



JAVA COLLECTIONS

ASSIGNMENT

AMIT DHAGE SIR

Ramavath Santhosh | 22MCF1R40 | 2nd Sem 1st year MCA

Q1. Convert ArrayList to String array

Method1 : Using ArrayList class get() method

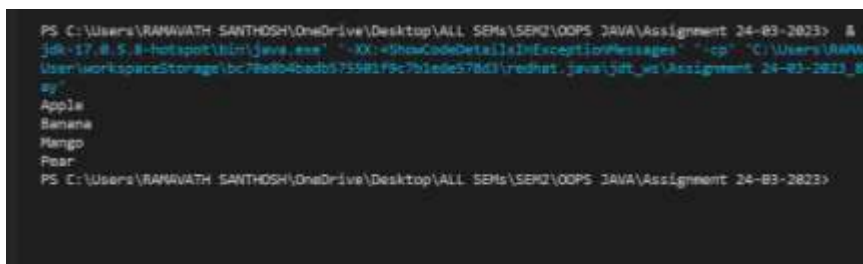
In this example we have converted the whole list to array in three steps

- Obtain the ArrayList size using size() method
- Fetch each element of the list using get() method
- Assigned each element to corresponding array element using assignment = operator
- Print String array

```
import java.util.*;
public class ConvertArrayListToArray {
    public static void main(String args[]) {
        // Creating and initializing ArrayList
        ArrayList<String> fruits = new ArrayList<>();
        fruits.add("Apple");
        fruits.add("Banana");
        fruits.add("Mango");
        fruits.add("Pear");

        // ArrayList to String array conversion
        String[] str = new String[fruits.size()];
        for(int i=0; i < fruits.size(); i++) {
            str[i] = fruits.get(i);
        }

        // Print elements using for-each loop
        for(String s : str) {
            System.out.println(s);
        }
    }
}
```



```
PS C:\Users\RAMWATH_SANTHOSH\OneDrive\Desktop\ALL_SEMs\SEM2\OOPS_JAVA\Assignment 24-03-2023> .\
jdk-17.0.5\bin\java.exe -XX:+ShowCodeDetailsInExceptionMessages -cp "C:\Users\RAMWATH_SANTHOSH\OneDrive\Desktop\ALL_SEMs\SEM2\OOPS_JAVA\Assignment 24-03-2023\ConvertArrayListToArray.class"
Apple
Banana
Mango
Pear
PS C:\Users\RAMWATH_SANTHOSH\OneDrive\Desktop\ALL_SEMs\SEM2\OOPS_JAVA\Assignment 24-03-2023>
```

Method 2 : Using ArrayList class toArray() method

In this example we will convert ArrayList to String array using toArray() method.

- Using toArray() method convert ArrayList to Object array.
- Iterate each element and convert them to the desired type using typecasting. Here we are converting to String type and added to the String array.
- Last step is to print the String array.

```
import java.util.*;

public class ConvertArrayListToArray2 {
    public static void main(String args[]) {
        // Instantiating and initializing ArrayList
        ArrayList<String> cities = new ArrayList<>();
        cities.add("Boston");
        cities.add("Dallas");
        cities.add("San jose");
        cities.add("Chicago");

        // ArrayList to String array conversion using toArray()
        String citinames[]=cities.toArray(new String[cities.size()]);

        // Printing elements using for-each loop
        for(String str : citinames) {
            System.out.println(str);
        }
    }
}
```



```
PS C:\Users\RAMAVATH SANTHOSH\OneDrive\Desktop\ALL SEMs\SEM2\OOPS JAVA\Assignment 24-03-2023> &
@i$.E-hotspot\bin\java.exe "-XX:+ShowCodeDetailsInExceptionMessages" -cp "C:\Users\RAMAVATH S
Storage\bc70a8b4badb575581f9c7b1ede578d3\redhat.java\jdk_ua\Assignment 24-03-2023_Ra8b5583\bin"
Boston
Dallas
San Jose
Chicago
PS C:\Users\RAMAVATH SANTHOSH\OneDrive\Desktop\ALL SEMs\SEM2\OOPS JAVA\Assignment 24-03-2023>
```

Method3 : Using Arrays class copyOf() method

In this example, we will convert the ArrayList to String array using Arrays class copyOf() method.

