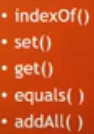
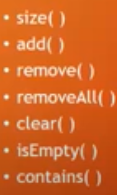
**ArrayList Coding** 



1. How to print ArrayList Elements ---------3 ways
2. Method used in Array List **a) Add (), b) remove(), and removeAll() methods**

import java.util.ArrayList;

import java.util.Iterator;

public class ArrayList1 {

public static void main(String[] args) {

ArrayList<Integer> number = new ArrayList<>();

System.out.println("Size = "+number.size()); //use size() to know the ArrayList size.

//Adding Elements

number.add(10); // Index [0]

number.add(11);

number.add(12);

number.add(20);

//Adding Another method --number.add(0, Integer.SIZE);

number.add(2, 50); //Index, Value.size

//Printing ways( 1)--using syso

System.out.println(number);------------------1)

// For size

System.out.println("Size = " + number.size());

//Printing ways(2) - using for each loop----------------------2)

for(int result: number) {

System.out.print(" " +result+ "");

}

System.out.println();

//Printing ways(3) - using for Iterator --------------------------3)

Iterator itr = number.iterator();

while (itr.hasNext()){

System.out.print(" "+ itr.next()); // print itr value one by one

}}} \*/

\* How to add ArrayList Elements -

Step 1 - type of ArrayList name then dot then take methods which you need like add methods

here Integer type ArrayList so we have to add

Step 2- Printing ways --1) Using Syso

2) for each loop

3) Iterator class's method

Step 3 - How to use 3) Iterator class's method to take print

a) take a Iterator class variable

b) then = sign and then ArrayList name then dot

c) use iterator() method.

So this iterator methods will store number ArrayList's value in the itr variable

d) using while loop. [ itr modde jotokhnon value thakbe totokhon hasNext() use korbo.

itr jotokhon empty hobe na totokhon Loop cholte thakbe] ie itr will check whether there are any

elements or not .\*/

**package** arraylistdemo;

**import** java.util.ArrayList;

**import** java.util.Iterator;

**public** **class** ArrayListDemoPrint {

**public** **static** **void** main(String[] args) {

ArrayList<Integer> number = **new** ArrayList<>();

//System.out.println("Size = "+number.size()); //use size() to know the ArrayList size.

//Adding Elements

number.add(10); // Index [0]

number.add(11);

number.add(22);

number.add(20);

number.add(3, 50); //Adding Another method --number.add(0, Integer.SIZE); //Index, Value.size// For size

System.***out***.println("Before removing ArrayList Contains"+number);

System.***out***.println();

System.***out***.println();

// For size

System.***out***.println("Size = " + number.size());

//Removing Elements

number.remove(2); // remove from which index . like we want index 2 -value remove then again ArrayList print

System.***out***.println("After removing ArrayList Contains "+number);

number.removeAll(number);

System.***out***.println("After removing all ArrayList Contains "+number);}}

/\*Result--

\* Before removing ArrayList Contains[10, 11, 22, 50, 20]

Size = 5

After removing ArrayList Contains [10, 11, 50, 20]

After removing all ArrayList Contains []\*/

**package** arraylistdemo;

**import** java.util.ArrayList;

**public** **class** ArrayListDemo3 {

**public** **static** **void** main(String[] args) {

ArrayList<Integer> number = **new** ArrayList<>();

//System.out.println("Size = "+number.size()); //use size() to know the ArrayList size.

//Adding Elements

number.add(10); // Index [0]

number.add(11);

number.add(22);

number.add(20);

number.add(3, 50); //Adding Another method --number.add(0, Integer.SIZE); //Index, Value.size// For size

System.***out***.println("Before removing ArrayList Contains"+number);

System.***out***.println();

System.***out***.println();

// For size

System.***out***.println("Size = " + number.size());

// Clear ()

/\*

\* number.clear(); System.out.println("After clear ArrayList Contains"+number);

\*/

// ArrayList Empty or Not . if it is empty then return true otherwise false

**boolean** result =number.isEmpty();

System.***out***.println(" "+"ArrayList is Empty : "+ result );

// Contains()- your require elements contains in ArrayList or notif it is empty then return true otherwise false.

**boolean** check = number.contains(20);

System.***out***.println("The number 20 is in the lists : "+ check );

// to know the value of Index number

**int** position = number.indexOf(11);

System.***out***.println("The index of 11 is : "+position);

// set(). to set the index value . here 2 parameter. which position , and who is the value.we can replace the value in the specific index

number.set(3, 40);

System.***out***.println("After setting :"+ number);

// using get() , we can get the value

**int** x = number.get(3); // 2nd index number we want to see,

System.***out***.println("Index 3 = "+ x);

}}

**package** arraylistdemo;

**import** java.util.ArrayList;

**public** **class** ArrayListMethod4 {

**public** **static** **void** main(String[] args) {

// Create 3 ArrayList

ArrayList<Integer> num1 = **new** ArrayList<>();

ArrayList<Integer> num2 = **new** ArrayList<>();

ArrayList<Integer> num3 = **new** ArrayList<>();

// Value Assign

//adding some value with num1 ArrayList

num1 .add(14);

num1 .add(24);

num1 .add(34);

num1 .add(44);

System.***out***.println("ArrayList number1 :"+num1);

num2 .add(10);

num2 .add(20);

num2 .add(30);

num2 .add(40);

System.***out***.println("ArrayList number2 :"+num2);

num3.addAll(num2); // Add value of num2 with num3

System.***out***.println("ArrayList number3 = "+num3);

// use equals() method to check the equality with 2 arrayList

**boolean** check = num1.equals(num2);

System.***out***.println("num1 == num2 :"+check);

}}

/\*

\* output

\*

\* ArrayList number1 :[14, 24, 34, 44]

ArrayList number2 :[10, 20, 30, 40]

ArrayList number3 = [10, 20, 30, 40]

num1 == num2 :false

\*/

/\*

with addAll() we can add another arrayList name

\*/

**package** arraylistdemo;

**import** java.util.ArrayList;

**import** java.util.Collections;

**public** **class** ArrayListSorting {

**public** **static** **void** main(String[] args) {

ArrayList<Integer> num1 = **new** ArrayList<>();

num1 .add(-14);

num1 .add(24);

num1 .add(4);

num1 .add(44);

num1 .add(-3);

num1 .add(40);

