LIST

- WHAT IS COLLECTION?

  collection is simply an object that represent a group of objects, known as elements.

- KEY INTERFACES IN COLLECTION FRAMEWORK?

  (1) COLLECTION: Root interface for all other collection types.

  (2) List: order collection that contains duplicate elements.

  (3) Set : UnOrder collection that doesn't contain duplicate.

  (4) Queue: a collection designed for holding elements prior to processing.

  (5) Deque: A double ended queue that allow insertion and removal from both ends.

  (6) map : an interface represents collection of key - value pair .

- KEY FEATURES OF LIST INTERFACE?

  > Order Maintain

  > Index based access

  > Allow duplicates

- LIST INTERFACE CLASSES?

  (1) ArrayList

  (2) LinkedList

  (3) Vector

  (4) Stack

  (5) CopyOnWriteArrayList

- IN TERNAL WORKING OF ARRAYLIST ?

  > uses dynamic array which can shink and grow according to needed,

  > default capacity 10

  > when we add element first check capacity if full it will create new larger capacity with 1.5 times according to initial capacity

and copy elements from old array and for copying elements takes O(n) times complexity

> when we remove first check index is within valid range, after remove elements shifted to left position by 1

- DIFFERENT WAYS TO CREATE ARRAYLIST?

(1) ArrayList<String> list = new ArrayList<>();

(2) List<String> list2 = Arrays.asList("Monday","tuesday"); it can't be return type ArrayList because it's private class.

(3) String [] str = {"Apple", "Banana","Mango"};

List<String> list3 = Arrays.asList(str);

(4) List<String> list4= List.of("sam","sahil","alam"); // immutable

(5) List<String> list5 = new ArrayList(list4); // for making modifiable list

- CODE FOR FOLLOWING LIST METHODS ?

```
public static void main(String[] args) {
  ArrayList<Integer> list = new ArrayList<>();


  // add ()
  list.add(5);
  list.add(9);
  list.add(15);

  // get ()
  System.out.println(list.get(2));


  // size ()
  System.out.println(list.size());


  // contains ()
  System.out.println(list.contains(5));


  // remove () with passing index
  System.out.println(list.remove(2));


  // remove () with passing object
  System.out.println(list.remove((Integer) 9));


  // add () with particular index like between 1 and 2 index
  list.add(2,50);
```

```
        // set () which will replace existing one
        list.set(2,30);


        // addAll ()
        List<Integer> li= new ArrayList<>();
        list.addAll(li);


        // toArray ()
        Integer[] integers = list.toArray(new Integer[0]);


        // sort () with list
        Collections.sort(list);


        // sort () with null
        list.sort(null);

        // for printing all array list elements way: 1
        for (int i=0; i<list.size(); i++){
            System.out.println(list.get(i));
        }


        // for printing all array list elements way: 2
        for (Integer el : list) {
            System.out.println(el);
        }


    }
```

- COMPARATOR ?

```
package org.example.LIST_1.ARRAYLIST;


import java.util.Comparator;

class com implements Comparator<String>{


    // normal logic for natural ascending sorting with integer.

    /*
     @Override
    public int compare(Integer o1, Integer o2) {
        return o1-o2;
    }
     */
```

```
    // normal logic for natural descending sorting with integer.

    /*
     @Override
    public int compare(Integer o1, Integer o2) {
        return o2-o1;
    }
     */


    // for string comp according to string length
    @Override
    public int compare(String o1, String o2) {
        return o1.length()-o2.length();
    }


}


public class Comparator_02 {
    public static void main(String[] args) {


        // normal logic for natural ascending sorting with integer.

        /*List<Integer> list = Arrays.asList(2, 3, 1, 4, 7, 6);
        list.sort(new com());
        System.out.println(list);*/


        // normal logic for natural descending sorting with integer.
       /*
        List<Integer> list = Arrays.asList(2, 3, 1, 4, 7, 6);
        list.sort(new com());
        System.out.println(list);
        */


        // list for string
     /*
        List<String> list = Arrays.asList("banana","apple","date");
        list.sort(new com());
        System.out.println(list);
        */


        // using java 8 for sorting
        /*
        list.sort((a,b)->b.length()-a.length());
        System.out.println(list);
         */
 }
}
```