- First, just like above example convert the ArrayList to Object array using toArray() method.
- Use Arrays.copyOf() method to convert Object array to String array.
- Show the String array.

```
import java.util.*;

public class ConvertArrayListToArray3 {
    public static void main(String args[]) {
        // Declaring and initializing ArrayList in one step
        ArrayList<String> browsers = new ArrayList<>();
        browsers.add("Google Chrome");
        browsers.add("Mozilla Firefox");
        browsers.add("Edge");
        browsers.add("Opera");

        //Converting ArrayList to String array using copyOf()
        String[] browsenames = Arrays.copyOf(browsers.toArray(), browsers.size(), String[].class);

        // Displaying elements using for-each loop
        for(String str : browsenames) {
            System.out.println(str);
        }
    }
}
```

```
PS C:\Users\RAMAVATH SANTHOSH\OneDrive\Desktop\ALL SEMs\SEPM\OOPS JAVA\Assignment 24-03-2023> & "C:\Program
0.5.8-hotspot\bin\java.exe" "-XX:+ShowCodeDetailsInExceptionMessages" "-cp" "C:\Users\RAMAVATH SANTHOSH\AppData
Storage\Local78a8b4a6d571581f9c7b1ade578d3f\redhat_java\jdk_wa\Assignment_24-03-2023_RaBossS3\bin" "ConvertArra
Google Chrome
Mozilla Firefox
Edge
Opera
PS C:\Users\RAMAVATH SANTHOSH\OneDrive\Desktop\ALL SEMs\SEPM\OOPS JAVA\Assignment 24-03-2023>
```

Q2. Write a program to traverse (or iterate) ArrayList?

Traverse ArrayList using for loop, while loop, advance for loop and iterator. This question tests knowledge of add() method of ArrayList and looping concept. Below example traverses ArrayList using advance for loop:

```
import java.util.*;
public class ArrayListLoopExample {
    public static void main(String args[]) {
        // initialize ArrayList
        ArrayList<Integer> al = new ArrayList<Integer>();
        // add elements to ArrayList object
        al.add(3);
        al.add(17);
        al.add(6);
        al.add(9);
        al.add(7);
        System.out.println("Using Advanced For Loop");
        // printing ArrayList
        for (Integer num : al) {
            System.out.println(num);
        }
    }
}
```

```
PS C:\Users\RAMAVATH SANTHOSH\OneDrive\Desktop\ALL SEMs\SEPM\OOPS JAVA\Assignment 24-03-2023> & "C
0.5.8-hotspot\bin\java.exe" "-XX:+ShowCodeDetailsInExceptionMessages" "-cp" "C:\Users\RAMAVATH SANT
Storage\Local78a8b4a6d571581f9c7b1ade578d3f\redhat_java\jdk_wa\Assignment_24-03-2023_RaBossS3\bin" "Ar
Using Advanced For Loop
3
17
6
9
7
PS C:\Users\RAMAVATH SANTHOSH\OneDrive\Desktop\ALL SEMs\SEPM\OOPS JAVA\Assignment 24-03-2023>
```

Q3. Program for How to Iterate HashSet Using Iterator

```
import java.util.*;

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

        // Declaring a HashSet
        HashSet<String> hashset = new HashSet<String>();
        // Add elements to HashSet
        hashset.add("Pear");
        hashset.add("Apple");
        hashset.add("Orange");
        hashset.add("Papaya");
        hashset.add("Banana");
        // Get iterator
        Iterator<String> it = hashset.iterator();
        // Show HashSet elements
```

```

System.out.println("HashSet contains: ");
while(it.hasNext()) {
    System.out.println(it.next());
}
}
}

```



```

PS C:\Users\RAHWATH SAMTHOSH\OneDrive\Desktop\ALL SEMs\SEM2\OOPS JAVA\Assignment 24-03-2023> java -cp C:\Users\RAHWATH SAMTHOSH\OneDrive\Desktop\ALL SEMs\SEM2\OOPS JAVA\Assignment 24-03-2023\src\HashSetIteration.class
HashSet contains:
Apple
Pear
Papaya
Orange
Banana
PS C:\Users\RAHWATH SAMTHOSH\OneDrive\Desktop\ALL SEMs\SEM2\OOPS JAVA\Assignment 24-03-2023>