System.***out***.println("Before sorting : "+ num1);

// Ascending . Collections class - sort()

Collections.*sort*(num1);

System.***out***.println("After sorting in ascending order : "+ num1);

Collections.*sort*(num1,Collections.*reverseOrder*());

System.***out***.println("After sorting in descending order : "+ num1);

}}

/\*

Output

Before sorting : [-14, 24, 4, 44, -3, 40]

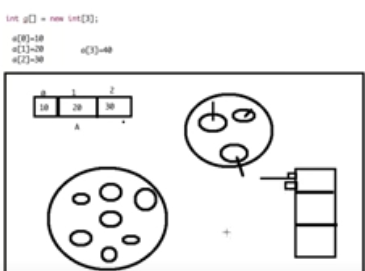
After sorting in ascending order : [-14, -3, 4, 24, 40, 44]

After sorting in descending order : [44, 40, 24, 4, -3, -14]

\*/---------------------------

ArrayList

Naveen Automation



**package** NaveenCollections;

**import** java.util.ArrayList;

**import** java.util.Iterator;

**public** **class** ArrayListGenericObject {

**public** **static** <E> **void** main(String[] args) {

// Generic ArrayList Object

//Here I want to store only Integer value in this particular arrayList.

ArrayList <Integer> ar1 = **new** ArrayList<Integer>();

ar1.add(100);

//ar1.add("Test"); // it will only allow the integer value. it will give you error if you want to store string type object because

System.***out***.println(ar1);

ArrayList <String> ar2 = **new** ArrayList <String>();

//array1.add(100); // it will only allow the string value.

ar2.add("They are always good"); // it will only add only string value.

ar2.add("I am good");

System.***out***.println(ar2);

System.***out***.println();

ArrayList <E> ar3 = **new** ArrayList<E>();

//Employee class object

Employee emp1 = **new** Employee("Rabiul", 30, "Business");

Employee emp2 = **new** Employee("Rashidul", 20,"QA");

Employee emp3 = **new** Employee("Ramiz", 40, "Developer");

ArrayList<Employee> ar4 = **new** ArrayList<Employee>(); //Employee Object it can store

ar4.add(emp1);

ar4.add(emp2);

ar4.add(emp2);

//Iterator to traverse the values

Iterator <Employee> itr1 = ar4.iterator();

// I have to traverse now

**while**(itr1.hasNext()) {

Employee employee = itr1.next();

System.***out***.println(employee.name);

System.***out***.println(employee.age);

System.***out***.println(employee.dept);

}

System.***out***.println("\*\*\*\*\*\*\*\*\*\*");

//==================================== Other Method --

// addAll(); Two object I have created ar5 and ar6.

ArrayList <String> ar5 = **new** ArrayList <String>();

ar5.add("Test");

ar5.add("Selenium");

ar5.add("QTP");

ArrayList <String> ar6 = **new** ArrayList <String>();

ar6.add("dev");

ar6.add("java");

ar6.add("QFT");

// Now I want to perform addAll(); how to addAll ()-- adding 2nd list into the First list object.

ar5.addAll(ar6);

// for Printing

**for**(**int** i = 0; i<ar5.size();i++) {

System.***out***.println(ar5.get(i));

}

System.***out***.println("-------------------------");

// Removing ar6 from ar5

ar5.removeAll(ar6);

**for**(**int** i = 0; i<ar5.size();i++) {

System.***out***.println(ar5.get(i));

}

System.***out***.println();

// retainAll()

ArrayList <String> ar7 = **new** ArrayList <String>();

ar7.add("Test");

ar7.add("Selenium");

ar7.add("QTP");

ArrayList <String> ar8 = **new** ArrayList <String>();

ar8.add("Test");

ar8.add("java");

ar7.retainAll(ar8);

**for** (**int** i = 0; i <ar7.size(); i++) {

System.***out***.println(ar7.get(i));

}

System.***out***.println("============-------");

//------------------------

//removeAll()

ar5.remove(ar6);

System.***out***.println("There is no value here -removing all ");

}}

---------------------------

**/\* 1) Non Generic /Generic in java --**

**\* Before JDK 1.5 Generic was not available.It is available after JDK 1.5. Now currently JDK 14 is available which is released March 17, 2020.**

**\* If you see the yellow line, you can see the warning line is same and if you mouse over and see that is the**

**|ArrayList is a raw type. References to generic type ArrayList<E> should be parameterized|**

**\*\*ArrayList is a raw type.-- it means that You have defined an ArrayList but What kind of values you are inserting, there is no definition for that.**

**\*\*\* References to generic type ArrayList<E> should be parameterized--This defined the JDK will say that java will give you one suggestion that**

**Please define the generic for this particular ArrayList, What kind of values you want to store in this particular arrayList ob-**

**So I can add any kind of value or any type of value**

**---So we are not defining what kind of Generic, what kind of types of data you want to store. So I can store different type of values**

**either it can be primitive data type or not primitive data type. So this is called non generic ArrayList Object. because we haven't define anything over here.**

**Q -How to define the Generic?**

**In Generic we can define all the primitive data type, non Primitive data type like String also,and some class object also.**

**To Define the generic**

**Here I want to store only Integer value in this particular arrayList.**

**So How to do this ?**

**a) After ArrayList <> angular bracket I have to used . this is called Generic. here Wrapper class i have to used in the left and right right.**

**b) Now see the yellow line gone. there is no warning. and this ArrayList will allow only integer value .**

**c) it will only allow the integer value. it will give you error if you want to store string type object because array is defined with integer.**

**It means I have defined the generic . This concept is called generic in java.**

**Before jdk 1.5 java collections framework generic was not supported . JAva generic allows to have at a time in this particular object we can store different**

**data type also.As well as the Specific data type also.**

**\*\*\*We can define the generic for anything. like String, boolean, double value.**

**\*\*\* In String ArrayList if you try to add some double value, integer value it will not allow it .This is called Generic concept in java.**

**>> In the String ArrayList If you want to store some specific information and all the info in the form of String values,**

**so it's a good practice to define the generic.**

**>> Question ---**

**Exception -Some time I am creating an ArrayList Object and I am not sure what kind of data it will be, it can be Integer ,or it can be String also**

