L57

Monday, February 7, 2022 2:23 PM

Data Structures - List, ArrayList

Saurav Hathi

https://www.youtube.com/channel/UCp6MFWao5vWRnyRCxBsKnfw

https://www.instagram.com/saurav_hathi/

List interface

Q1. A List represents an ordered collection which allows duplicate elements.

List also allows index-based access of elements like an array. It can be considered to be a flexible array.

Below are some of the concrete classes which implement List interface:

1. ArrayList - most commonly used class whenever we want a flexible array behaviour
2. LinkedList - used when we want a doubly-linked list behaviour. It has extra methods like *addFirst*, *addLast* which are not present in **List** interface
3. Vector - is a legacy class. ArrayList is recommended to be used instead of Vector.
4. Stack - provides last-in-first-out (LIFO) implementation for objects. It is also a legacy class. It extends Vector. ArrayDeque is a recommended to be used for a LIFO functionality instead of this class.

Apart from the methods like add(E e), remove(Object obj), clear(), size(), etc., which the List interface inherits from its super interface Collection, it provides some extra index-based methods as given below:

1. **add**(int index, E element) - inserts the element at the given index.
2. **addAll**(int index, Collection c) - inserts all elements of Collection c at the given index.
3. **get**(int index) - returns the element at the given index in the List.
4. **remove**(int index) - removes the element at the given index in the List.
5. **set**(int index, E element) - replaces the element at the given index in the List, and returns the previous element at that index.

There are many more methods in the List interface, above mentioned are some of the most commonly used ones.

Read the instructions mentioned in the comments and fill in the missing code.

```
package q11366;
import java.util.*;
public class ListDemo
{
  public static void main (String[]args)
  {
    List namesList = new ArrayList ();
    /*
       Below code adds two elements into the namesList
     */
     namesList.add ("John Napier");
     namesList.add ("Isaac Newton");
     System.out.println (namesList);
    /*
     Below code adds "C V Ramana" to the namesList at index 1
     */
     namesList.add (1, "C V Ramana");
     System.out.println (namesList);
    /*
       Insert a line of code below this comment
       which adds "Charles Babbage" to the namesList at index 1
       Hint: See how "C V Ramana" was added above
     */
     namesList.add (1, "Charles Babbage");
     System.out.println (namesList);
    /*
       Insert a line of code below this comment

       which removes the element at index 0

       Hint : Use the remove(...) method
     */
     namesList.remove (0);
     System.out.println (namesList);
     namesList.set (2, "Bose");
     System.out.println (namesList);
  }
}
```

ArrayList - methods and usage

Q1. Whenever we want a growable array implementation we use an ArrayList.

ArrayList has 3 constructors.

1. ArrayList() - the default constructor creates an empty ArrayList
2. ArrayList(Collection c) - it creates an ArrayList with the contents of the collection passed as argument.
3. ArrayList(int initialCapacity) - it creates an empty ArrayList with the given initial capacity.

ArrayList internally stores all the references of added elements in an array. The initial array size depends on which of the above three constructors is used to create the ArrayList instance.

The size of this internal array is called **capacity.**

We neither access this internal array, nor should be bothered about this array. It is useful to know about this array to understand the difference between the terms size and capacity.

When we refer to the size of an ArrayList, we are talking about the **count of elements stored** in that array.

When this array is filled with elements to its capacity, in order to accommodate new elements, **ArrayList** silently replaces the filled array with a new array of bigger capacity. It also restores all the existing elements in the old array into this new array before performing the add operation with the new element.

Note that whenever you call the size() method on an ArrayList, it always returns the current count of elements it holds. It has nothing to do with the internal capacity.

When we know the count of elements we will be storing in an ArrayList, it is efficient to provide it as the initialCapacity (as a argument to the ArrayList constructor) so that the ArrayList can avoid the internal capacity adjustments while elements are being added.

For example, if we create an ArrayList called cList with an **initialCapacity** of 20, and do not add any elements to cList, it still remains an empty list. Meaning, it has an internal capacity to store 20 elements before resizing itself. As long as there are no elements added to the cList, a call to the size() method on cList will always return 0, even though its internal capacity is 20.

