**Array**

Array is a collection of similar type of elements that have contiguous memory location.

**Java array** is an object the contains elements of similar data type. It is a data structure where we store similar elements. We can store only fixed set of elements in a java array.

Array in java is index based, first element of the array is stored at 0 index.



**Advantage of Java Array**

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

**Disadvantage of Java Array**

* **Size Limit:** We can store only 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.

**Types of Array in java**

There are two types of array.

* Single Dimensional Array
* Multidimensional Array

**Single Dimensional Array**

**Syntax :**

1. dataType[] arr; (or)
2. dataType []arr; (or)
3. dataType arr[];

**Instantiation of an Array in java**

1. arrayRefVar=new datatype[size];

**Declaration, Instantiation and Initialization**

1. int a[]={33,3,4,5};//declaration, instantiation and initialization
2. **print an array?**
3. **print an array using for each loop?**
4. **Print minimum of array elements?**
5. **Print maximum of array elements?**
6. **Print sum of array elements?**
7. **Print even and odd sum of array elements?**
8. **Print sorting of array elements?**
9. **Print search of array elements?**

**Multidimensional array**

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

**Syntax to Declare Multidimensional Array**

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

**Example to instantiate Multidimensional Array**

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

**Example to initialize Multidimensional Array**

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 array**

1. print multidimensional array?
2. print addition of 2 matrix?
3. print substraction of 2 matrix?
4. print multiplication of 2 matrix?
5. print trancepose of matrix?

**import** java.util.Scanner;

**class** MatrixMultiplication

{

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

{

**int** m, n, p, q, sum = 0, c, d, k;

Scanner in = **new** Scanner(System.in);

System.out.println("Enter the number of rows and columns of first matrix");

m = in.nextInt();

n = in.nextInt();

**int** first[][] = **new** **int**[m][n];

System.out.println("Enter the elements of first matrix");

**for** ( c = 0 ; c < m ; c++ )

**for** ( d = 0 ; d < n ; d++ )

first[c][d] = in.nextInt();

System.out.println("Enter the number of rows and columns of second matrix");

p = in.nextInt();

q = in.nextInt();

**if** ( n != p )

System.out.println("Matrices with entered orders can't be multiplied with each other.");

**else**

{

**int** second[][] = **new** **int**[p][q];

**int** multiply[][] = **new** **int**[m][q];

System.out.println("Enter the elements of second matrix");

**for** ( c = 0 ; c < p ; c++ )

**for** ( d = 0 ; d < q ; d++ )

second[c][d] = in.nextInt();

**for** ( c = 0 ; c < m ; c++ )

{

**for** ( d = 0 ; d < q ; d++ )

{

**for** ( k = 0 ; k < p ; k++ )

{

sum = sum + first[c][k]\*second[k][d];

}

multiply[c][d] = sum;

sum = 0;

}

}

System.out.println("Product of entered matrices:-");

**for** ( c = 0 ; c < m ; c++ )

{

**for** ( d = 0 ; d < q ; d++ )

System.out.print(multiply[c][d]+"**\t**");

System.out.print("**\n**");

}

}

}

}

**import** java.util.Scanner;

**class** LinearSearch

{

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

{

**int** c, n, search, array[];

Scanner in = **new** Scanner(System.in);

System.out.println("Enter number of elements");

n = in.nextInt();

array = **new** **int**[n];

System.out.println("Enter " + n + " integers");

**for** (c = 0; c < n; c++)

array[c] = in.nextInt();

System.out.println("Enter value to find");

search = in.nextInt();

**for** (c = 0; c < n; c++)

{

**if** (array[c] == search) */\* Searching element is present \*/*

{

System.out.println(search + " is present at location " + (c + 1) + ".");

**break**;

}

}

**if** (c == n) */\* Searching element is absent \*/*

System.out.println(search + " is not present in array.");

}

}

**import** java.util.Scanner;

**class** BubbleSort {

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

**int** n, c, d, swap;

Scanner in = **new** Scanner(System.in);

System.out.println("Input number of integers to sort");

n = in.nextInt();

**int** array[] = **new** **int**[n];

System.out.println("Enter " + n + " integers");

**for** (c = 0; c < n; c++)

array[c] = in.nextInt();

**for** (c = 0; c < ( n - 1 ); c++) {

**for** (d = 0; d < n - c - 1; d++) {

**if** (array[d] > array[d+1]) */\* For descending order use < \*/*

{

swap = array[d];

array[d] = array[d+1];

array[d+1] = swap;

}

}

}

System.out.println("Sorted list of numbers");

**for** (c = 0; c < n; c++)

System.out.println(array[c]);

}

}