```

Q4. Program for How to Iterate HashSet using for each loop

```

import java.util.*;

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

        // Declaring a HashSet
        HashSet<String> hashset = new HashSet<String>();
        // Add elements to HashSet
        hashset.add("Pear");
        hashset.add("Apple");
        hashset.add("Orange");
        hashset.add("Papaya");
        hashset.add("Banana");

        System.out.println("HashSet contains :");
        // Using for each loop
        for(String str : hashset){
            System.out.println(str);
        }
    }
}

```



```

PS C:\Users\RAHWATH SAMTHOSH\OneDrive\Desktop\ALL SEMs\SEM2\OOPS JAVA\Assignment 24-03-2023> java -cp C:\Users\RAHWATH SAMTHOSH\OneDrive\Desktop\ALL SEMs\SEM2\OOPS JAVA\Assignment 24-03-2023\src\HashSetIteration.class
HashSet contains :
Apple
Pear
Papaya
Orange
Banana
PS C:\Users\RAHWATH SAMTHOSH\OneDrive\Desktop\ALL SEMs\SEM2\OOPS JAVA\Assignment 24-03-2023>

```

Q5. Given an element write a program to check if element(value) exists in ArrayList?

```

import java.util.*;

public class ArrayListContainsExample {
    public static void main(String args[]) {
        // initialize ArrayList
        ArrayList<Integer> al = new ArrayList<Integer>();
        // add elements to ArrayList object
        al.add(3);
    }
}

```

```

        al.add(17);
        al.add(6);
        al.add(9);
        al.add(7);
        // check if ArrayList contains element
        if (al.contains(7)) {
            System.out.println("7 was found in the list");
        } else {
            System.out.println("7 was not found in the list");
        }
    }
}

```



A screenshot of a Java program's output. The command prompt shows the file path and the execution of the program. The output line reads: "7 was found in the list".

Q6 Given an element write a program to check if element exists in HashSet?

```

import java.util.*;
public class HashSetContainsExample {
    public static void main(String args[]) {
        // initialize HashSet
        HashSet<Integer> hs = new HashSet<Integer>();
        // add elements to HashSet object
        hs.add(3);
        hs.add(17);
        hs.add(6);
        hs.add(9);
        hs.add(7);
        // check if HashSet contains element
        if (hs.contains(7)) {
            System.out.println("7 was found in the list");
        } else {
            System.out.println("7 was not found in the list");
        }
    }
}

```



A screenshot of a Java program's output. The command prompt shows the file path and the execution of the program. The output line reads: "7 was found in the list".

Q7. Convert an Array to ArrayList in Java

Method 1 : Using Arrays.asList() method

In this example, we are converting an Array to ArrayList using Arrays class asList() method. The syntax is given below:

Syntax :
 ArrayList<String> list = new ArrayList<>(Arrays.asList(arrayname));

```

import java.util.*;
public class ConvertArrayToArrayList {
    public static void main(String args[]) {
        // Declaring and initializing Array
        String[] cities={"Boston", "Dallas", "New York", "Chicago"};

        //Converting Array to ArrayList using Arrays.asList()
        ArrayList<String> list= new ArrayList<>(Arrays.asList(cities));

        // Add more elements to the converted list
        list.add("San Francisco");
        list.add("San jose");

        // Print arraylist elements using for-each loop
        for(String s : list) {
            System.out.println(s);
        }
    }
}

```



```

PS C:\Users\ANANDH\OneDrive\Desktop\ALL SEMS\SEM5\OOPS JAVA\Assignment 24-03-2023> java -cp C:\Users\ANANDH\OneDrive\Desktop\ALL SEMS\SEM5\OOPS JAVA\Assignment 24-03-2023\src\ConvertArrayToArrayList ConvertArrayToArrayList
Boston
Dallas
New York
Chicago
San Francisco
San Jose

