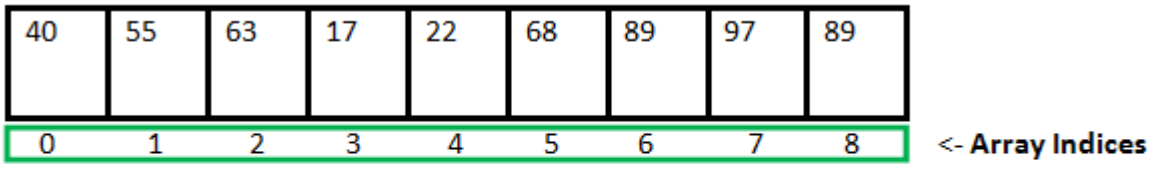
Array in java is a group of like-typed variables referred to by a common name.

Arrays in Java work differently than they do in C/C++.

Following are some important points about Java arrays.

- In Java, all arrays are dynamically allocated.
- Arrays are stored in contiguous memory
- Since arrays are objects in Java, we can find their length using the object property *length*.
- A Java array variable can also be declared like other variables with [] after the data type.
- The variables in the **array are ordered, and each has an index beginning with 0.**
- Java array can also be used as a static field, a local variable, or a method parameter.
- **The size of the array cannot be altered(once initialized).** However, an array reference can be made to point to another array.



Array Length = 9
First Index = 0
Last Index = 8

Creating, initializing, and accessing an Array

One-Dimensional Arrays:

The general form of a one-dimensional array declaration is

type var-name[];
OR
type[] var-name;

// both are valid declarations

int intArray[];
or int[] intArray;

Instantiating an Array in Java

- When an array is declared, only a reference of an array is created.
- To create or give memory to the array, you create an array like this:

The general form of *new* as it applies to one-dimensional arrays appears as follows:
var-name = new type [size];

Here, ***type*** specifies the type of data being allocated, ***size*** determines the number of elements in the array, and *var-name* is the name of the array variable that is linked to the array.

To use *new* to allocate an array, **you must specify the type and number of elements to allocate.**

Example:

int intArray[]; //declaring array

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

OR

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

In a situation where the size of the array and variables of the array are already known, array literals can be used.

int[] intArray = new int[]{ 1,2,3,4,5,6,7,8,9,10 };

// Declaring array literal

- The length of this array determines the length of the created array.
- There is no need to write the new int[] part in the latest versions of Java.
-

Accessing Java Array Elements using for Loop

- Each element in the array is accessed via its index.
- The index begins with 0 and ends at (total array size)-1.
- All the elements of array can be accessed using Java for Loop.

```
// accessing the elements of the specified array
for (int i = 0; i < arr.length; i++)
  System.out.println("Element at index " + i +  " : "+ arr[i]);
```

```
public class StudentInfo {
    public static void main(String[] args) {
        // Arrays to store student information
        String[] names = {"Alice", "Bob", "Charlie", "David", "Eva"};
        double[] marks = {85.5, 92.0, 78.5, 88.0, 76.5};
        String[] courses = {"Math", "English", "Science", "History", "Art"};

        // Display student information
        System.out.println("Student Information:");
        System.out.println("-----------------------------");
        System.out.println("Name\t\tMarks\t\tCourse");
        System.out.println("-----------------------------");

        for (int i = 0; i < names.length; i++) {
```

```
            System.out.println(names[i] + "\t\t" + marks[i] + "\t\t" + courses[i]);
        }
    }
}
```

Create a java program to sort array elements using bubble sort

```
public class BubbleSortExample {
    static void bubbleSort(int[] arr) {
        int n = arr.length;
        int temp = 0;
         for(int i=0; i < n; i++){
                 for(int j=1; j < (n-i); j++){
                         if(arr[j-1] > arr[j]){
                                 //swap elements
                                 temp = arr[j-1];
                                 arr[j-1] = arr[j];
                                 arr[j] = temp;
                         }

                 }
        }

    }
    public static void main(String[] args) {
            int arr[] ={3,60,35,2,45,320,5};

            System.out.println("Array Before Bubble Sort");
            for(int i=0; i < arr.length; i++){
                    System.out.print(arr[i] + " ");
            }
            System.out.println();

            bubbleSort(arr);//sorting array elements using bubble sort

            System.out.println("Array After Bubble Sort");
```

```
            for(int i=0; i < arr.length; i++){
                System.out.print(arr[i] + " ");
            }

        }
}
```

Java Multidimensional Arrays

A multidimensional array is an array of arrays. Each element of a multidimensional array is an array itself. For example,

```
int[][] a = new int[3][4];
```

	Column 1	Column 2	Column 3	Column 4
Row 1	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 2	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 3	a[2][0]	a[2][1]	a[2][2]	a[2][3]

How to initialize a 2d array in Java?

