Chapter 6: Arrays in Java

Declaring Array Variables

To use an array in a program, you must declare a variable to reference the array, and you must specify the type of array the variable can reference. Here is the syntax for declaring an array variable –

In Java, here is how we can declare an array.

```
dataType arrayName[];
```

- `dataType` - it can be primitive data types like `int`, `char`, `double`, `byte`, etc.
- `arrayName` - it is an identifier

For example,

```
double data[];
```

Another example:

```
int intArray[];    //declaring array
intArray = new int[20];  // allocating memory to array
```

OR

```
int intArray[] = new int[20]; // combining both statements in one
```

- Arrays are nothing but a data structure that is used to hold the data elements of the same type in a sequential fashion.

How to Initialize Arrays in Java?

In Java, we can initialize arrays during declaration. For example,

```
//declare and initialize and array
int age[] = {12, 4, 5, 2, 5};
```

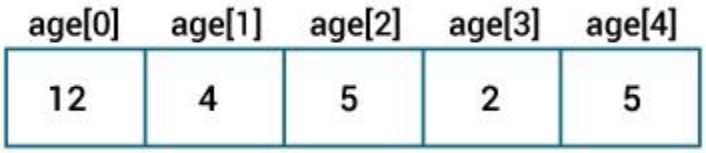
Here, we have created an array named age and initialized it with the values inside the curly brackets.

Note that we have not provided the size of the array. In this case, the Java compiler automatically specifies the size by counting the number of elements in the array (i.e. 5).

In the Java array, each memory location is associated with a number. The number is known as an array index. We can also initialize arrays in Java, using the index number. For example,

```
// declare an array
int[] age = new int[5];

// initialize array
age[0] = 12;
age[1] = 4;
age[2] = 5;
..
```



How to Access Elements of an Array in Java?

We can access the element of an array using the index number. Here is the syntax for accessing elements of an array,

```
// access array elements
array[index]
```

Let's see an example of accessing array elements using index numbers

Example: Access Array Elements

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

   // create an array
   int age[] = {12, 4, 5, 2, 5};

   // access each array elements
   System.out.println("Accessing Elements of Array:");
   System.out.println("First Element: " + age[0]);
   System.out.println("Second Element: " + age[1]);
   System.out.println("Third Element: " + age[2]);
   System.out.println("Fourth Element: " + age[3]);
   System.out.println("Fifth Element: " + age[4]);
 }
}
```

Output

```
Accessing Elements of Array:
First Element: 12
Second Element: 4
Third Element: 5
Fourth Element: 2
Fifth Element: 5
```

Looping Through Array Elements

In Java, we can also loop through each element of the array. For example,

Example: Using For Loop

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

   // create an array
   int age[] = {12, 4, 5};

   // loop through the array
   // using for loop
   System.out.println("Using for Loop:");
   for(int i = 0; i < age.length; i++) {
```

```
      System.out.println(age[i]);
    }
  }
}
```

Output

```
Using for Loop:
12
4
5
```

In the above example, we are using the for Loop in Java to iterate through each element of the array. Notice the expression inside the loop,

```
age.length
```

Here, we are using the `length` property of the array to get the size of the array.

Example: Compute Sum and Average of Array Elements

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

    int numbers [] = {2, -9, 0, 5, 12, -25, 22, 9, 8, 12};
    int sum = 0;
    Double average;
    // access all elements using for each loop
    // add each element in sum
    for (int number: numbers) {
      sum += number;
    }
    // get the total number of elements
    int arrayLength = numbers.length;
    // calculate the average
    // convert the average from int to double
    average =  ((double)sum / (double)arrayLength);

    System.out.println("Sum = " + sum);
    System.out.println("Average = " + average);
  }
}
```

Output:

```
Sum = 36
Average = 3.6
```

In the above example, we have created an array of named `numbers`. We have used the `for...each` loop to access each element of the array.

Inside the loop, we are calculating the sum of each element. Notice the line,

```
int arrayLength = number.length;
```

Here, we are using the length attribute of the array to calculate the size of the array.

We then calculate the average using:

```
average = ((double)sum / (double)arrayLength);
```

As you can see, we are converting the `int` value into `double`. This is called type casting in Java.

Let's create a program that takes a single-dimensional array as input.

(reading array elements from console

```
import java.util.Scanner;
public class ArrayInputExample1
{
    public static void main(String[] args)
    {
       int n;
 Scanner sc=new Scanner(System.in);
 System.out.print("Enter the number of elements you want to store: ");
 n=sc.nextInt();  //reading the number of elements from the that we want to ente
  int array[] = new int[10];  //creates an array in the memory of length 10
  System.out.println("Enter the elements of the array: ");
     for(int i=0; i<n; i++)
     {
        array[i]=sc.nextInt();  //reading array elements from the user
      }
        System.out.println("Array elements are: ");
         // accessing array elements using the for loop
           for (int i=0; i<n; i++)
             {
                 System.out.println(array[i]);
              }
    }
}
```

Output:

```
Enter the number of elements you want to store: 6
Enter the elements of the array:
67
23
45
12
77
90
Array elements are:
67
23
45
12
77
90
```