ShreeSoft INFORMATICS

Address: Tirumala Plaza, Upendra Nagar, Cidco, Nashik.

Join us on Telegram:
https://t.me/+K3YuDgt7xyA5NTg9

Join us on WhatsApp:
https://chat.whatsapp.com/BrLM0ZoWMbpA8nXi9Mrocl

Contact: +91 8308341531

Array in Java

Introduction:

- Variable are used to store the data

```
int x = 10;
int y = 20;
int z = 30;
```

one variable is used to store one value.

What if we want to store 1000 values, we need to create 1000 variable. It is not good practice because it will decrease readability, maintainability of the program.

Let see one more another example as

Suppose if we want to read and print student details like rollno, name, marks, percentage for 1 student then we need 4 variables roll no of type int, name of type string, marks of int and percentage of type float. But if we want to read and print 100 student details then we need 400 variable (4 x 100 = 400) which is impossible to declare and handle 400 variable in program. So we can solve this problem by using concept array.

What is Array?

- Array is referenced datatype to store list of element (primitive type or object type) where all the element are of same datatype.
  In other words, an array is a fixed-size data structure that is used to hold more than one value of the same data type.
- Arrays are created dynamically during runtime in Java. Arrays are stored in contiguous memory [consecutive memory locations].
- Arrays are always created in HEAP.
- The variables in the array are ordered, and each has an index beginning with 0 and end with size-1.
- In C/C++, we can find length of array using **sizeof** operator. But in java, array are object, so we can find their length using the object property **length.**
- Java array can also be used as a static field, a local variable, or a method parameter.
- The size of an array must be specified by int or short value and not long.
- The direct superclass of an array type is Object.
- Every array type implements the interfaces Cloneable and java.io.Serializable.
- This storage of arrays helps us randomly access the elements of an array [Support Random Access].
- The size of the array cannot be altered (once initialized). However, an array reference can be made to point to another array.

- The C compiler doesn't check if the program is accessing a valid array index, while in Java, JVM does that and throws ArrayIndexOutOfBoundException, if the program tries to access an invalid array index.

Advantages of Arrays:

1. The main advantage of array-based structure is that it organizes data in such a way that it can be easily manipulated.
2. We can access randomly nth element in an array at a constant time.
3. Arrays generally use less space because no variable needs to support navigation.
4. It makes the code optimized, we can retrieve or sort the data easily.

Disadvantages of Array:

1. The size of the array is fixed, which cannot be increased or decreased later. To solve this problem, a collection framework is used in java.
2. The shortage and wastage problem occurs.
3. It can be allowed only similar type elements
4. Insertion and deletion operations requires shifting of elements.

Real-time Applications of Arrays:

1. **Signal Processing:** Arrays are used in signal processing to represent a set of samples that are collected over time. This can be used in applications such as speech recognition, image processing, and radar systems.
2. **Multimedia Applications:** Arrays are used in multimedia applications such as video and audio processing, where they are used to store the pixel or audio samples. For example, an array can be used to store the RGB values of an image.
3. **Data Mining:** Arrays are used in data mining applications to represent large datasets. This allows for efficient data access and processing, which is important in real-time applications.
4. **Robotics**: Arrays are used in robotics to represent the position and orientation of objects in 3D space. This can be used in applications such as motion planning and object recognition.
5. **Real-time Monitoring and Control Systems:** Arrays are used in real-time monitoring and control systems to store sensor data and control signals. This allows for real-time processing and decision-making, which is important in applications such as industrial automation and aerospace systems.
6. **Financial Analysis:** Arrays are used in financial analysis to store historical stock prices and other financial data. This allows for efficient data access and analysis, which is important in real-time trading systems.
7. **Scientific Computing:** Arrays are used in scientific computing to represent numerical data, such as measurements from experiments and simulations. This allows for efficient data processing and visualization, which is important in real-time scientific analysis and experimentation.
8. Arrays can be used to implement many CPU Scheduling techniques.

Types of Array:

There are two types of array as

1) One Dimensional Array
2) Two or Multidimensional Array
3) Three Dimensional Array

One Dimensional Array:

Definition:

One dimensional array is single list of element where all the element are of same datatype.