Follow the instructions provided in the comments in the below program and accordingly fill in the missing code.

```
package q11367;
import java.util.*;
public class ArrayListDemo
{
  public static void main (String[]args)
  {
    /*
       In the below lines we are creating an empty ArrayList

       called aList and printing its size and elements.

       Printing an ArrayList instance (aList) will print its elements in brackets.

       For example, when we print aList, we will notice [] (empty brackets),

       as aList is empty.
     */
      List aList = new ArrayList ();
      System.out.println ("aList.size() = " + aList.size ());
      System.out.println ("aList = " + aList);

    /*
       The below line of code adds a string element

       called "First Entry" to aList
     */

      aList.add ("First Entry");

    /*
       Insert a line of code below this comment

       to add a string element called "Second Entry" */ aList.
      add ("Second Entry");
      System.out.println ("aList.size() = " + aList.size ());
      System.out.println ("aList = " + aList);
      List bList = new ArrayList (aList);
      System.out.println ("bList.size() = " + bList.size ());
      System.out.println ("bList = " + bList);
      List cList = new ArrayList (20);
      System.out.println ("cList.size() = " + cList.size ());
      System.out.println ("cList = " + cList);
  }
}
```

Q2. See the code and retype the same to learn how to iterate over the elements stored in a ArrayList.

The class scans through all the arguments passed to the main method, and stores them into an ArrayList if the argument's first char is in uppercase.

The program first uses the **for-each** loop to print all the stored names from the ArrayList one name on each line.

It later uses a normal **for statement** to iterate over the elements of the ArrayList. Note the usage of the get(int index) method. This method of iteration is used when we also want to keep track of the index of the element being retrieved.

```
package q11368;
import java.util.*;
public class ArrayListIterationDemo {
    public static void main(String[] args) {
        List namesList = new ArrayList();
        for (String argument : args) {
            if (Character.isUpperCase(argument.charAt(0))) {
                namesList.add(argument);
            }
        }
        for (Object name : namesList) {
            System.out.println(name);
        }
        for (int i = 0; i < namesList.size(); i++) {
            Object name = namesList.get(i);
            System.out.println("Name at index " + i + " is : " + name);
        }
    }
}
```

Q3. See and retype the below code to familiarize yourself with some of the commonly used methods in ArrayList.

The class iterates through all the arguments passed to the main method, and stores them into an ArrayList, which is later manipulated using its methods.

Correlate the code and output to understand the usage of the methods.

```
package q11369;
import java.util.*;
public class ArrayListMethodsDemo {
    public static void main(String[] args) {
        List aList = new ArrayList(args.length);
        for (String argument : args) {
            aList.add(argument);
        }
        System.out.println("aList = " + aList);
        System.out.println("aList.size() = " + aList.size());
        Object removedElement = aList.remove(3);
        System.out.println("removedElement = " + removedElement);
        System.out.println("aList = " + aList);
        Object replacedElement = aList.set(0, "Steve Jobs");
        System.out.println("aList = " + aList);
        aList.add(0, "Bill Gates");
        System.out.println("aList = " + aList);
    }
}

```

Practice Programs on List

Q1. Write a program to understand how to insert elements into an ArrayList using the add method .

Create a class ListDemo with a main method. Create an instance of ArrayList and add days of week from Sunday through Saturday to the list and print the same.

The result should be as follows:
[Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday]

Note: complete the code between the comments:
// write your code below this
// write your code above this

```
package q11956;
import java.util.*;
public class ListDemo
{
  public static void main (String[]args)
  {
    // Notice the use of generics. We will learn more about them later.
    // The type parameter <String> will ensure that your code cannot add any
    // other object than those of type String
    List < String > namesList = new ArrayList < String > ();
    // write your code below this

    namesList.add ("Sunday");

    namesList.add ("Monday");

    namesList.add ("Tuesday");

    namesList.add ("Wednesday");

    namesList.add ("Thursday");

    namesList.add ("Friday");

    namesList.add ("Saturday");

    // write your code above this
    System.out.println (namesList);
  }
}

```

Q2. Write a program to understand how to remove elements from an ArrayList using the remove method .