```

Method 2 : Using Collections.addAll() method

This method does the same as Arrays.asList() method however it is much faster hence performance wise this is the best way to convert Array to ArrayList. The syntax for the addAll() method is given below:

Syntax:

Collections.addAll(arraylist, array);

```

import java.util.*;
public class ConvertArrayToArrayList2 {
    public static void main(String args[]) {
        // Creating and initializing Array
        String[] strArray = {"AAA", "BBB", "CCC", "DDD"};

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

        //Converting Array to ArrayList using addAll() method
        Collections.addAll(al, strArray);

        // Add more elements to the converted list
        al.add("YYY");
        al.add("ZZZ");

        // Displaying arraylist elements using for-each loop
        for(String s : al) {
            System.out.println(s);
        }
    }
}

```

```
PS C:\Users\JRWANATH> SAATCHI\Developer\Desktop\ALL_SOPS\SOPS\SOPS -ShowMsgName
C:\Users\JRWANATH> "cp" "C:\Users\JRWANATH\SAATCHI\Developer\Desktop\
2019_Rationalize\bin" "C:\Users\JRWANATH\Desktop\2019\
All
RBI
CCC
ODD
YYY
ZZZ
PS C:\Users\JRWANATH> SAATCHI\Developer\Desktop\ALL_SOPS\SOPS\SOPS -ShowMsgName
```

Method 3 : Using add() method

If we don't want to use java built-in methods then we can use add() method to convert an array to ArrayList. This is a manual way of adding all the array elements to the ArrayList as shown below:

```
import java.util.*;

public class ConvertArrayToArrayList3 {
    public static void main(String args[]) {
        // Declaring and instantiating ArrayList in one step
        ArrayList<String> al = new ArrayList();

        // Given initialized array
        String[] strArray = {"Cocacola", "Pepsi", "Fanta", "Dr Pepper"};

        //Converting Array to ArrayList manually
        for (int i=0; i < strArray.length ; i++) {
            // Adding every element of array to the ArrayList
            al.add(strArray[i]);
        }

        // Showing arraylist elements using for-each loop
        for(String str1 : al) {
            System.out.println(str1);
        }
    }
}
```

```
PS C:\Users\UWWATH\SARITHORA\Desktop\ALL_SOPS\USP2\COOPS_3WAY\Assignment-24-03-2023>
java -cp "X:\JDKCode\src\bin\acceptcookies.jar;" -D"C:\Users\UWWATH\SARITHORA\Desktop\
AllSops\Vendor\Java\jdk-as\Assignment-24-03-2023\_saathira@jain"> ConvertArrayToMaya13x73
Cocacola
Peppi
Fanta
Dr Pepper
PS C:\Users\UWWATH\SARITHORA\Desktop\ALL_SOPS\USP2\COOPS_3WAY\Assignment-24-03-2023>
```

Q8. ArrayList size() example in Java

When you create an Object of an ArrayList, it is created as empty ArrayList and its size() will be zero. If you add elements one by one then size grows one by one.

1. Java Program to find Length/Size of Integer ArrayList

```
import java.util.*;
import java.io.*;

/* Write a program to determine the size/length of the ArrayList*/
public class ArrayListSize
{
    public static void main (String[] args)
    {
        // Create an Integer ArrayList Object
        ArrayList<Integer> arrlist=new ArrayList<Integer>();
```



```

// Print initial size of ArrayList
System.out.println("Size before adding elements: "+arrlist.size());

// Adding elements to ArrayList Object
arrlist.add(11);
arrlist.add(3);
arrlist.add(5);
arrlist.add(4);
arrlist.add(9);


/* Print size of ArrayList
   after adding elements */
System.out.println("Size after adding elements: "+arrlist.size());

// Removing elements from ArrayList
arrlist.remove(1);
arrlist.remove(2);

/* Print size of ArrayList
   after removing elements */
System.out.println("Size after removing elements: "+arrlist.size());

// Print ArrayList
System.out.println("Resulting ArrayList: ");
for(int num: arrlist){
    System.out.println(num);
}
}
}