Syntax to declare 1D array:

Datatype arrayname[]; OR Datatype[] arrayname;

Where

Datatype represent the type of value stored in array

arrayname represent the name of array.

Example:

int x[];

Note: In java, we don't need to specify size of an array because memory not allocated for array at the time declaration.

Creating an Array:

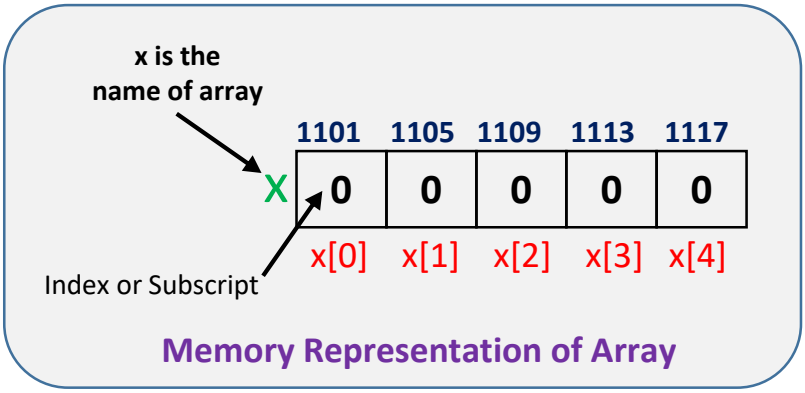
- Unlike declaration of the primitive data type variable, the declaration of an array variable does not create any space in memory for the array. It creates only a memory space for the reference to an array.
- If an array variable does not have a reference to an array, the value of the variable is null. We cannot assign values or data to an array unless it has already been created.
- In Java, an array is an object. Every object in java belongs to a class, so we can create an array object by using a new keyword and assign its reference to the variable.

  Syntax:

  arrayname = new datatype[size];

  Example:

  x = new int[5];



Memory Representation of Array

- Declaring an array variable, creating an array, and assigning the array object reference to the variable can be combined in one statement like this:

Syntax:

datatype[ ] arrayname = new datatype[arraySize];

OR

datatype arrayname[ ] = new datatype[arraySize];

Example:

int x[] = new int[5];

Some cases remember to creation of array

Case 1: At the time of array creation compulsory we should specify the size otherwise we will get compile time error.

Example:

```
int[] x=new int[3]; // valid
int[] x=new int[]; //C.E: array dimension missing
```

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:
Variable must provide either dimension expressions or an array initializer
at com.shreesoft.ArrayExample.ArrayIntialization.main(ArrayIntialization.java:6)
```

Case 2: It is legal to have an array with size zero in java.

Example:

```
int[] x=new int[0];
System.out.println(x.length); // 0
```

Case 3: If we are taking array size with negative int value then we will get runtime exception saying NegativeArraySizeException.

Example:

```
int[] a=new int[-3]; //R.E:NegativeArraySizeException
```

Case 4: The allowed data types to specify array size are byte, short, char, int. By mistake if we are using any other type we will get compile time error.

Example:

```
int[] a=new int['a']; // valid

byte b=10;
int[] a=new int[b]; // valid

short s=20;
int[] a=new int[s]; // valid

int[] a=new int[10L]; // C.E: possible loss of precision //(invalid)
int[] a=new int[10.5]; //C.E: possible loss of precision //(invalid)
```

Case 5: The maximum allowed array size in java is maximum value of int size [2147483647].

Example:

