**How to compare two ArrayList in Java**

public static void main(String [] args)

{

ArrayList<String> al1= new ArrayList<String>();

al1.add("hi");

al1.add("How are you");

al1.add("Good Morning");

al1.add("bye");

al1.add("Good night");

ArrayList<String> al2= new ArrayList<String>();

al2.add("Howdy");

al2.add("Good Evening");

al2.add("bye");

al2.add("Good night");

//Storing the comparison output in ArrayList<String>

ArrayList<String> al3= new ArrayList<String>();

for (String temp : al1)

al3.add(al2.contains(temp) ? "Yes" : "No");

System.out.println(al3);

**}**

**swapping two elements in ArrayList**

public class SwappingExample {

public static void main(String a[]){

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

al.add("Sachin");

al.add("Rahul");

al.add("Saurav");

al.add("Sunil");

al.add("Kapil");

al.add("Vinod");

System.out.println("ArrayList before Swap:");

for(String temp: al){

System.out.println(temp);

}

//Swapping 2nd(index 1) element with the 5th(index 4) element

Collections.swap(al, 1, 4);

System.out.println("ArrayList after swap:");

for(String temp: al){

System.out.println(temp);

}

}

}

**How to serialize ArrayList in java**

[ArrayList](https://docs.oracle.com/javase/6/docs/api/java/util/ArrayList.html) is [serializable](https://docs.oracle.com/javase/6/docs/api/java/io/Serializable.html) by **default**. This means you need not to implement Serializable interface explicitly in order to serialize an ArrayList. In this tutorial we will learn how to serialize and de-serialize an ArrayList.

package beginnersbook.com;

import java.util.ArrayList;

import java.io.\*;

public class ArrayListSerialization

{

public static void main(String [] args)

{

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

al.add("Hello");

al.add("Hi");

al.add("Howdy");

try{

FileOutputStream fos= new FileOutputStream("myfile");

ObjectOutputStream oos= new ObjectOutputStream(fos);

oos.writeObject(al);

oos.close();

fos.close();

}catch(IOException ioe){

ioe.printStackTrace();

}

}

}

**De-Serialization**:

In this class we are retrieving the stream of bytes from myfile which we have stored using the above class. We are type casting the returned object to ArrayList and displaying the elements of ArrayList.

package beginnersbook.com;

import java.io.\*;

import java.util.ArrayList;

public class DeSerializationClass

{

public static void main(String [] args)

{

ArrayList<String> arraylist= new ArrayList<String>();

try

{

FileInputStream fis = new FileInputStream("myfile");

ObjectInputStream ois = new ObjectInputStream(fis);

arraylist = (ArrayList) ois.readObject();

ois.close();

fis.close();

}catch(IOException ioe){

ioe.printStackTrace();

return;

}catch(ClassNotFoundException c){

System.out.println("Class not found");

c.printStackTrace();

return;

}

for(String tmp: arraylist){

System.out.println(tmp);

}

}

}

**How to join/combine two ArrayLists in java**

import java.util.ArrayList;

public class Details

{

public static void main(String [] args)

{

//First ArrayList

ArrayList<String> arraylist1=new ArrayList<String>();

arraylist1.add("AL1: E1");

arraylist1.add("AL1: E2");

arraylist1.add("AL1: E3");

//Second ArrayList

ArrayList<String> arraylist2=new ArrayList<String>();

arraylist2.add("AL2: E1");

arraylist2.add("AL2: E2");

arraylist2.add("AL2: E3");

//New ArrayList

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

al.addAll(arraylist1);

al.addAll(arraylist2);

//Displaying elements of the joined ArrayList

for(String temp: al){

System.out.println(temp);

}

}

}

**How to clone an ArrayList to another ArrayList**

In this tutorial we will learn how to clone an ArrayList to another one. We would be using [clone() method](https://docs.oracle.com/javase/7/docs/api/java/util/ArrayList.html#clone()) of ArrayList class to serve our purpose.

import java.util.ArrayList;

public class Details {

public static void main(String a[]){

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

//Adding elements to the ArrayList

al.add("Apple");

al.add("Orange");

al.add("Mango");

al.add("Grapes");

System.out.println("ArrayList: "+al);

ArrayList<String> al2 = (ArrayList<String>)al.clone();

System.out.println("Shallow copy of ArrayList: "+ al2);

//add and remove on original ArrayList

al.add("Fig");

al.remove("Orange");

//Display of both ArrayLists after add & remove

System.out.println("Original ArrayList:"+al);

System.out.println("Cloned ArrayList:"+al2);

}

}

Output:

ArrayList: [Apple, Orange, Mango, Grapes]

Shallow copy of ArrayList: [Apple, Orange, Mango, Grapes]

Original ArrayList:[Apple, Mango, Grapes, Fig]

Cloned ArrayList:[Apple, Orange, Mango, Grapes]

[**Increase the capacity(size) of ArrayList**](https://beginnersbook.com/2013/12/java-arraylist-ensurecapacity-method-example/)

ArrayList internally implements growable dynamic array which means it can increase and decrease its size automatically. If we try to add an element to a already full ArrayList then it automatically re-sized internally to accommodate the new element however sometimes its not a good approach.

Consider a scenario when there is a need to add huge number of elements to an already full ArrayList, in such case ArrayList has to be resized several number of times which would result in a poor performance. For such scenarios ensureCapacity() method of Java.util.ArrayList class is very useful as it increases the size of the ArrayList by a specified capacity.

import java.util.ArrayList;

public class EnsureCapacityExample {

public static void main(String args[]) {

// ArrayList with Capacity 4

ArrayList<String> al = new ArrayList<String>(4);

//Added 4 elements

al.add("Hi");

al.add("Hello");

al.add("Bye");

al.add("GM");

//Increase capacity to 5

al.ensureCapacity(5);

al.add("GE");

// let us print all the elements available in list

for (String temp: al) {

System.out.println(temp);

}

}

}