```



```

PS C:\Users\ANWATH\Documents\Desktop\ALL_SEPS\SEPS\OOPS_JAVA\Assignment_
\Ques_09> .\Q9.java
Size before adding elements: 0
Size after adding elements: 5
Size after removing elements: 3
Resulting ArrayList:
11
3
4
PS C:\Users\ANWATH\Documents\Desktop\ALL_SEPS\SEPS\OOPS_JAVA\Assignment_

```

Q9. Java Program to find Length/Size of String ArrayList

```

import java.util.*;
import java.io.*;

/* Program to find size of ArrayList in Java */
public class ArrayListSize
{
    public static void main (String[] args)
    {
        System.out.println("Java Program to find the size of ArrayList");

        // Create an String ArrayList Object
        ArrayList<String> listOfCities = new ArrayList<>();

        int size = listOfCities.size();
        // Print initial size of ArrayList
        System.out.println("size of ArrayList after creation: " + size);

        // Adding elements to ArrayList Object
        listOfCities.add("California");
        listOfCities.add("Boston");
    }
}

```

```

listOfCities.add("New York");

size = listOfCities.size();
/* Print size of ArrayList
after adding elements */
System.out.println("size of ArrayList after adding elements: " + size);

// clear() method removes all elements
listOfCities.clear();

size = listOfCities.size();
System.out.println("size of ArrayList after clearing elements: " + size);
}
}

```

Q10. Write a program to add elements to HashSet?

```

import java.util.*;
public class HashSetAddExample {
    public static void main(String args[]) {
        // initialize HashSet
        HashSet<Integer> hs = new HashSet<Integer>();
        // add elements to HashSet object
        hs.add(3);
        hs.add(17);
        hs.add(6);
        hs.add(9);
        hs.add(7);
        System.out.println("Using Advanced For Loop");
        // printing HashSet
        for (Integer num : hs) {
            System.out.println(num);
        }
    }
}

```



Q11. Get Size of HashMap Example

HashMap with Integer keys and String values

```

import java.util.*;
public class HashMapSizeExample {
    public static void main(String args[]) {

        // Creating HashMap object with Integer keys and String values
        HashMap<Integer,String> map = new HashMap<>();

        // Adding elements to the HashMap object
    }
}

```

```

        map.put(1, "CocoCola");
        map.put(2, "Pepsi");
        map.put(3, "Thums Up");
        map.put(4, "Fanta");

        // Calculating the size of the HashMap using size() method
        System.out.println(" Size of the given HashMap is: "+ map.size());
    }
}

```

Q12. Program for How to Check if a HashMap is Empty or Not

```

import java.util.HashMap;

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

        // Creating HashMap object with Integer keys and String values
        HashMap<Integer, String> map = new HashMap<>();

        // Checking whether HashMap is empty or not
        System.out.println("Checking Is HashMap empty?: " + map.isEmpty());

        // Adding elements to the HashMap object
        map.put(100, "Jack");
        map.put(200, "John");
        map.put(300, "Smith");

        // Checking again whether HashMap is empty or not
        System.out.println("Checking Is HashMap empty?: "+ map.isEmpty());
    }
}

```

Output:

```

Checking Is HashMap empty?: true
Checking Is HashMap empty?: false

```

Using size() method

I have already explained the size() method . We can easily check if HashMap is empty or not by using size() method. If size() method returns 0 then the HashMap is empty otherwise not.

```

import java.util.*;

public class HashMapEmptyExample2 {
    public static void main(String args[]) {

        // Creating HashMap object with String keys and String values
        HashMap<String, String> map = new HashMap<>();

        // Checking whether HashMap is empty or not using size() method
        System.out.println("Checking Is HashMap empty using size() method?: " + (map.size()==0));
    }
}

```

```

// Putting elements to the HashMap object
map.put("100", "Java");
map.put("1000", "Python");
map.put("10000", "Javascript");

// Checking again whether HashMap is empty or not using size() method
System.out.println("Checking Is HashMap empty using size() method?: "+ (map.size()==0));
}
}