```
int[] a1=new int[2147483647]; //valid
int[] a2=new int[2147483648]; // C.E: integer number too large: 2147483648(invalid)
```

In the first case we may get RE : OutOfMemoryError.

Initialization of Array:

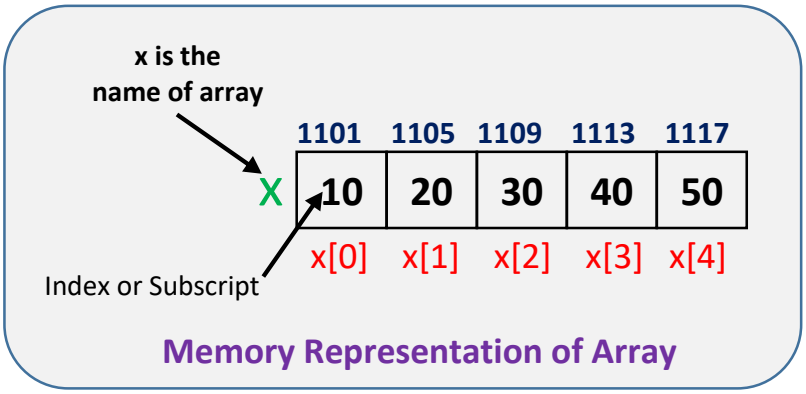
As we can see in variable concept, we can assign value to variable using assignment operator is called as initialization of variable.
Similarly we can also assign some values to array. There are two way we can initialize array as

1) After array creation:

Example:

```
int x[] = new int[5];

x[0] = 10;
x[1] = 20;
x[2] = 30;
x[3] = 40;
x[4] = 50;
```



Memory Representation of Array

2) At the time of array declaration or creation

Example 1: We can create primitive int type array x as

```
int[] x = {10,20,30,40,50};
OR
int[] x = new int[] {10,20,30,40,50};
```

Example 2: We can also create array for referenced datatype like String.

```
String[] empnames = {“Manoj”, “Pramod”, “Om”, “Varad”};
OR
String[] empnames = {new String(“Manoj”), new String(“Pramod”), new String(“Om”), new String(“Varad”)};
```

Example 3: We can also create array for userdefined class objects.

```
class Employee {
    int id;
    String ename;
    Employee(int id, String ename) {
        this.id = id;
        this.ename = ename;
    }
}
public class ReferencedObjectArray {
    public static void main(String[] args) {
        Employee[] listEmployees = {new Employee(11,"Manoj"), new Employee(12,"Varad")};

        System.out.println("---------First Employee---------");
        System.out.println("ID:"+listEmployees[0].id);
        System.out.println("Name:"+listEmployees[0].ename);

