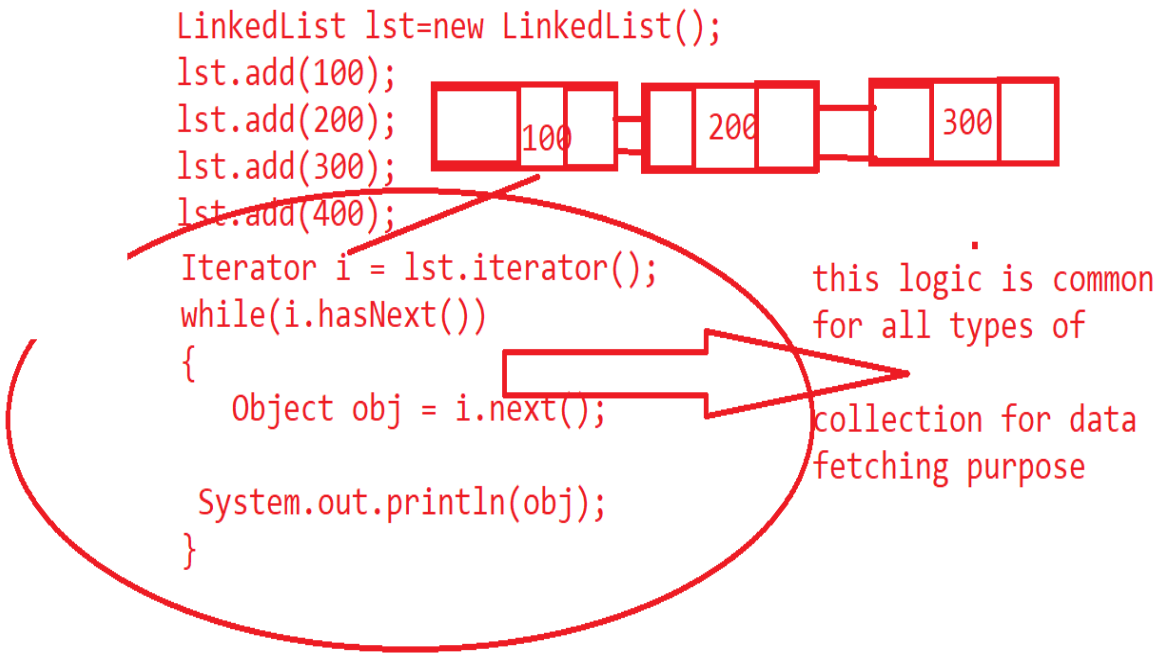
While loop working shown below

As per our example first Iterator reference i.e i points 100 element it is first element in stack. Means while(i.hasNext()) first check 100 present in stack or not so in our example 100 is present in Stack so i.hasNext() method return true means your loop will be execute and then next statement get executed i.e i.next() this method can fetch element from collection and move cursor on next element i.e 200 as per our example so again while loop get executed means while(i.hasNext()) get executed and again while(i.hasNext()) check second element present in stack or not means it will check 200 as per our example so 200 present in stack so this statement return true and then again loop executed and i.next() method get executed and this method fetch 200 from stack and move cursor on next element i.e 300 an so on when all element fetch by Iterator and if element not present in stack then while(i.hasNext()) method return false and loop get terminated.

Why Iterable provide the iterator() and refence of Iterator interface

The Major benefit of iterator it is parent of all types of collection means we can use the Iterator interface by using iterator() method with all types of collection. Means we can use the above mention logic with stack for data fetching with all types of collection

Sample code Example



In collection hierarchy we have the second top most interface is java.util.Collection

Q. why java.util.Collection interface is parent of all collection APIs

Collection provide the methods to us for common operation with collection means Collection interface contain methods which is commonly required to all types of collection.

Common operation with collection

1) adding element collection

2) removing element from collection

3) search element from collection

4) count the number of element in collection

Etc

Means above operation is required to all types of collection means above operation can perform with stack, queue, linkedlist ,vector,array etc So Collection api design the Collection interface and provide the common method to all types of collection for perform above operation.