**I am sure about it what kind of data at the run it will be produced and I have to store in some string in some ArrayList Variable then How to do that ?**

**Ans - In that case we just pass e. Like ArrayList <E> ar2 = new ArrayList<e>();**

**Here I am calling a static method that's why it is showing looks error. Add type Parameter <E> to main(String [])**

**But Generally we don't use.**

**Some time what happens that inside this ArrayList object I can Store different class object also/ user define class object also**

**in java arrayList we can store. So How to do that ?**

**Ans - Inside the same package I will create a very simple class let's see Employee class I will create without main method.**

**For Employee I have couple of information that Let's see String name; String dept; int age. Three diff values we have for this**

**particular employee. and I will create a Constructor of this particular Employee class and I will define all these three variables**

**initialize while creating the object.**

**So whenever I am creating this Employee class object I will pass those values. Those values will be given to this particular name .**

**So this name and global name is different .How will initialize these three class variable . using this keyword.**

**So in ArrayListGenericObject class I will create --**

**-->three employee class object I will be creating over here.**

**-->then create a ArrayList**

**---> Then I have to define the Generic. So to define the particular generic I want to store all the three Employee object.**

**So I have to make this particular ArrayList as Employee type. It can Store Employee Object.**

**so Employee class name / Employee object I have to store.**

**---> Using Iterator to traverse the value . create Iterator employee object and I have to traverse now.**

**----> then Print**

**\*\*\* If you write integer in this particular `ar1 , So I can store only Integer. and If I write String then I can store only String Values,**

**\*But If you write ArrayList<Employee>, it means I can store Employee object in this particular ar4 object.**

**\***

**--->>Using Iterator to traverse the value -- If you want to use all the values of this particular ArrayList from the particular ar4.**

**Now I will be using Iterator for that. I will store in this particular**

**---->>> How to traverse ? Simple using While loop. I can not for loop , because Iterator does not work on the basis of indexes.**

**This is not Index. This is Kind of object . So Employee object I have to store.**

**--->> In this particular Iterator if the next values is available, So what will happen?**

**---itr1.next();- in itr1.next(); it will give you the first particular emp1 because ar4 we have implemented Iterator.**

**Now all these emp1, emp2 and emp3 will be stored in this particular iterator.Now the moment you use itr1.next(), it will return the first**

**value ie Employee type -emp1 .**

**Question --How you store the specific user define class object in the Particular ArrayList ?**

**Ans -- in short**

**1) Create a user define class in the same package**

**2)Create a simple constructor with the (three) variable**

**3) back to ArrayList object class and Store the values like this ---**

**ArrayList<Employee> ar4 = new ArrayList<Employee>(); //Employee Object it can store**

**ar4.add(emp1);**

**ar4.add(emp2);**

**ar4.add(emp2);**

**4) using the Iterator we have to print all the values .So we have to iterate. We have to traverse all the values from**

**this three different object. like this---**

**//Iterator to traverse the values**

**Iterator <Employee> itr1 = ar4.iterator();**

**// I have to traverse now**

**while(itr1.hasNext()) {**

**Employee employee = itr1.next();**

**System.out.println(employee.name);**

**System.out.println(employee.age);**

**System.out.println(employee.dept);**

**}**

**==================================**

**addAll();**

**Step --**

**1)Two object I have created ar5 and ar6.**

**2)Inside this particular ar5 all ar6 values will be stored.**

**3) for Printing use for loop.**

**4) Performing the action everything on ar5, What ar5's addAll() method will do ? addAll() method merge two arrayList together.**

**it means all the ar6 object/Elememt will be added/ stored inside the ar5.**

**5) So Early ar5 has three values and now ar5 has 6 values using addAll().**

**// ----------------------------------**

**\* removeAll() or**

**\* removeAll ar6 values from ar5. -using for loop as same before.**

**\***

**\* //------------------**

**\* retained --**

**\* If you want to check the intersection part/ Common part that is called retainAll().**

**\***

**\* Step --**

**\* 1) Create again two object or like I have created here ar5 and ar6**

**\* 2) find the common value in both the ArrayList object. So Test is the common value**

**\* 3) use retainAll() method.**

**\* 4) use for loop for printing**

**\* 5) here 'Test' is common .So Test will be printed over here.**

**\* =======================================**

**\* Question**

**\* 1) What is Generic and Non Generic ArrayList?**

**\* 2) Different Characteristics of ArrayList?**

**\* 3) How to define the String type arrayList?**

**\* 4) How to define the Primitive/Integer type ArrayList ?**

**\* 5) How to define User define Class object?**

**\* 6)**

**\*/…………………..**

**package** NaveenCollections;

**public** **class** Employee {

String name;

**int** age;

String dept;

Employee(String name , **int** age, String dept){

**this**.name = name;

**this**.age = age;

**this**.dept = dept;

}}

**package** NaveenCollections;

**public** **class** ArrayListEmployee {

String name;

**int** age;

String dept;

ArrayListEmployee(String name , **int** age, String dept){

**this**.name = name;

**this**.age = age;

**this**.dept = dept;

}}

========================

**NonGeneric ArrayList**

**package** NaveenCollections;

**import** java.util.ArrayList;

**public** **class** ArrayListConceptNonGeneric {

**public** **static** **void** main(String[] args) {

**int** a [] = **new** **int**[3]; // static array

// Dynamic Array- ArrayList

ArrayList ar = **new** ArrayList();

ar.add(10); // 0th position

ar.add(20); // ist position

ar.add(30); // 2nd position

//Size of ArrayList

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

ar.add(40); //3rd position

ar.add(50); // 4th position

ar.add(50); //Duplicate value will be stored

//add any kind of value or any type of value

ar.add(13.50); //double

ar.add("I am good"); // String

ar.add("a"); // Character

ar.add(**true**); // boolean

System.***out***.println(ar.size()); // Size of Array

System.***out***.println(ar.get(4)); // to get the value from a index. Random Access. we just the index name/ number.

//it will give whatever the value is available on that particular location.