        System.out.println("\n---------Second Employee---------");
        System.out.println("ID:"+listEmployees[1].id);
        System.out.println("Name:"+listEmployees[1].ename);
    }
```

}

1. Write a program to read n element in an array and print it in forward and backward direction.
2. Write a program to read n element in an array and calculate sum and average.
3. Write a program to read n element in an array and count no. of even and odd number.
4. Write a program to read n element in an array and count no. of positive and negative number.
5. Write a program to read n element in an array and find maximum and minimum from array.
6. Write a program to read n element in an array and insert given new element at given position.
7. Write a program to read n element in an array and delete the given element from array.
8. Write a program to read n element in an array and search the given element in array or not.
9. Write a program to read n element in an array and sort it then print it.
10. Write a program to find duplicate elements in an array.
11. Write a program to find second largest element in an array of integers.
12. Write a program to check the equality of two arrays.
13. Write a program to find the union and intersection of two arrays.
14. Write a program to find missing number in an array.
15. Write a program to reverse an array without using an additional array.
16. Write a program to separate even and odd element in two different array from original array.
17. Write a program to merge two different array in to one array.
18. Write a program to cyclically rotate an array by one.
19. Java program to move all zero at the end of the array

Passing One Dimensional arrays to methods:

- Just like we can pass primitive type values and objects to methods,
- it is also possible to pass arrays and individual array elements as arguments to methods and returns arrays from methods.
- When we pass an array argument to a method in java, actually, the reference of the array is passed to the method, not value.
- Once an array is passed as an argument to a method, all elements of the array can be accessed by statements of the method.

Example Program:

20) Write a program to create method readArray() to read n element in an array and also create printArray() method to print array element on screen.
21) Write a program to create method insertElement () to insert the given element at given position in an array.
22) Write a program to create method deleteElement () to delete the element at given position in an array.
23) Write a program to create boolean method searchElement () to search the element in given array. If found return true otherwise return false.
24) Write a program to create method sortArray() to sort the list of element of given array.
25) Write a program to create method mergeArray() to merge two arrays.

Two Dimensional Array:

Definition:

Two dimensional array is a multiple list of element where all the element are of same datatype.

Syntax to declare 2D array:

Datatype arrayname[][]; OR Datatype[][] arrayname;

Where

Datatype represent the type of value stored in array

arrayname represent the name of array.

Example:

int x[][];

Creating an Array: In 2D array also for allocating memory, we need to use new operator.

Syntax:

arrayname = new datatype[size1][size2];

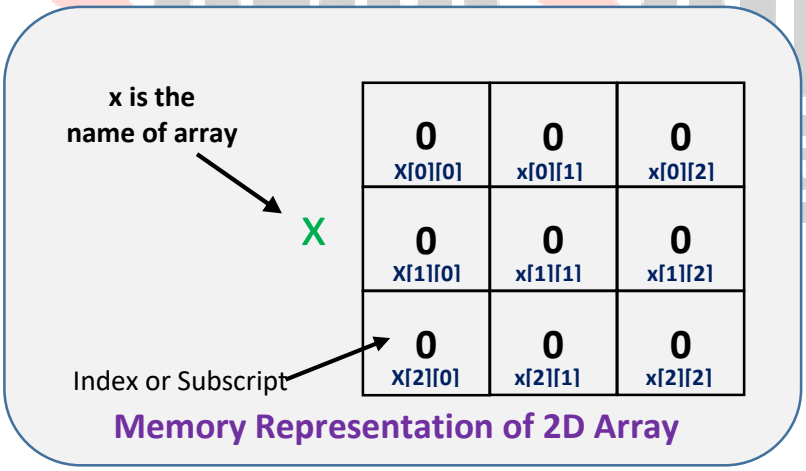
where,

size1 → It represent the no. of rows. i.e. No. of list

size2 → It represent the no. of column in each row. i.e. no. of column in each list.

Example:

x = new int[3][3];



Memory Representation of 2D Array

- Declaring an 2D array, creating an array, and assigning the array object reference to the variable can be combined in one statement like this:

Syntax:

datatype[ ][ ] arrayname = new datatype[Size1][Size2];

OR

datatype arrayname[ ][ ] = new datatype[Size1][Size2];

Example:

int x[][] = new int[3][3];

Initialization of 2D Array:

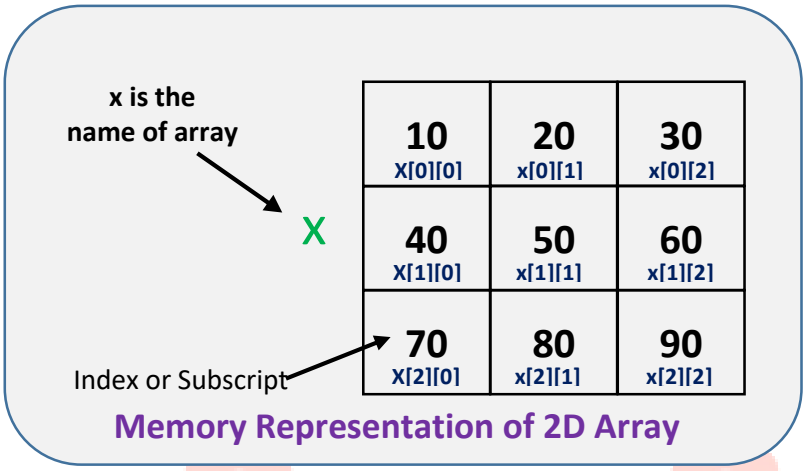
As we can see in One Dimensional array, we can assign values to array using assignment operator and curly braces is called as initialization of variable.

Similarly we can also assign some values to 2D array. There are two way we can initialize 2D array as

1) At the time of Declaration or creation:

Example:

```
int x[][] = {{10,20,30},{40,50,60},{70,80,90}};
```



Memory Representation of 2D Array

2) After the array declaration or creation:

Example:

