METHODS IN COLLECTON AND LISTS

**Collection Methods:**

1.add(Object o)

2.addAll(Collection C)

3.remove(Object o)

4.removeAll(collection C)

5.retainAll(Collection C)

6.Clear()

7.isEmpty()

8. Size()

9.contains()

10.containsAll(Collection e)

11.toArray(Collection C)

**List Methods**

(arraylist , linked list)

1.add(Object o)

2.add(index,Object o)

3.addAll(index , Collection c)

4.remove(index)

5.get(index)

6.set(index,Object o)

+ collection Methods

**Array List**

(heterogenous data storage)

1.ArrayList al = new ArrayList( );

2.ArrayList <String> al = new <String>();

Methods

1.All the Collection methods

2.add(index,Object o)

3.addAll(index , Collection c)

4.set(index,Object o)

5.Collections..sort(al)

6.Collections.shuffle(al)

//Declare ArrayList

//ArrayList al=new ArrayList();

//ArrayList <Integer> al=new ArrayList<Integer>();

//ArrayList <String> al=new ArrayList<String>();

// List al = new ArrayList();

Add new elements to the arraylist

al.add(100);

al.add(“welcome”);

al.add(‘A’);

System.out.Println(al)

Three approaches to read the arraylist

1. for loop

for(int i=0;i<=al.size();i++)

{System.out.Println(al.get(i));

}

1. for ..each loop

for(Object e:al)  
{  
 System.*out*.println(e);  
}

1. iterator()

Iterator it = al.iterator();  
while(it.hasNext())  
{  
 System.*out*.println(it.next());  
}

ADD AND REMOVE ALL

ArrayList al=new ArrayList();  
  
 al.add("b");  
 al.add("a");  
 al.add("c");  
 al.add("d");  
 al.add("g");  
  
ArrayList al2 = new ArrayList();  
al2.addAll(al);  
al2.removeAll(al);  
System.*out*.println(al2);

SORT IN ARRAY LIST

ArrayList al=new ArrayList();  
  
 al.add("b");  
 al.add("a");  
 al.add("c");  
 al.add("d");  
 al.add("g");  
  
ArrayList al2 = new ArrayList();  
al2.addAll(al);  
  
  
Collections.*sort*(al);  
System.*out*.println(al);

SORT ARRAYLIST IN REVERSE ORDER

ArrayList al=new ArrayList();  
  
 al.add("b");  
 al.add("a");  
 al.add("c");  
 al.add("d");  
 al.add("g");  
  
ArrayList al2 = new ArrayList();  
al2.addAll(al);  
  
  
Collections.*sort*(al,Collections.*reverseOrder*());  
System.*out*.println(al);

SHUFFLE

Collections.shuffle(al);

TO CONVERT ARRAYS TO ARRAYLIST

String arr[] = {"Dog","cat"};  
for(String value:arr)  
{  
 System.*out*.println(value);  
}  
  
ArrayList al = new ArrayList(Arrays.*asList*(arr));  
System.*out*.println(al);

**ARRAYS WHEN TO USE** : TO HAVE MORE NUMBER OF RETRIEVING OPERATIONS

**WHEN SHOULD WE USE LINKED LIST :** INSERT AND DELETE

We should not use linked list when we have more number of retrievals operation

**1.**insertion/deletion: Linked List

2.Retriving : Arrays

Most of the times linked list are used to Create stacks and queues.

STACK : FILO

QUEUE: FIFO

**LINKED LIST ATTRIBUTES**

1. DUPLICATE ENTRIES
2. INSERTION ORDER
3. NULL

**GENERAL METHODS**

1. **add(x)**
2. **add(index,obj)**
3. **addAll(Collection)**
4. **remove(obj)**
5. **remove(index)**
6. **removeAll(Collection)**
7. **l.get(index)**
8. **l.set(index,obj)**

**SPECIAL MEHTODS OF LINKED LIST**

1. **addFirst(obj)**
2. **addLast(obj)**
3. **removeFirst(obj)**
4. **removeLast(obj)**
5. **getFirst(obj)**
6. **getLast(obj)**

//LinkedList l=new LinkedList();  
//LinkedList <Integer> s = new LinkedList<Integer>();  
  
LinkedList l = new LinkedList();  
l.add(100);  
l.add("wel");  
l.add(3.0);  
l.add('a');  
l.add(true);  
l.add(null);  
System.*out*.println(l);

//LinkedList l=new LinkedList();  
//LinkedList <Integer> s = new LinkedList<Integer>();  
 // l.add(3,"Java")  
 //l.remove(3)  
 //l.size()  
 //l.get(2)

//l.set(5,’X’)

//l.contains(‘Java’)//true

//l.contains(‘python’)//false

//l.isEmpty()false

**READING ELEMENTS FROM LINKED LIST USING FOR LOOP**

for(int i=0;i<l.size();i++)

{

l.get(i);

}

**READING ELEMENTS FROM LINKED LIST USING for.. each LOOP**

for(Object e:l)

{

System.out.println(e);

}

**READING ELEMENTS FROM LINKED LIST USING ITERATOR**

Iterator it =l.iterator();

While(it.hasNext())

{

System.out.println(it.next());

}

LinkedList u = new LinkedList();  
u.addAll(l);  
System.*out*.println(u);  
u.remove(0);  
u.removeAll(u);  
 Collections.*sort*(u,Collections.*reverseOrder*());  
System.*out*.println(u);  
 Collections.*sort*(u);  
 System.*out*.println(u);  
 Collections.*shuffle*(u);  
 System.*out*.println(u);

SPECIAL METHODS

l.addFirst("tiger");  
l.addLast("hey");

System.*out*.println(l.getFirst());  
System.*out*.println(l.getLast());  
  
l.removeFirst();  
l.removeLast();