// To print all values

**for** (**int** i = 0; i <ar.size(); i++) { // i<5 = because the last index 5

System.***out***.println(ar.get(i)); // how to get the value

}}}

**/\*What do you mean by ArrayList Concept?**

**1) Dynamic Array. Arraylist is so dynamic in nature. That's the property of ArrayList.Here we don't define the size but in static array**

**we always define the size.if size is 3 .and we can not go beyond that particular size.So Maximum 3 values we can store here.**

**But In ArrayList there is no limit for size , we can store end number of values in particular ArrayList object.**

**properties of Array List -**

**a) it can contain duplicate values /elements also.**

**b) Maintain insertion order .it insert the values on the basis of ordering like first location 0 , then 1 then 2 then 3 likew this --.**

**like we are inserting the values but it will maintain on the basis of ordering. 0, 1,2,3,4,5,6,....**

**c) It is not synchronized that means it is not thread safe. that is ArrayList is slow as compared to other connections.**

**d) If you want to fetch the value from that particular ArrayList, it allows you random access to fetch any elements. because**

**it stores the values on the basis of indexes.**

**Now if you want to try what exactly the value is available on 4th position or 3rd position. using get() method. and pass the index.**

**This is| int i [] = new int[3];| normal static array if i try to achieve the forth value or fifth value .**

**it will give you ArrayIndexOutOfBound Exception. because this is static in nature. The size is fixed. Generally**

**don't use an static array for designing the application or dynamic application.**

**The size is three it means I can Store only and only in this particular array or variable Maximum 3 integer values. This is bigest problem**

**that is size is fixed**

**2) How to solve this problem?**

**To resolve this problem we use dynamic array. one is the most important dynamic array is available that is called ArrayList.**

**In ArrayList there is no problem with the size.The moment you add the value it automatically the size will be increased.**

**Example - Simple One Object I create like This is ArrayList Object which is available in the form of class object. and you**

**will store number of values here. n number values you can store in this particulare arraylist because size does not matter in ArrayList**

**concept.**

**3) How to do that ?**

**i) we have one default class available that is called ArrayList and this is default I am creating object of ArrayList which is**

**representing by ar..**

**ii) This arrayList we have to import from java.util package**

**iii) it has some methods like add() .ar.add() method is there.**

**iv) it has some methods like add() .ar.add(),**

**v) here like I have stored 3 values here. If you check what is the current size. So size or length of ArrayList is going on. syso(ar.size()).**

**it will give you this will be stored at zeroth position ar.add(10); and then first position and then 2nd position. it means three values are there.**

**vi) If I add 2 more values, ar.add(40);and ar.add(50);.So this particular values will be stored at 3rd position and then 4th position .**

**So here values i have stored. Last index is 4 and size of ArrayList.**

**v) in code line yellow line means warning.**

**vi) To print all the values from the ArrayList we use 1) using for loop or 2) using Iterator also.**

**Whenever any collection is there in java either it is array or ArrayList to retrieve / print all the values we have to use for loop.**

**Because it stores values on the basis of indexes.So I can start my index 0 to 5. in Loop we also write like this 0 then i++ then 1,2,3 the n4 then 5..**

**One by One loop will be executed and one by one retrieve the value like this.This is how we use for loop to retrieve / achieve all the values**

**\*/**

**-0=**

**Generic statement about ArrayList**

**What is a Behavior ?**

**How Exactly ArrayList work Internally?**

**What is Capacity ?**

**How to declare an ArrayList ?**

**How to Iterate an ArrayList ?**

**What are the advantage ArrayList with Array?**

**============**

**ArrayList is a default class which is available and Implementing List Interface that is available in java**

**It will behave like a dynamic Array also.**

**Dynamic Array means -When I declared an simple Array not arrayList, So How will you declare**

**Let’s see**

**int** [] a = **new** **int**[4];

It means this is an Integer Array that I have declared , but the problem with this particular Array is I can store maximum 4 values . Not more than 4 values.Otherwise it will start giving you array index out of bound exception because inside the memory It is already divided into four equal parts. there is no fifth value.

So if we create the indexing over here this is zero this is one, two,three and four and then I can store the value on these indexing only zero one,two and three.

I can not go beyond this [outside of index . it may not be left or right] and I can go beyond 0 also.

**package** ArrayListConcept;

**public** **class** ArrayListDemo {

**public** **static** **void** main(String[] args) {

//Dynamic Array means -When I declared an simple Array not arrayList, So How will you declare

//Let’s see

**int** i[] = **new** **int**[4];

i[0]= 100;

System.***out***.println(i[0]);

/\*i[1]= 60;

i[2]= 101;

i\*/[3] =110;

For example if I write let's see I want to store some value I can store between i[0] to i[3] three only .let's see i[0]= 100;

The moment i'll print usig System.***out***.println(i[0]); the value of i[0]. i'm getting i[zero]= 100. and

the moment I print the value let's see what exactly the value is available on i[3]. so i[3] I don't have any value but if the

value is not available**, it will start giving you zero. so it will give you the default value of integer over here**.

but the moment I try to store the value on fourth index but there is no fourth index. let's say I'm storing 200. So at line number i[4], it will start giving you error over here. We will get **array index out of bondexception** over here at line number i[4].

Because there is no i[4] inside the memory that is the problem in Array. The size is already fixed and it's already divided into four equal parts.

So if I try to write with minus one. minus one is possible it means

before zero? no that is also not possible. you will start getting the same error that no array index out of bond exception. your index is minus one which does not exist over here. so this is the problem with the airing. I cannot go beyond the size. I cannot go beyond the size. So these type of arrays are called static array.

**package** ArrayListConcept;

**public** **class** ArrayListDemo {

**public** **static** **void** main(String[] args) {

**int** i[] = **new** **int**[4];

i[0]= 100;

System.***out***.println(i[0]); // print the value of i[Zero]

System.***out***.println(i[3]);

} }

Dynamic array-

To solve this particular problem we have to use a dynamic array.

The concept of dynamic array which is available in the form of

Arraylist. now arraylist is a class.

**So there are three ways of declaring error list**-

1. one is you simply create a default ArrayList with default generics.In this case simple create the object of Array list.

**ArrayList ar = new ArrayList(); like that .here A** capital because it's a class.

1. Arraylist is available in a different package. so you have to import this particular package this particular class from the java.util package. and simply import that
2. but if you notice One thing this default generics are not recommended, because it's giving you one warning **that ArrayList is a raw type. References to generic type ArrayList<E> should be parameterized** .