```
int[][] x = new int[3][3];
```

```
int[0][0] = 10;
int[0][1] = 20;
int[0][2] = 30;

int[1][0] = 40;
int[1][1] = 50;
int[1][2] = 60;

int[2][0] = 70;
int[2][1] = 80;
int[3][2] = 90;
```

```
int i, j, k=10;
for( i=0; i<3; i++)
{
        for( j=0; j<3; j++)
        {
                x[i][j] = k;
                k = k + 10;
        }
}
```

26) Write a program to read Matrix of size 3 x 3 and print it in Matrix format.
27) Write a program to read 2 Matrix of size 3 x 3 and print Addition of Matrix.
28) Write a program to read 2 Matrix of size 3 x 3 and print Subtraction of Matrix.
29) Write a program to read 2 Matrix of size 3 x 3 and print Multiplication of Matrix.
30) Write a program to check whether a given matrix is Lower Triangular Matrix or not.
31) Write a program to check whether a given matrix is Upper Triangular Matrix or not.
32) Write a program to calculate sum of each row of given matrix and print it in each row at end.

Passing 2D Array as Parameter to Methods:

We can also passed 2D array as parameter to methods. For this we just need pass the name of array and reference of that array will be passed automatically.

33) write a program to create readMatrix() method to read the matrix of size 3 x 3 and also create printMatrix() method that will print it in matrix format.
34) Write a program to create addMatrix() method that will return addition of 2 Matrix.

Three Dimensional Array:

Three-dimensional arrays are complex for multi-dimensional arrays. A three dimensional can be defined as an array of two-dimensional arrays.