Methods of Collection interface

```
public interface java.util.Collection<E> extends java.lang.Iterable<E> {
  public abstract int size();
  public abstract boolean isEmpty();
  public abstract boolean contains(java.lang.Object);
  public abstract java.util.Iterator<E> iterator();
  public abstract java.lang.Object[] toArray();
  public abstract <T> T[] toArray(T[]);
  public abstract boolean add(E);
  public abstract boolean remove(java.lang.Object);
  public abstract boolean containsAll(java.util.Collection<?>);
  public abstract boolean addAll(java.util.Collection<? extends E>);
  public abstract boolean removeAll(java.util.Collection<?>);
  public boolean removeIf(java.util.function.Predicate<? super E>);
  public abstract boolean retainAll(java.util.Collection<?>);
  public abstract void clear();
  public abstract boolean equals(java.lang.Object);
  public abstract int hashCode();
  public java.util.Spliterator<E> spliterator();
  public java.util.stream.Stream<E> stream();
  public java.util.stream.Stream<E> parallelStream();
}
```

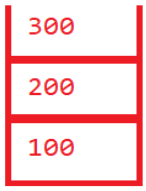
Now we will discuss about method of Collection interface

int size(): this method is used return number of element present in collection.

```
Stack s = new Stack();
s.add(100);
s.add(200);
s.add(300);
     3
int nelement = s.size();

System.out.println("Number of element  "+nelement);

Output:
Number of element   : 3
```



boolean isEmpty(): this method check wheather collection is empty or not if empty return true otherwise return false.

```
ArrayList al = new ArrayList();
al.add(100);
al.add(200);
al.add(300);
      false
boolean b = al.isEmpty();

if(b) if(false)
{
  System.out.println("ArrayList is     empty");
}
else
{
  System.out.println("ArrayList is not empty");
}

 Output:  ArrayList is not empty
```

If we think above code then we call the isEmpty() method of ArrayList and ArrayList contain 3 element so it is not empty so isEmpty() method return false in variable b and if we pass b in if statement so if return false so else get executed so we get output ArrayList is not empty.

boolean contains(Object value): this method is used for searching purpose means if we want to search specific element in collection framework then we can use the contains method and this method return true if we found the element in collection otherwise return false.

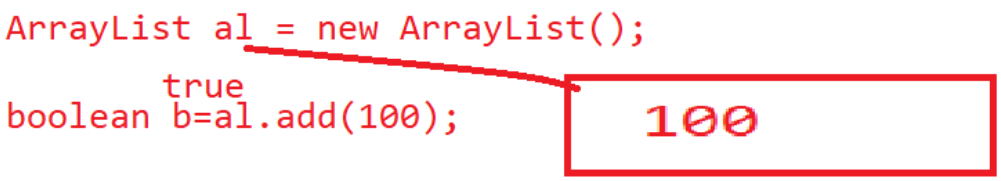
```
ArrayList al = new ArrayList();
al.add(100);
al.add(200);
al.add(300);
        true
boolean b = al.contains(200);

if(b) if(true)
{
  System.out.println("element found");
}
else
{
   System.out.println("Element not found");
}

Output :
Element Found
```

If we think about above code we pass 200 value in contains method and 200 persent in ArrayList so this method return true in variable b and we pass variable b in if statement means if contain true value so if get executed and we found the element found.

boolean add(E): this method is used for add the element in collection if element added then return true otherwise return false.




```
if(b) if(true)
{
  System.out.println("Element Added ");
}
else
{
   System.out.println("Eleement Not Added");
}

Output:
_____________
Element Added.
```

If we think about above code we call the add() method with 100 value when element added in collection this method return true in variable b and we pass b in if means if contain true so we get output element Added

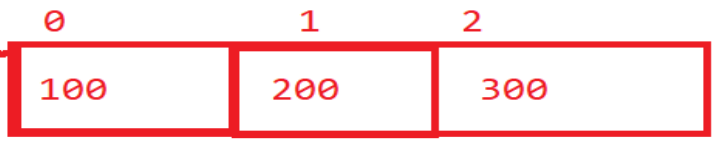
boolean remove(Object value): this method can remove the element from collection and if element remove from collection return true otherwise return false.

```
ArrayList  al = new ArrayList();

al.add(100);
al.add(200);
al.add(300);

boolean  b = al.remove(200);

if(b)
{
 System.out.println("Element Removed Success");
}
else
{
  System.out.println("element not removed");
}
```



If we think about above code we pass the 200 value in remove() method and this value present in ArrayList and remove() method remove the value from ArrayList and return true value in variable b and we pass the b in if statement and b contain true means if also true and we get output element removed success.

Basically there are three types of collection in java

Types of Collection in java

1) List: List collection can store the any type of data and infinite size and arrange data using indexing format but allow the duplicated values or data.

2) Set: Set Collection can store any type of data and infinite size and manage data using hasing technique and not allow the duplicate elements.

3) Queue: Queue is collection can manage data in first in first out format and arrange data using indexing and allow the duplicated data

Now we will Discuss about the List Collection