**Raw type** means I can add any kind of value over here.so let's see if I'm adding air dot add

ar.add(100);

ar.add(150);

ar.add(200);

and I'm adding some string value let's see-

ar.add("Testing");

ar.add("t");

ar.add("Testing");

ar.add("t");

and i'm writing some double value or some character value

**ar.add(12.30);**

I can add some boolean value also let's

**ar.add(true);**

then if you print your ArrayList. then printing directly -> ar. so it's printing all the values but this is raw type. it means any kind of data I can store over here. but java doesn't like this. Java says that no. This is a raw type it means you are trying to add any kind of value, Java will give you a warning it's not an

error it will give you a warning it's saying that reference to type in ArrayList should be parameterized.

It means I have to give the generics over here.

So how to get the generics for such type of Array list where we can allow all the different types of values.

So I simply write a bracket <> like that and I simply write object type of data I want. So both the sides left hand side and the right hand side you place object type of data like that >

**ArrayList <Object> ar = new ArrayList <Object> ();.**

After that all the warnings are gone over here.

Note-Array list will store the data on the basis of indexing.

ArrayList <Object> ar = **new** ArrayList <Object> ();

ar.add(100); //0

ar.add(150);// 1

ar.add(200); // 2

ar.add("Testing"); // 3

ar.add("t"); //4

ar.add("Testing"); // 5

ar.add(12.30); // 6

ar.add("t"); // 7

ar.add(**true**); // 8

so ar.add(100); this will be stored at zero position. This ar.add(150);// this will be stored at first position like that. It will maintain the order.

Arraylist will maintain the order, maintain the indexing to store the value.

**Q what exactly the value on a specific index?**

So I simple print system.out.println and say ar.get() method .

**So add method is to add the values and get() method is to get the values.**

Q-what is the value is available on the second position ?