Syntax to declare to Three-dimensional array as

datatype [ ][ ][ ] arrayname;

Example: int arr[2][3][4];

Creating an Array: In 3D array also declaring array doen not mean that memory allocation happened. The memory allocation will be done by using new operator because array does not have memory allocate the compile time.

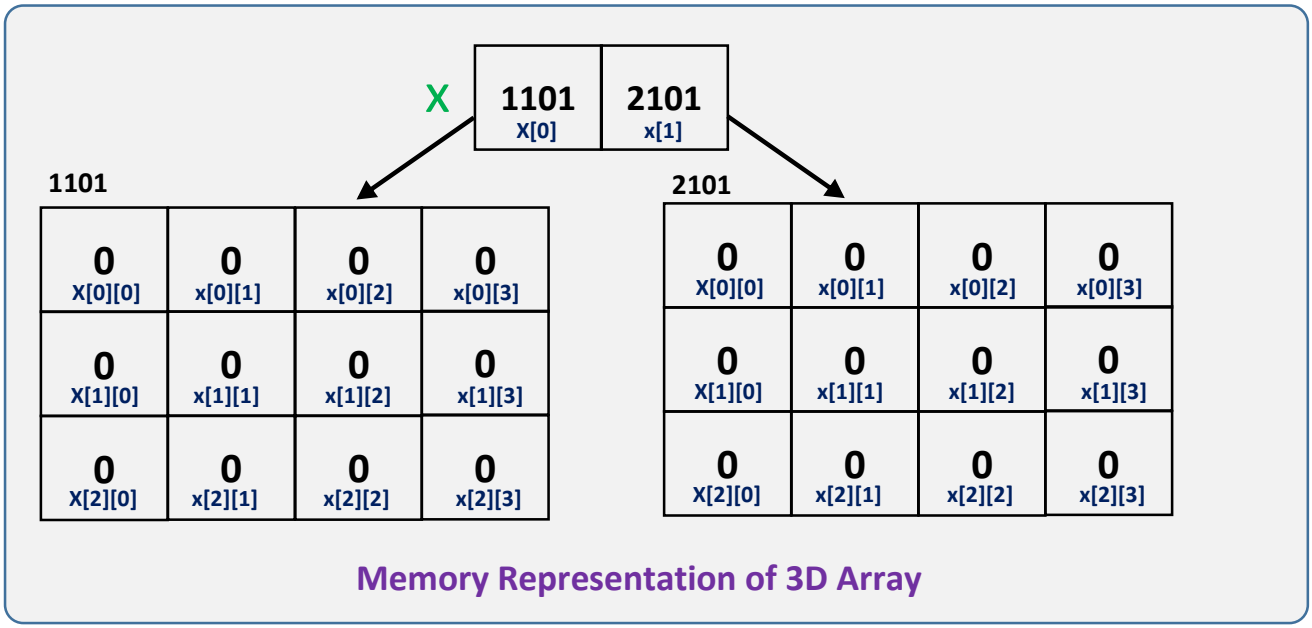
Syntax:

datatype [ ][ ][ ] arrayname = new datatype [Size1][Size2][Size3];

Size1 → The number of Tables/Arrays
Size2 → The second dimension signifies the total number of rows an array will have.
Size3 → The third dimension indicates the total columns in the 3d array.

Example:

int[ ][ ][ ] x = new int[2][3][4];

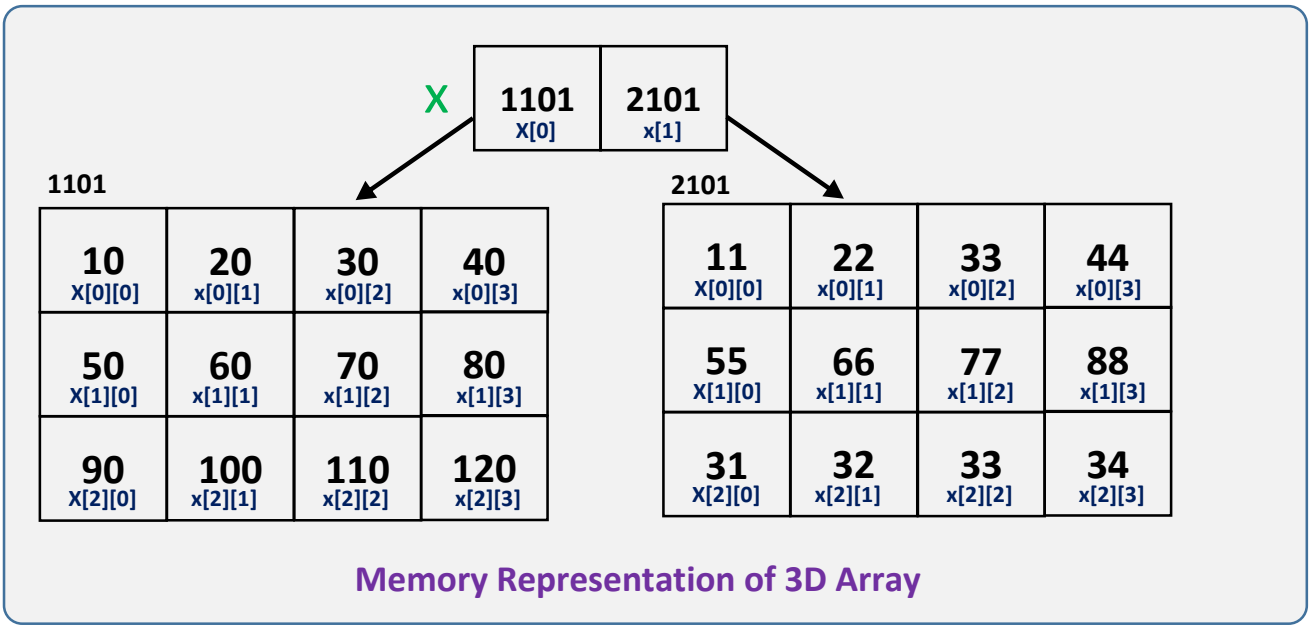


Memory Representation of 3D Array

Initialization of 3D Array:

1) At the time of array declaration or creation

```
int arr[][][] = {{{10,20,30,40},{50,60,70,80},{90,100,110,120}},{{11,22,33,44},{55,66,77,88},{31,32,33,34}}};
```



Memory Representation of 3D Array

2) After the declaration or creation of Array:

Example: **int[][][] x = new int[2][3][4];**

```
int[0][0][0] = 10;
int[0][0][1] = 20;
int[0][0][2] = 30;
int[0][0][3] = 40;

int[0][1][0] = 50;
int[0][1][1] = 60;
int[0][1][2] = 70;
int[0][1][3] = 80;

int[0][2][0] = 90;
int[0][2][1] = 100;
int[0][2][2] = 110;
int[0][2][3] = 120;
```

```
int i, j, k, m=10;
for( i=0; i<1; i++)
{
    for( j=0; j<3; j++)
    {
        for(k=0; k<4;k++)
        {
            x[i][j] = m;
            m = m + 10;
        }
    }
}
```