Create a class ListDemo with a main method. Follow the given instructions while writing the program:

- Create an ArrayList with the following elements: Mercury, Venus, Earth, Mars.
- Print all the elements in the list
- Remove the element at index 1
- Print the resultant list

The result should be as follows:
[Mercury, Venus, Earth, Mars]
[Mercury, Earth, Mars]

```java
package q11957;
import java.util.*;
public class ListDemo
{
  public static void main (String[]args)
  {
    List namesList = new ArrayList ();
    // Write your code here


      namesList.add ("Mercury");
      namesList.add ("Venus");
      namesList.add ("Earth");
      namesList.add ("Mars");


      System.out.println (namesList);
      namesList.remove (1);


      System.out.println (namesList);
  }
}

```

Q3. Write a program to understand how to retrive an element in an ArrayList using the method get.

Create a class ListDemo with a main method and get the element at index 1.

Write the missing code in the below program.

```java
package q11958;
import java.util.*;
public class ListDemo {
    public static void main(String[] args) {
        List<String> namesList = new ArrayList<String>();
        namesList.add("Mercury");
        namesList.add("Venus");
        namesList.add("Earth");
        namesList.add("Mars");
        System.out.println(namesList.get(1)); // write your logic here
    }
}

```

Q4. Create a class ListDemo with a main method. Create an instance of ArrayList and change the element at the given index position using the set method.

Write the missing code in the below program. Follow the instructions given in the program.

```java
package q11959;
import java.util.*;
public class ListDemo {
    public static void main(String[] args) {
        List namesList = new ArrayList();
        namesList.add("Mercury");
        namesList.add("Venus");
        namesList.add("Earth");
        namesList.add("Mars");
        System.out.println(namesList);

        // change the element at index 0 to Sun
        namesList.set(0,"Sun");

        // change the element at index 2 to Jupiter
        namesList.set(2,"Jupiter");


        System.out.println(namesList);

    }
}

```

Q5. Write a program to understand how to insert all the elements of one ArrayList to another ArrayList using the method addAll().

Create a class ListDemo with a main method. Follow the given instructions while writing the program.

- Create an instance of ArrayList and add the following elemetns : Mercury, Venus, Earth, Mars.
- Create another instances of ArrayList and add the following elements: Jupiter, Saturn, Uranus, Neptune
- Add all the elements in the Second ArrayList to first and print the result.

Write the missing code where comments line are given.

```java
package q11960;
import java.util.*;
public class ListDemo
{
  public static void main (String[]args)
  {
    List < String > namesList1 = new ArrayList < String > ();

    // add List1 elements here
    namesList1.add ("Mercury");
    namesList1.add ("Venus");
    namesList1.add ("Earth");
    namesList1.add ("Mars");

    System.out.print ("Elements in List1: ");
    System.out.println (namesList1);

    List < String > namesList2 = new ArrayList < String > ();

    // add List2 elements here
    namesList2.add ("Jupiter");
    namesList2.add ("Saturn");
    namesList2.add ("Uranus");
    namesList2.add ("Neptune");

    System.out.print ("Elements in List2: ");
    System.out.println (namesList2);
    System.out.println ("After adding List2 elements to List1");

    // add all List2 elements to List1
    namesList1.addAll (namesList2);
    for (Object name:namesList1)
      {
    System.out.println (name);
      }
    System.out.println ("After adding List2 elements at index 2 the list becomes:");
    // add all LIst2 elements at index 2
    namesList1.addAll (2, namesList2);
  for (Object name:namesList1)
      {
    System.out.println (name);
      }
  }
}
```

Q6. Write a program to understand how to insert and get elements into an ArrayList using the add and get methods .

Create a class ListDemo with a main method. Create an instance of ArrayList and add days of week from Sunday through Saturday to the list and print the same, also get the element at index **3** and print the result

The result should be as follows:
[Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday]
The element at the given index is Wednesday

Note: complete the code between the comments:
// write your code below this
// write your code above this

```java
package q24078;
import java.util.*;
public class ListDemo
{
  public static void main (String[]args)
  {
    // Notice the use of generics. We will learn more about them later.
    // The type parameter <String> will ensure that your code cannot add any
    // other object than those of type String
    List < String > namesList = new ArrayList < String > ();
    // write your code below this
    namesList.add ("Sunday");

    namesList.add ("Monday");

    namesList.add ("Tuesday");

    namesList.add ("Wednesday");

    namesList.add ("Thursday");

    namesList.add ("Friday");

    namesList.add ("Saturday");

    // write your code above this
    System.out.println (namesList);
    // get the element at index 3 and print the same
    System.out.println ("The element at the given index is " +
            namesList.get (3));
  }
}
```

Q7. Create a class ListDemo with a main method. The method takes inputs from the command line arguments. Create an instance of ArrayList and add these inputs to the the list and print the same.