So 2 simplear.get2. 2 is testing so we will

08:55

be getting

08:56

testing over here simple

08:59

like that perfect now if i ask you

09:02

system dot outer pentalin

09:04

what exactly the value is available on

09:06

ar dot

09:07

a get i'm passing on six index

09:11

on six there is no six index that does

09:13

not even exist inside the memory

09:15

but this particular error list the

09:17

maximum the value the maximum the

09:19

highest index

09:20

is 5 over here but there is no 6

09:23

so in that case again you will start

09:25

getting the same error

09:26

index out of bound exception over here

09:30

right case because it does not even

09:31

exist same thing if i write can we have

09:33

the negative indexing let's see

09:35

i'm trying to say that okay ar dot get

09:37

minus 1 let's see what is happening same

09:39

error

09:40

so negative indexing is also not allowed

09:43

in java

09:44

remember this thing it's allowed in

09:45

python but in java it is not allowed

09:48

so it will start giving you exception

09:49

over here like that

09:51

but let's see after adding five values

09:55

right what is the current size so 0 to 5

09:57

means 6 values are there so if i really

10:00

want to check

10:00

what is the size of this particular

10:02

error list so simple right ar dot

10:05

size over here so ar dot sign will give

10:07

you

10:08

whatever the current size is six over

10:10

here it means six elements are available

10:12

so if i ask you that okay what is the

10:15

lowest index

10:16

always remember guys for error list or

10:18

array even

10:20

lowest index if i ask you that okay what

10:22

is the lowest index is equal to

10:24

what will happen i really want to print

10:26

array index

10:28

0 right so 0th index is the lowest index

10:32

if i ask you what is the

10:34

highest index so highest index is what

10:36

highest index i could i can write is

10:38

equal to

10:39

whatever the length that you are getting

10:40

air dot size

10:42

minus one so highest index

10:46

you will get the lowest index equal to 0

10:48

highest and is equal to 5

10:49

and the size of the array is 6 right now

10:53

now the good thing is that the moment i

10:55

add some more values let's add some more

10:57

values over here

10:58

after that let's see i'm adding ar.add

11:02

i'm adding 400 and ar dot add

11:06

i'm adding 500 now the size got

11:09

increased automatically

11:11

programmatically okay internally i'm

11:14

adding two values

11:15

so internally what will happen

11:17

automatically it will add two more

11:18

values on sixth position

11:19

and this is on seventh position now i'll

11:23

write system.out.println

11:25

i want to see that okay what is the

11:26

current size so ar dot size

11:28

what will you get now you will start

11:31

getting

11:32

eight the earlier size was six after

11:34

adding two more values now we are

11:36

getting eight over here

11:38

right so that's why we hence proved that

11:41

error lists are dynamic in nature the

11:43

moment you add the values

11:44

automatically it will be increased over

11:46

here like that

11:48

now one important thing i'm going to

11:50

explain you guys that

11:51

the moment you create an error list

11:54

right okay so let me create a new class

11:56

so this is a simple very basic system

11:57

about error list

11:58

let's create one more class over here

12:01

and the class name is

12:04

virtual capacity concept of error list

12:07

so i'm simply writing virtual capacity

12:08

very very important question for

12:11

interview point of view

12:12

you should know they will ask you the

12:13

time of interview what do you mean by

12:15

virtual capacity of

12:16

error list so let's see i'm going to

12:18

create one error list over here

12:21

error list ar is equal to new error list

12:25

okay like that and import this

12:28

particular error list

12:30

now what type of data i can add let's

12:32

see only object type of data

12:34

any kind of object type of data i can

12:37

store over here

12:38

so that the warning is gone this warning

12:42

we are getting because we are not using

12:43

this particular variable so far

12:45

now what happens the moment to declare

12:47

this particular variable

12:48

or this this particular error list

12:50

inside the memory guys what

12:52

happens inside the memory it will

12:56

create an analyst object like that

12:59

assume that and by default it will be

13:02

divided into 10 equal parts

13:03

so this is 1 two three four

13:07

five six seven and let's see eight

13:10

and then one more let's see nine like

13:13

that so zero to

13:14

nine so it will be divided into ten

13:17

equal parts

13:18

internally but this is called the

13:20

virtual capacity guys so initial virtual

13:22

capacity of the area list

13:24

is 10 by default this is not the

13:27

physical capacity

13:28

if i ask you that okay over here what is

13:31

the physical capacity it means what is

13:32

the actual size of this particular error

13:34

list

13:35

the actual size of error list is zero

13:38

but internally java guys are very smart

13:39

java guys say that okay fine we will not

13:41

give you

13:42

randomly 100 size or something like this

13:44

we will just give you that okay initial

13:46

10 capacity we will reserve

13:48

as a virtual capacity for you and then

13:50

the moment you add the value

13:52

we will keep increasing your physical

13:53

capacity and then the virtual capacity

13:56

will be

13:56

decreasing so let's see how exactly it

13:58

works now

14:01

always remember whenever we are

14:02

calculating system dot out of twin

14:04

talent ar dot size

14:06

let me write inside the bracket ar dot

14:08

size

14:10

what is the size right now so ar dot

14:13

size will always work upon the physical

14:14

capacity that physically how many values

14:16

are there

14:17

is there any memory allocation no

14:19

virtual capacity will not take any space

14:21

because this is a virtual space and

14:23

virtual space doesn't have any bytes or

14:26

bits

14:26

so if you run it so let's run with job

14:29

application

14:30

and you see that okay the size is we are

14:32

getting zero so always remember it will

14:34

calculate the pc pc means the physical

14:36

capacity right now is zero

14:40

okay we don't have any formula to check

14:42

that okay about how many virtual

14:43

capacity initially

14:45

but the moment i add the values let's

14:47

see i'm adding system dot sorry air dot

14:49

add i'm adding 100 over here

14:51

so what will happen the 100 will be

14:54

immediately added over here like that

14:56

100

14:57

now what is the physical capacity i'll

15:00

check

15:00

system dot auto printer and to check the

15:02

physical capacity i have to write ar dot

15:05

size once again and then if you run it

15:07

now the physical capacity got increased

15:09

by one

15:10

so now this is showing the physical

15:12

capacity is equal to 0

15:13

pc is equal to 1

15:17

but what is this virtual capacity how

15:20

will you determine

15:21

that there is a virtual capacity of 10

15:23

so just to showcase this thing guys

15:25

what i'm going to do let's see i'm going

15:26

to add some more values ar dot add

15:29

let's say i'm adding 200 ar dot add

15:33

300 ar dot add

15:36

let's see 400 perfect and then i'm

15:39

adding one more size over here i mean

15:41

getting one more size over here

15:43

so this is one and three more values so

15:45

i should get

15:46

size is equal to four so size will be

15:49

checking what is the current physical

15:50

capacity

15:52

so let's run it and let's see four

15:55

now i'm putting in a debugger over here

15:57

let's see at line number 12

15:59

and then run this particular program in

16:00

a debugging mode and we will see

16:02

internally

16:03

how exactly it is storing the values so

16:04

right click on it go to debug as a java

16:07

application

16:09

so we see that okay it got stopped over

16:11

here like that

16:12

and you just mouse over on ar what do

16:15

you see

16:15

size is zero over here you can see so

16:18