```



```

PS C:\Users\RAWWATH SANHOS\OneDrive\Desktop\ALL SDPs\SDP2\OOPS JAVA\Assignment 24-03-2023>
.\java.exe -Xrs:showcodeDetails:injectionMessages -cp 'C:\Users\RAWWATH SANHOS\AppData
\Local\Temp\java\jdk-9s\Assignment_24-03-2023\HashMap\src' 'HashMapEmptyExample2'
Checking Is HashMap empty using size() method?: true
Checking Is HashMap empty using size() method?: false
PS C:\Users\RAWWATH SANHOS\OneDrive\Desktop\ALL SDPs\SDP2\OOPS JAVA\Assignment 24-03-2023>

```

Q13. Program for Sorting HashMap by Keys

```

import java.util.*;

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

        // Creating a HashMap of int keys and String values
        HashMap<Integer, String> hashmap = new HashMap<Integer, String>();

        // Adding Key and Value pairs to HashMap
        hashmap.put(22,"A");
        hashmap.put(55,"B");
        hashmap.put(33,"Z");
        hashmap.put(44,"M");
        hashmap.put(99,"I");
        hashmap.put(88,"X");

        System.out.println("Before Sorting:");
        Set set = hashmap.entrySet();
        Iterator iterator = set.iterator();
        while(iterator.hasNext()) {
            Map.Entry pair = (Map.Entry)iterator.next();
            System.out.print(pair.getKey() + ": ");
            System.out.println(pair.getValue());
        }
        Map<Integer, String> map = new TreeMap<Integer, String>(hashmap);    System.out.println("After
Sorting:");
        Set set2 = map.entrySet();
        Iterator iterator2 = set2.iterator();
        while(iterator2.hasNext()) {
            Map.Entry pair = (Map.Entry)iterator2.next();
            System.out.print(pair.getKey() + ": ");
            System.out.println(pair.getValue());
        }
    }
}

```

```

Before Sorting:
33: Z
99: I
22: A
55: B
88: X
44: M
After Sorting:
22: A
33: Z
44: M
55: B
88: X
99: I
PS C:\Users\WAWA\OneDrive\Desktop\All_Soft\

```

HashMap Sorting by Values

In this example we are sorting HashMap by values using Comparator.

Program for Sorting HashMap by Values

```

import java.util.*;
public class HashMapSortByValueExample {
    public static void main(String args[]) {

        // Creating a HashMap of int keys and String values
        HashMap<Integer, String> hashmap = new HashMap<Integer, String>();

        // Adding Key and Value pairs to HashMap
        hashmap.put(22,"A");
        hashmap.put(55,"B");
        hashmap.put(33,"Z");
        hashmap.put(44,"M");
        hashmap.put(99,"I");
        hashmap.put(88,"X");

        System.out.println("Before Sorting:");
        Set set = hashmap.entrySet();
        Iterator iterator = set.iterator();
        while(iterator.hasNext()) {
            Map.Entry pair = (Map.Entry)iterator.next();
            System.out.print(pair.getKey() + ": ");
            System.out.println(pair.getValue());
        }
        Map<Integer, String> map = sortByValues(hashmap);
        System.out.println("After Sorting:");
        Set set2 = map.entrySet();
        Iterator iterator2 = set2.iterator();
        while(iterator2.hasNext()) {
            Map.Entry pair = (Map.Entry)iterator2.next();
            System.out.print(pair.getKey() + ": ");
            System.out.println(pair.getValue());
        }
    }

    private static HashMap sortByValues(HashMap map) {
        List list = new LinkedList(map.entrySet());
        // Defined Custom Comparator here
        Collections.sort(list, new Comparator() {
            public int compare(Object o1, Object o2) {

```



```

    public int getStudentage() {
return studentage;
    }
    public void setStudentage(int studentage) {
this.studentage = studentage;
    }
}

```

We will add Students to the ArrayList object.

```

import java.util.*;

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

        ArrayList<Student> arraylist = new ArrayList<Student>();
        arraylist.add(new Student(111, "John", 23));
        arraylist.add(new Student(222, "Messi", 29));
        arraylist.add(new Student(333, "Ronaldo", 31));

        Collections.sort(arraylist);

        for(Student str: arraylist){
            System.out.println(str);
        }
    }
}

```

We tried to call the Collections.sort() method on List of Objects and got the following error message.