Sample Input and Output:

Cmd Args : Ganga Yamuna Krishna Godavari
[Ganga, Yamuna, Krishna, Godavari]

```java
package q24079;
import java.util.*;
public class ListDemo {
    public static void main(String[] args) {
        // Notice the use of generics. We will learn more about them later.
        // The type parameter <String> will ensure that your code cannot add any
        // other object than those of type String
        List<String> namesList = new ArrayList<String>();
        // write your code below this
        for(int i=0;i<args.length;i++){

                        namesList.add(args[i]);

        }



        // write your code above this
        System.out.println(namesList);

    }
}

```

Q8. Create a class ListDemo with a main method. The method takes inputs from the command line arguments. Create an instance of **ArrayList** and add these inputs to the the list and print the same, and also get the element at index 2. Print the output as shown in the example.

Sample Input and Output:
Cmd Args : Ganga Krishna Godavari Sindu Narmada
[Ganga, Krishna, Godavari, Sindu, Narmada]
The element at index 2 is Godavari

```java
package q24081;
import java.util.*;
public class ListDemo {
    public static void main(String[] args) {
        // Notice the use of generics. We will learn more about them later.
        // The type parameter <String> will ensure that your code cannot add any
        // other object than those of type String
        List<String> namesList = new ArrayList<String>();
        // write your code below this
        for(int i=0;i<args.length;i++){

                        namesList.add(args[i]);

        }

        // write your code above this
        System.out.println(namesList);
        System.out.println("The element at index 2 is "+namesList.get(2));
        // get the element at index 2 and print the same

    }
}

```

Q9. Create a class ListDemo with a main method. The method takes inputs from the command line arguments. Create an instance of **ArrayList** and add these inputs to the the list and print the same, and remove the element at index 2. Print the output as shown in the example.

Sample Input and Output:
Cmd Args : Red Blue Green Yellow
Before removing the element at index 2 the list is
[Red, Blue, Green, Yellow]
After removing the element at index 2 is the list is
[Red, Blue, Yellow]

```java
package q24082;
import java.util.*;
public class ListDemo
{
  public static void main (String[]args)
  {
    // Notice the use of generics. We will learn more about them later.
    // The type parameter <String> will ensure that your code cannot add any
    // other object than those of type String
    List < String > namesList = new ArrayList < String > ();
    // write your code below this
    for (int i = 0; i < args.length; i++)
      {

    namesList.add (args[i]);

      }

    System.out.println ("Before removing the element at index 2 the list is\n" +namesList);

    namesList.remove (2);

    System.out.println ("After removing the element at index 2 is the list is\n" +namesList);
  }
}

```

Q10. Create a class ListDemo with a main method. The method takes inputs from the command line arguments. Create an instance of **ArrayList** and add these inputs to the the list. Iterate over the **ArrayList** using an **Iterator**, and print the output as shown in the example.

Sample Input and Output:
Cmd Args : Rama Krishna Seetha Radha Lakshmi
Rama
Krishna
Seetha
Radha
Lakshmi

```java
package q24083;
import java.util.*;
public class ListDemo {
    public static void main(String[] args) {
        // Notice the use of generics. We will learn more about them later.
        // The type parameter <String> will ensure that your code cannot add any
        // other object than those of type String
        List<String> namesList = new ArrayList<String>();
        // write your code below this

        for(int i=0;i<args.length;i++){

                        namesList.add(args[i]);

        }

       Iterator itr = namesList.iterator();// Write your code here
        while (itr.hasNext()) {
            System.out.println(itr.next()); // print the list
        }

    }
}

```

Q11. Create a class ListDemo with a main method. The method takes inputs from the command line arguments. Create an instance of **ArrayList** and add these inputs to the the list.