this is the physical capacity size

16:20

you can see it is zero okay you can see

16:23

there are no data size is equal to zero

16:25

and let's print the size press f6

16:28

again press f6 now add we have added one

16:31

more value at the zeroth position

16:33

over here now you bounce over on ar you

16:36

can see that okay

16:37

immediately at zeroth position hundred

16:39

got added and you just open this

16:41

can you see this

16:45

can you see zero two i hope you can see

16:48

that zero to nine

16:49

it means internally the virtual capacity

16:52

got created on the first zeroth index we

16:55

have added hundred over here

16:56

brought from one to nine all these

16:59

values are null right now because we

17:00

don't have any value

17:01

so currently what is a virtual capacity

17:04

the virtual capacity is equal to

17:07

what is equal to nine right now and

17:09

physical capacity is equal to one

17:11

but first time if you declare this

17:13

particular variable first time the

17:14

virtual capacity is equal to

17:16

10 always remember ways right and the

17:19

moment we add the value

17:21

one by one the physical capacity will

17:22

keep increasing and the virtual capacity

17:25

will keep

17:26

a decreasing over here right now let's

17:28

see i'm adding three more values

17:30

18 19 and 20 line number

17:33

now let's check what is the ar now you

17:36

see that okay ar

17:38

count is equal to 4 because we have

17:40

added 4 values

17:42

now if you check this from 0 to 3 you

17:45

can see 0 1 2

17:46

3 all these four indexes from 0 to 3

17:50

values got added now the virtual

17:52

capacity is 4 to 9

17:54

it means 6 the virtual capacity is 6 now

17:58

so likewise the default virtual capacity

18:01

is

18:02

10 in java for error list remember this

18:06

thing

18:06

now let's see this is another concept

18:08

that okay what if i want to

18:10

change the initial virtual capacity so

18:12

if you see that okay let's see i'm

18:13

declaring

18:14

20 over here i don't want 10 i said okay

18:16

no reserve 10

18:17

virtual capacity for this particular

18:19

error list so if you run

18:21

you just pass in the constructor pass 20

18:23

over here and let's

18:24

debug as java application once again and

18:27

let's see what happens

18:28

now you add first value air dot add 100

18:32

and then mouse over on ar

18:34

and check what exactly it is showing for

18:36

the virtual capacity

18:37

so let's open this now let's

18:41

see this can you see that 0 to

18:44

19 initially got created on the first

18:47

position

18:48

integer got added it means 100 got added

18:50

you can open this particular integer

18:53

and you can see that okay value is equal

18:54

to 100 but from 1 to 19

18:57

everything is null it means we have

18:59

changed the initial virtual capacity to

19:01

20 now so instead of 10 it's giving you

19:03

20. so this is another interview

19:05

question that okay how will you change

19:07

the virtual capacity

19:08

you can pass the capacity number over

19:10

here i want 10 25 or whatever

19:14

like that okay guys so remember this

19:17

concept as well

19:19

right so this is about the virtual

19:21

capacity now

19:23

can we store only

19:26

specific type of data in a particular

19:28

error list yes so let's talk about

19:31

generics

19:35

error list so again very important

19:38

and practically also we have to use

19:40

always the generics type of data

19:42

inside the error list it means whenever

19:43

you are creating an error list you have

19:45

to decide

19:46

that what kind of data you want so let's

19:48

see i'm creating one error list

19:50

i want only integer type of data let's

19:52

say i want to store only student marks

19:55

so i'll store like that and i'll write a

19:57

wrapper class over here that is

19:58

integer integer let's see this is my

20:01

marks list

20:02

is equal to new error list

20:06

and then i'll write integer over here

20:11

like that and import your error list now

20:13

in this particular error list

20:15

i can add only and only integer value

20:17

this is not object type or raw type

20:19

the moment i write add you can see that

20:21

only integer is coming

20:23

it means i can store only integer so i

20:24

can store only hundred

20:26

so the moment i try to add mark list dot

20:28

add let's see i'm adding 12.33 this is a

20:31

double value

20:32

then it will start giving you the error

20:33

over here so now in this particular

20:36

error list i cannot

20:37

store other than integer values so this

20:39

is strictly not allowed i cannot do that

20:42

so remember this thing this is called

20:44

generics in

20:45

error list now can i store

20:49

other values yeah you can store double

20:52

values

20:53

so both the sides you have to write

20:54

double and let's see some

20:56

double list you want to create let's see

20:58

this is my ar d

21:00

now in this particular ar double you can

21:03

add

21:04

only and only double values 1.33 can i

21:07

add ard dot add

21:09

100 that 100 also you can add it

21:13

yes or no no because 100.0 you can add

21:17

it

21:17

but only 100 you cannot add because 100

21:19

is the pure integer value

21:21

i cannot add it same thing if you really

21:23

want to store let's see some student

21:24

names

21:25

so you can create one arraylist which

21:28

will take only student names and i know

21:30

student names will be in a string type

21:32

so i can store

21:33

let's see this is my student

21:37

i'll write student names

21:40

and both the sides we have to write the

21:42

string over here

21:44

now student names i simply write dot

21:47

add i can add only and only two student

21:50

names let's see my student name is

21:51

tom over here so i can simply add tom

21:54

over here so likewise you can create the

21:57

genetics

21:58

and what type of data you want in your

22:01

error is you can simply write that

22:02

particular type of data

22:04

and remember error list or all the

22:06

collection they will store only and only

22:08

object not the primitive data type if

22:10

you really want to add 100 as a

22:12

primitive value

22:13

then you have to create the wrapper

22:14

class so it will automatically it will

22:16

do the auto boxing

22:18

and up casting into integer and then you

22:21

have to use

22:22

integer over here if the moment you

22:23

write in both the sides let's see if i'm

22:26

writing end over here like that

22:27

this is not allowed okay so you have to

22:30

write the wrapper class both the sides

22:32

remember this thing okay

22:35

like that so this is about the generics

22:38

error list

22:39

now let's talk about one more concept

22:41

that how will you iterate your error

22:43

list

22:43

what are the different ways of getting

22:46

the values

22:47

from the error list so that is again

22:49

very very common

22:50

a topic interval equation you're

22:52

designing your framework or using

22:54

practically also

22:55

you have to know what are the different

22:57

ways of iterating or getting the values

22:59

from the error list so let's see

23:01

uh error list

23:06

iteration and select the main method

23:10

click on finish okay now let me remove

23:13

this

23:13

drawing now

23:17

let's see i'm going to create one error

23:19

list where i add some values over here

23:21

so let's see

23:23

i'm going to create one error list which

23:24

is a string type of data i'm going to

23:26

store

23:27

so a string let's see this is my student

23:30

list

23:31

is equal to new error list and

23:34

both the sites we have to write student

23:36

over here

23:40

okay like that sorry string over here

23:46

and then import this fine now in this

23:50

particular list i'm going to add some

23:51

values over here let's see dot add

23:53

i'm adding tom like that

23:57

and same way i'm adding some more

23:59

students

24:01

let's see naveen let's see steve

24:05

and let's see lisa

24:08

these four students let's see i have

24:10

added and it will store the value on the

24:12

basis of indexing so this will be 0

24:14

this is 1 this is 2 and this is 3.