Here is how we can initialize a 2-dimensional array in Java.

```
int[][] a = {
    {1, 2, 3},
    {4, 5, 6, 9},
    {7},
};
```

Example: Print all elements of 2d array Using Loop

```
class MultidimensionalArray {
    public static void main(String[] args)
{
        int[][] a = {
            {1, -2, 3},
            {-4, -5, 6, 9},
            {7},
        };

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

Array Addition

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

class ArrTest {
    public static void main(String[] args)
    {
        int[][] arr1 = { { 1, 2, 3 }, { 4, 5, 6 } };
        int[][] arr2 = { { 4, 5, 6 }, { 1, 3, 2 } };
        int[][] sum = new int[2][3];

        // adding two 2D arrays element-wise
        for (int i = 0; i < arr1.length; i++) {
            for (int j = 0; j < arr1[0].length; j++) {
                sum[i][j] = arr1[i][j] + arr2[i][j];
            }
        }

        System.out.println("Resultant 2D array: ");
        for (int i = 0; i < sum.length; i++) {
            System.out.println(Arrays.toString(sum[i]));
        }
    }
}
```

Example: Program to Multiply Two Matrices

```
public class MultiplyMatrices {

    public static void main(String[] args) {
        int r1 = 2, c1 = 3;
        int r2 = 3, c2 = 2;
        int[][] firstMatrix = { {3, -2, 5}, {3, 0, 4} };
        int[][] secondMatrix = { {2, 3}, {-9, 0}, {0, 4} };

        // Mutliplying Two matrices
        int[][] product = new int[r1][c2];
```

```
        for(int i = 0; i < r1; i++) {
            for (int j = 0; j < c2; j++) {
                for (int k = 0; k < c1; k++) {
                    product[i][j] += firstMatrix[i][k] * secondMatrix[k][j];
                }
            }
        }

        // Displaying the result
        System.out.println("Multiplication of two matrices is: ");
        for(int[] row : product) {
            for (int column : row) {
                System.out.print(column + "    ");
            }
            System.out.println();
        }
    }
}
```

Matrix Multiplication

```
import java.util.Scanner;

public class MatrixMultiplicationExample {

  public static void main(String args[]) {
    int row1, col1, row2, col2;
    Scanner s = new Scanner(System.in);

    // Input dimensions of First Matrix: A
    System.out.print("Enter number of rows in first matrix: ");
    row1 = s.nextInt();

    System.out.print("Enter number of columns in first matrix: ");
    col1 = s.nextInt();

    // Input dimensions of second matrix: B
    System.out.print("Enter number of rows in second matrix: ");
    row2 = s.nextInt();

    System.out.print("Enter number of columns in second matrix: ");
    col2 = s.nextInt();

    // Requirement check for matrix multiplication
  if (col1 != row2) {
      System.out.println("Matrix multiplication is not possible");
      return;
    }
    int a[][] = new int[row1][col1];
    int b[][] = new int[row2][col2];
    int c[][] = new int[row1][col2];

    // Input the values of matrices
    System.out.println("\nEnter values for matrix A : ");
    for (int i = 0; i < row1; i++) {
      for (int j = 0; j < col1; j++) a[i][j] = s.nextInt();
    }
```

```
    System.out.println("\nEnter values for matrix B : ");
    for (int i = 0; i < row2; i++) {
      for (int j = 0; j < col2; j++) b[i][j] = s.nextInt();
    }

    // Perform matrix multiplication
    // Using for loop
    System.out.println("\nMatrix multiplication is : ");
    for (int i = 0; i < row1; i++) {
      for (int j = 0; j < col2; j++) {
        // Initialize the element C(i,j) with zero
        c[i][j] = 0;

        // Dot product calculation
        for (int k = 0; k < col1; k++) {
          c[i][j] += a[i][k] * b[k][j];
        }

        System.out.print(c[i][j] + " ");
      }
      System.out.println();
    }
  }
}
```