Exception in thread "main" java.lang.Error: Unresolved compilation problem:
 Bound mismatch: The generic method sort(List) of type Collections is not applicable for the arguments
 (ArrayList). The inferred type Student is not a valid substitute for the bounded parameter > at
 ArrayListSort.main(ArrayListSort.java:15)
 Reason : I just called the sort method on an ArrayList of Objects which doesn't work until unless we use
 interfaces like Comparable and Comparator. Now we will use Comparable and Comparator to get the sorting
 done in our way.

Q15 Sorting of ArrayList(Object) Using Comparable

Suppose we need to sort the ArrayList object based on student Age property. To achieve this we will first implement Comparable interface and then override the compareTo() method.

```

public class Student implements Comparable {
    private String studentname;
    private int rollno;
    private int studentage;

    public Student(int rollno, String studentname, int studentage) {
        this.rollno = rollno;
        this.studentname = studentname;
        this.studentage = studentage;
    }
}

```

```

...
//getter and setter methods same as the above example
...
@Override
public int compareTo(Student comparestu) {
    int compareage=((Student)comparestu).getStudentage();
    /* For Ascending order*/
    return this.studentage-compareage;

    /* For Descending order do like this */
    //return compareage-this.studentage;
}

@Override
public String toString() {
    return "[ rollno=" + rollno + ", name=" + studentname + ", age=" + studentage + " ]";
}
}

```

Now we can call Collections.sort() on ArrayList

```

import java.util.*;

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

        ArrayList<Student> arraylist = new ArrayList<Student>();
        arraylist.add(new Student(222, "Messi", 29));
        arraylist.add(new Student(333, "Ronaldo", 31));
        arraylist.add(new Student(111, "john", 23));

        Collections.sort(arraylist);

        for(Student str: arraylist){
            System.out.println(str);
        }
    }
}

```

Why do we need Comparator when we Already have Comparable

If you want to have more than one way of sorting your class, you must implement Comparator.

Q16 Sorting ArrayList(Object) Multiple Properties Using Comparator

To implement Comparator we need to override compare method for sorting.

```

import java.util.Comparator;
public class Student {
    private String studentname;
    private int rollno;
    private int studentage;

    public Student(int rollno, String studentname, int studentage) {

```



```

        this.rollno = rollno;
        this.studentname = studentname;
        this.studentage = studentage;
    }
    ...
    //Getter and setter methods same as the above examples
    ...
    /*Comparator for sorting the list by Student Name*/
    public static Comparator<Student> StuNameComparator = new Comparator<Student>() {

    public int compare(Student s1, Student s2) {
        String StudentName1 = s1.getStudentname().toUpperCase();
        String StudentName2 = s2.getStudentname().toUpperCase();

        //ascending order
        return StudentName1.compareTo(StudentName2);

        //descending order
        //return StudentName2.compareTo(StudentName1);
    }
    };

    /*Comparator for sorting the list by roll no*/
    public static Comparator<Student> StuRollno = new Comparator<Student>() {

    public int compare(Student s1, Student s2) {

        int rollno1 = s1.getRollno();
        int rollno2 = s2.getRollno();

        /*For ascending order*/
        return rollno1-rollno2;

        /*For descending order*/
        //rollno2-rollno1;
    }
    };
    @Override
    public String toString() {
        return "[ rollno=" + rollno + ", name=" + studentname + ", age=" + studentage + "]";
    }
}

```

Q17 ArrayList Class

```

import java.util.*;
public class ArrayListSort {

    public static void main(String args[]){
        ArrayList<Student> arraylist = new ArrayList<Student>();
        arraylist.add(new Student(111, "John", 30));
        arraylist.add(new Student(333, "Ronaldo", 31));
        arraylist.add(new Student(222, "Messi", 29));

        /*Sorting based on Student Name*/
        System.out.println("Student Name Sorting:");
        Collections.sort(arraylist, Student.StuNameComparator);
    }
}

```

```
for(Student str: arraylist){
System.out.println(str);
}

/* Sorting on Rollno property*/
System.out.println("RollNum Sorting:");
Collections.sort(arraylist, Student.StuRollno);
for(Student str: arraylist){
System.out.println(str);
}
}
}
```