24:18

right now i want to iterate the value so

24:21

i'll be using a first method as a

24:22

typical for loop

24:24

okay so i'll be writing let's see

24:25

typical for loop we can use that

24:28

in simple right a for loop integer i

24:31

is equal to 0 because i know that it

24:33

restore the value on the basis of

24:34

indexing

24:36

i less than your student list

24:40

dot size and then i plus plus

24:44

simple and then i'll be printing the

24:45

value so i simply write

24:47

system.out.println

24:48

what is your list name my list name is

24:50

your student list

24:51

dot get and what is the index index is i

24:54

over here

24:55

that's it so my index is equal to i

24:57

equal to 0 first time

24:59

i less than size is equal to four you

25:00

can see four values are there

25:03

condition is satisfied get i it means

25:05

get zero so let me give me the tom

25:08

then i plus plus i will be one two and

25:10

three so i'll be getting tom naveen

25:12

steven lisa

25:14

one by one on the console so you can see

25:17

that okay tom naveen is steve lisza i'm

25:19

getting it

25:20

so this is a typical for loop i can use

25:22

it second way is that you can use a

25:24

simple for each loop also

25:26

so how to use that i'll be writing for

25:29

and then i'll be writing what type of

25:30

data that you have in this particular

25:32

error list we have a string type of data

25:35

so i'll be creating one string

25:36

variability string s

25:39

colon and what is your student what is

25:41

your list

25:42

so my student list is my list name this

25:45

is my object name

25:46

and then i'll simply write system.org i

25:49

print s over here

25:50

that's it so you can create any variable

25:52

name whatever variable name so this

25:54

s will go to the list one by one it will

25:56

pick the data one by one

25:58

by using the for loop and then we are

26:00

printing it on the console

26:01

so you can see that okay tom levine

26:03

steve these are getting printed once

26:04

again

26:05

let me put a separator so that you will

26:07

see separately

26:10

like that

26:13

so let's run it so you can see that tom

26:16

naveen steve visa

26:17

again getting printed over here like

26:18

that

26:21

okay third way is that you can use

26:24

lambda function also guys so how to use

26:27

lambda function

26:29

so simple i can simply use let's see

26:33

according to jade after jdk8 you can use

26:37

simple let's see lambda over here with

26:40

for each

26:40

so how to use that i simply write

26:44

student list dot i'll be using

26:47

stream class okay stream method which

26:49

will give you a stream and then i'll be

26:51

using a dot for each

26:53

and then what you want i want get all

26:56

the elements put a lambda over here

26:58

so i'm using streams with lambda

26:59

actually

27:02

okay java streams with lambda and then i

27:04

simply want to print system dot out of

27:05

twin talent

27:06

print all the values that's it so with

27:09

one line statement you can print all the

27:10

values i'll put a separator over here

27:13

so that you can see the output properly

27:15

now let's run it

27:17

you can see again tom naveen steve lisa

27:19

and

27:20

printing it on the console so this is

27:22

another way of

27:23

printing the values okay

27:27

next thing is that you can use iterator

27:30

so as i told you that okay iterator is

27:32

an interface

27:33

which is used to iterate the values from

27:35

the collection

27:36

error list is also a collection so i can

27:38

apply iterator there as well

27:40

so let's see how to do that so for doing

27:42

this let's see

27:44

this is my error list on this particular

27:46

error list i'll apply my iterator

27:48

so i simply write dot iterator over here

27:50

so the moment i apply iterator

27:52

if you mouse over this method i'll apply

27:54

it will give you iterator of a string

27:55

over here

27:56

so i'll store inside the iterator of a

28:00

string

28:02

here let's see this is iterator id

28:05

reference that i have created and this

28:07

iterator you have to import

28:10

from java.util package perfect

28:13

and then i can create a while loop let's

28:15

see while i simply write

28:17

it dot has next it means

28:20

keep running this particular while loop

28:23

until

28:24

this iterator is having the next value

28:26

in this particular error list

28:27

because on this particular list we have

28:29

applied the iterator so iterator will

28:31

store all the values in this particular

28:32

id object

28:33

and then we are picking that okay if i t

28:35

has a next value

28:37

if it is having it then print it how to

28:39

print you simply write

28:41

i t dot next i i'll be writing it

28:44

printed up to the next value after lisa

28:47

there is no next value so condition is

28:49

false

28:50

so it will not print on the console so

28:51

let's see i'll put again one more

28:53

separator over here so that you can see

28:55

the output separately

28:56

and then you run it you can see again

28:59

it's printing tom

29:00

steve over there like that

29:02

[Music]

29:03

simple so with the help of iterator also

29:06

you can simply do that so this is

29:08

one two three four different ways i have

29:12

told you to print all the values from

29:15

error list

29:16

remember this thing guys simple concept

29:19

very simple right okay

29:22

uh other than that i think in this

29:25

particular first chapter

29:26

that is good yeah one more thing how to

29:28

declare

29:29

an error list that i told you right

29:31

simple this is a way you can declare you

29:33

can provide the generics also

29:35

there is one more way you can pass if

29:38

you pass zero it means nothing

29:40

the default capacity will be 10 you can

29:41

pass the capacity also like c6

29:44

you can create one error list with

29:47

another error list

29:48

so how to do that so very simple right

29:54

list with other connection

29:58

how to declare that

30:01

okay so let's see i'm going to create

30:04

one error list

30:06

sorry

30:08

one error list and let's see this time

30:11

error list i'm giving

30:13

some name list okay is equal to

30:17

new error list or let's see

30:20

i simply say this is my number numbers

30:23

is equal to new

30:24

error list i'm giving like that

30:27

okay and then i want to store integer

30:29

type of data

30:31

both the sides i'll write integer over

30:32

here

30:34

because numbers will be integer and you

30:37

put a bracket

30:37

and inside the constructor you simply

30:39

write one class is there arrays

30:41

dot as list it means what kind of list

30:44

you want

30:45

i simply pass 10 comma 20 comma 30 comma

30:48

40.

30:52

like this it means inside the

30:54

constructor of the error list

30:56

see guys this is the object whenever we

30:58

are creating the object in java

30:59

constructor will be called

31:00

so arrays dot as list and passing these

31:03

four values over here

31:04

now this particular numbers if you

31:06

directly printed on the console system

31:08

dot our repentant i simply write

31:10

numbers and let's see what happens

31:12

numbers

31:15

so let's run it so we can see 10 20 30

31:17

40 getting printed on the console and

31:19

you can write a for loop also

31:21

get method and all those things you can

31:23

do that to get the values

31:24

from this particular numbers error list

31:26

so what we are doing we are creating one

31:28

error list with the help of

31:29

another error list over here like that

31:33

so that also you can simply do that

31:36

okay can we create other than uh this so

31:38

yeah we can do some strings also

31:40

so let's see now next time i'm creating

31:42

one string over here

31:44

so let's see this is our names and

31:46

arrows dot as list

31:48

and the string will be what in double

31:49

quotes so i'll be writing let's see

31:51

give me all the programming languages so

31:53

java

31:55

and then i'm writing let's see python

31:58

and let's see i'm writing js means

32:00

javascript and then let's see i'm

32:02

writing php

32:04

like that and both the sides we have to

32:06

write string over here

32:09

that also you can do that and then i

32:11

really want to bring names so simple

32:13

right

32:13

or you can iterate also or you can

32:15

directly print names over here

32:17

and then you run it and let's see so you

32:19

can see java python jsphp

32:22

so this is zeroth value first value

32:24

second value and third value

32:26

if i say that okay hey what is the

32:28

current size of this particular names

32:30

so i simply write names dot size over

32:33

there

32:34

and then you run it it will give you

32:36

size is

32:37

four over here like that okay guys

32:41

so these are the four important things

32:42

first of all i'll quickly repeat

32:45

how to declare an error list what do you

32:46

mean by generics what do you mean

32:48

raw type how to get the value what do

32:50

you mean by array index out of bond

32:52

exception

32:52

negative indexing is not allowed in java

32:55

what is lowest index what is highest

32:57

index

32:58

right in this particular case you will

33:00

be getting what

33:01

array index

33:05

out of bound exception you will be

33:08

getting it

33:10

people will ask you what do you mean by

33:11

array index outdoor bound exception so

33:13

you have to explain like that

33:14

what are the different ways of declaring

33:16

an error list so you can declare with

33:17

other list also

33:19

now what is the concept of virtual

33:20

capacity and the physical capacity what

33:22

is the default capacity so always

33:24

remember the default capacity is

33:28

is always 10 default capacity of error

33:31

list is 10 this is the right answer

33:33

10. what do you mean by generic generic