Java Arrays

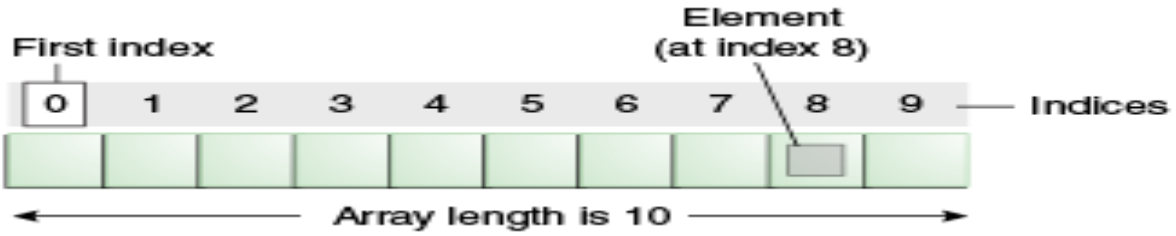
An array is a collection of similar types of elements which has contiguous memory location.

Java array is an object which contains elements of a similar data type. Additionally, The elements of an array are stored in a contiguous memory location. It is a data structure where we store similar elements. We can store only a fixed set of elements in a Java array.

Arrays in Java are index-based, the first element of the array is stored at the 0th index, the 2nd element is stored on the 1st index and so on.

In Java, an array is an object of a dynamically generated class. Java array inherits the Object class, and implements the Serializable as well as Cloneable interfaces. We can store primitive values or objects in an array in Java. Like C/C++, we can also create single dimensional or multidimensional arrays in Java.

Moreover, Java provides the feature of anonymous arrays which is not available in C/C++.



Advantages

- **Code Optimization:** It makes the code optimized, we can retrieve or sort the data efficiently.
- **Random access:** We can get any data located at an index position.

Disadvantages

- **Size Limit:** We can store only the fixed size of elements in the array. It doesn't grow its size at runtime. To solve this problem, collection framework is used in Java which grows automatically.

Types of Array in java

There are two types of array.

- Single Dimensional Array
- Multidimensional Array

Single Dimensional Array in Java

Syntax to Declare an Array in Java

dataType[] arr; (or)
dataType []arr; (or)

dataType arr[];

Instantiation of an Array in Java

1. arrayRefVar=**new** datatype[size];

Example of Java Array

Let's see the simple example of java array, where we are going to declare, instantiate, initialize and traverse an array.

/Java Program to illustrate how to declare, instantiate, initialize

```
//and traverse the Java array.
class Testarray{
public static void main(String args[]){
int a[]=new int[5];//declaration and instantiation
a[0]=10;//initialization
a[1]=20;
a[2]=70;
a[3]=40;
a[4]=50;
//traversing array
for(int i=0;i<a.length;i++)//length is the property of array
System.out.println(a[i]);
}}
```

Output:

```
10
20
70
40
50
```

```
import java.util.*;
public class array1
{
public static void main (String args[])
{
Scanner sc=new Scanner(System.in);
int[] num = {1,2,3,4};
String[] str = {"meera","prabha", "devyani"};
System.out.println("The num is");
for (int i=0;i<= 3;i++)
{
	System.out.println(num[i]);

}
System.out.println("The names is");
for (int j=0;j<= 2;j++)
{
	System.out.println(str[j]);

}
}
}
```

import java.util.*;

```
public class array2
{
public static void main (String args[])
{
Scanner sc=new Scanner(System.in);
int num[]=new int[3];

for (int i=0;i<= 2;i++)
{
	num[i]=sc.nextInt();

}

System.out.println("The num is");
for (int i=0;i<= 2;i++)
{
	System.out.println(num[i]);

}


String str[]=new String[3];

for (int i=0;i<= 2;i++)
{
	str[i]=sc.nextLine();

}

System.out.println("The names is");
for (int i=0;i<= 2;i++)
{
```

```
System.out.println(str[i]);


}

}

}
```

Declaration, Instantiation and Initialization of Java Array

We can declare, instantiate and initialize the java array together by:

1. **int** a[]={33,3,4,5};**//declaration, instantiation and initialization**

Let's see the simple example to print this array.

```
1. //Java Program to illustrate the use of declaration, instantiation
      //and initialization of Java array in a single line
class Testarray1{
public static void main(String args[]){
int a[]={33,3,4,5};//declaration, instantiation and initialization
//printing array
for(int i=0;i<a.length;i++)//length is the property of array
System.out.println(a[i]);
}}
```

Output:

```
33
3
4
5
```

For-each Loop for Java Array

We can also print the Java array using **for-each loop**

. The Java for-each loop prints the array elements one by one. It holds an array element in a variable, then executes the body of the loop.