Follow the given instructions and print the output as shown in the example

- add element **Green** at index **2**
- Print the resultant list after adding the element
- get the element at index **2** and print the same
- remove the element at index 2
- Print the resultant list after removing the element

Sample Input and Output:
Cmd Args : One Two Three Four
After adding the given element at index 2 the list becomes
[One, Two, Green, Three, Four]
The element at index 2 is Green
After removing the element at index 2 the list becomes
[One, Two, Three, Four]

```java
package q24084;
import java.util.*;
public class ListDemo
{
  public static void main (String[]args)
  {
    // Notice the use of generics. We will learn more about them later.
    // The type parameter <String> will ensure that your code cannot add any
    // other object than those of type String
    List < String > namesList = new ArrayList < String > ();
    // write your code below this
    // add the given elemnet at index 2

    // print the list
    // get the element at index 2
    // remove the element at index 2
    // print the list after removing the element

    for (int i = 0; i < args.length; i++)
      {

    namesList.add (args[i]);

      }

    namesList.add (2, "Green");

    System.out.println ("After adding the given element at index 2 the list becomes\n"+ namesList);


    System.out.println ("The element at index 2 is " + namesList.get (2));

    namesList.remove (2);

    System.out.println ("After removing the element at index 2 the list becomes\n" + namesList);

  }
}
```

Practice Programs on ArrayList

Q1. Fill the missing code in the below program to learn how to iterate over the elements stored in a ArrayList.

Write a Java program with a class name ArrayListIterationDemo with a main method. The method takes inputs from the command line arguments. If the first character of the argument is in upperrcase add it to the namesList and print all the elements in the list.

```
package q11955;
import java.util.*;
public class ArrayListIterationDemo
{
  public static void main (String[]args)
  {
    List < String > namesList = new ArrayList < String > ();

    for (int i = 0; i < args.length; i++)
      {
    if (args[i].charAt (0) >= 'A' && args[i].charAt (0) <= 'Z')
      {
        // add arguments to the namesList
        namesList.add (args[i]);

      }
      }

    for (Object name:namesList)
      {

    // print elements in the namesList

    System.out.println (name);

      }
  }
}

```

Q2. Fill the missing code in the below program to learn how to get elemetns stored in a ArrayList.

Write a Java program with a class name ArrayListIterationDemo with a main method. The method takes inputs from the command line arguments. Print the list of all the elements with their respective indices as shown in Sample Input and Output.

Sample Input and Output:
Cmd Args : Hyderabad Mumbai Karnataka Tamilnadu
Name at index 0 is : Hyderabad
Name at index 1 is : Mumbai
Name at index 2 is : Karnataka
Name at index 3 is : Tamilnadu

```
package q23673;
import java.util.*;
public class ArrayListIterationDemo
{
  public static void main (String[]args)
  {

    List < String > namesList = new ArrayList < String > ();

    for (int i = 0; i < args.length; i++)
      {

    namesList.add (args[i]);

      }

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

    System.out.println ("Name at index " + i + " is : " +
            namesList.get (i));

      }

  }

}

```

Q3. Fill the missing code in the below program to learn how to get elemetns stored in a ArrayList.

Write a Java program with a class name ArrayListIterationDemo with a main method. The method takes inputs from the command line arguments. Print the element at the index **2** using get(int index) method.

Sample Input and Output:
Cmd Args : Welcome to Hyderabad
Name at index 2 is : Hyderabad

```java
package q23676;
import java.util.*;
public class ArrayListIterationDemo
{
  public static void main (String[]args)
  {

    List < String > namesList = new ArrayList < String > ();

    for (int i = 0; i < args.length; i++)
      {

    namesList.add (args[i]);

      }

    System.out.println ("Name at index 2 is : " + namesList.get (2));




  }

}

```

Q5. Write a Java program with a class name ArrayListDemo with a main method. The method takes inputs from the command line arguments. Print the size of the list using the method size. Fill the missing code in the below program

Sample Input and Output:
Cmd Args : Ganga Godavari Krishna Narmada Sindu
[Ganga, Godavari, Krishna, Narmada, Sindu]
Size of the list is : 5

```java
package q24085;
import java.util.*;
public class ArrayListIterationDemo
{
  public static void main (String[]args)
  {
    List < String > namesList = new ArrayList < String > ();
    // write your code here

    for (int i = 0; i < args.length; i++)
      {

    namesList.add (args[i]);

      }

    System.out.println (namesList);

    System.out.println ("Size of the list is : " + namesList.size ());


  }

}

```