The syntax of the for-each loop is given below:

```
for(data_type variable:array){
//body of the loop
}
```

Let us see the example of print the elements of Java array using the for-each loop

```
//Java Program to print the array elements using for-each loop
class Testarray1{
```

```
    public static void main(String args[]){
    int arr[]={33,3,4,5};
    //printing array using for-each loop
    for(int i:arr)
    System.out.println(i);
    }}
```

Output:

```
33
3
4
5
```

for(i=0;i>=10;i++);

{

}

Multidimensional array

Multidimensional Array in Java

In such case, data is stored in row and column based index (also known as matrix form).

Syntax to Declare Multidimensional Array in Java

1. dataType[][] arrayRefVar; (or)
2. dataType [][]arrayRefVar; (or)
3. dataType arrayRefVar[][]; (or)
4. dataType []arrayRefVar[];

Example to instantiate Multidimensional Array in Java

1. **int**[][] arr=**new int**[3][3];//3 row and 3 column

Example to initialize Multidimensional Array in Java

1. arr[0][0]=1;
2. arr[0][1]=2;
3. arr[0][2]=3;
4. arr[1][0]=4;
5. arr[1][1]=5;
6. arr[1][2]=6;
7. arr[2][0]=7;
8. arr[2][1]=8;
9. arr[2][2]=9;

Example of Multidimensional Java Array

Let's see the simple example to declare, instantiate, initialize and print the 2 Dimensional array.

```
//Java Program to illustrate the use of multidimensional array
class Testarray3{
public static void main(String args[]){
//declaring and initializing 2D array
int arr[][]={{1,2,3},{2,4,5},{4,4,5}};
//printing 2D array
for(int i=0;i<3;i++){
 for(int j=0;j<3;j++){
   System.out.print(arr[i][j]+" ");
 }
 System.out.println();
}
}}
```

Output:

```
1 2 3
2 4 5
4 4 5
```

Ex:

```
import java.util.*;

public class array3

{

public static void main (String args[])

{

Scanner sc=new Scanner(System.in);

int num[][]=new int[2][2];


for (int i=0;i<= 1;i++)

{

for (int j=0;j<=1;j++)

{

        num[i][j]=sc.nextInt();


}

}
```

```
System.out.println("The num is");
for (int i=0;i<= 1;i++)
{
for (int j=0;j<=1;j++)
{

    System.out.println(num[i][j]);

}

}
}
}
```

Ex:

```
import java.util.*;
public class arraymulti
{
public static void main (String args[])
{
Scanner sc=new Scanner(System.in);
int[][] num = {{1,2},{4,5}};

System.out.println("The multi array is");

for (int i=0;i<= 1;i++)
{
for (int j=0;j<= 1;j++)
{
    System.out.println(num[i][j]);
}
}
```

```
}
}
```

Jagged Array in Java

If we are creating an odd number of columns in a 2D array, it is known as a jagged array.

In other words, it is an array of arrays with different numbers of columns.

A jagged array is an array of arrays such that member arrays can be of different sizes, i.e., we can create a 2-D array but with a variable number of columns in each row. These types of arrays are also known as Jagged arrays.

```
//Java Program to illustrate the jagged array
class TestJaggedArray{
   public static void main(String[] args){
      //declaring a 2D array with odd columns
      int arr[][] = new int[3][];
      arr[0] = new int[3];
      arr[1] = new int[4];
      arr[2] = new int[2];
      //initializing a jagged array
      int count = 0;
      for (int i=0; i<arr.length; i++)
         for(int j=0; j<arr[i].length; j++)
            arr[i][j] = count++;
      //printing the data of a jagged array
      for (int i=0; i<arr.length; i++){
         for (int j=0; j<arr[i].length; j++){
            System.out.print(arr[i][j]+" ");
         }
         System.out.println();//new line
      }
   }
}
```

Output:

```
0 1 2
3 4 5 6
7 8
```

In Java, an array is an object. For array objects, a proxy class is created whose name can be obtained by the getClass().getName() method on the object.

//Java Program to get the class name of array in Java

```
class Testarray4{
public static void main(String args[]){
//declaration and initialization of array
int arr[]={4,4,5};
//getting the class name of Java array
Class c=arr.getClass();
String name=c.getName();
//printing the class name of Java array
System.out.println(name);

}}
```

Output:

```
I
```