

**1.What do you mean by an Array?**

**=>**It refers to index collection of fixed no of homogeneous data elements. Single variable holding multiple values which improves readability of the program.

If we use a traditional approach, then to store 5 values we need to create 5 variables.   
Similarly to store 100 values we need to create 100 variables.   
The drawback in the traditional approach is that remembering the variables names is complex, so to avoid this problem we need to use “Arrays”.

**2. How to create an Array?**

**=>Array** declaration  
**1.Single Dimension Array**

**Declaration of array.**  
int[] a;//recommended to use as variable is separated from type.

* int a[]
* int []a
* int[6] a; // compile time error. we cannot specify the size.

**Array Construction**  
Every array in java is an object hence we create using a new operator.

**Example** int[] a;

a=new int[5];

or

int[] a =new int[5];

**Rules For Declaring Array**

**Rule1**   
At the time of Array construction compulsorily we should specify the size.

**Example:**   
int[] a=new int[5];

int[] a =new int[];//ce:: array dimension is missing.

**Rule2**   
It is legal to have an array with size zero.

**Example:**   
int[] a =new int[0];

System.out.println(a.length);// 0

**Rule3**   
If we declare an array with negative size it would result in a Negative Array size exception.

**Example:**   
int[] a=new int[-5]; //NegativeArraySizeException.

**Rule4**   
The allowed datatypes to specify the size are byte,short,int,char.

**Example:**   
int[] a =new int[5];

byte b=10;

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

short s=25;

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

char c='A';

int[] a=new int[c];//valid

int[] a=new int[10L];//CE

int[] a=new int[3.5f];//CE

**Rule5**  
The maximum allowed array size in java is the maximum value of int size.

int[] a=new int[2147483647]; //but valid:: OutOfMemoryError

int[] a=new int[2147483648]; //CE

**3. Can we change the size of an array at run time?**

=> Once an array has been created, its size cannot be changed. Instead, an array can only be "resized" by creating a new array with the appropriate size and copying the elements from the existing array to the new one.

String[] listOfCities = new String[3]; // array created with size 3.

listOfCities[0] = "New York";

listOfCities[1] = "London";

listOfCities[2] = "Berlin";

Suppose (for example) that a new element needs to be added to the listOfCities array defined as above. To do this, you will need to:

1. create a new array with size 4,
2. copy the existing 3 elements of the old array to the new array at offsets 0, 1 and 2, and
3. add the new element to the new array at offset 3.

**4.Can you declare an array without assigning the size of an array?**

=> Yes. We can declare an array without size but before using it needs to be initialized.

**Example**

|  |
| --- |
| **int**[] myarray = {1,3,5,7}; |

In the above statement, the length of the array is determined by the number of elements. Also, as you can see there is no need to specify size. More importantly, the declaration, instantiation and the initialization of the array is done in a single statement.

**5.What is the default value of Array?**

=> java will assign the default value 0 to each element of the array in the case of an int array. Similarly, in the case of a float array, it will be 0.0 ,in the case of a boolean array, it will be false, in the case of a String and reference array the default value is null in java, and in the case of a char array, the default value is Unicode (\u0000).

**Example:**

class ArrayDemo {

public static void main(String[] args)

{

System.out.println("String array default values:");

String str[] = new String[5];

for (String s : str)

System.out.print(s + " ");

System.out.println(

"\n\nInteger array default values:");

int num[] = new int[5];

for (int val : num)

System.out.print(val + " ");

System.out.println(

"\n\nDouble array default values:");

double dnum[] = new double[5];

for (double val : dnum)

System.out.print(val + " ");

System.out.println(

"\n\nBoolean array default values:");

boolean bnum[] = new boolean[5];

for (boolean val : bnum)

System.out.print(val + " ");

System.out.println(

"\n\nReference Array default values:");

ArrayDemo ademo[] = new ArrayDemo[5];

for (ArrayDemo val : ademo)

System.out.print(val + " ");

}

}

**Output**

String array default values:

null null null null null

Integer array default values:

0 0 0 0 0

Double array default values:

0.0 0.0 0.0 0.0 0.0

Boolean array default values:

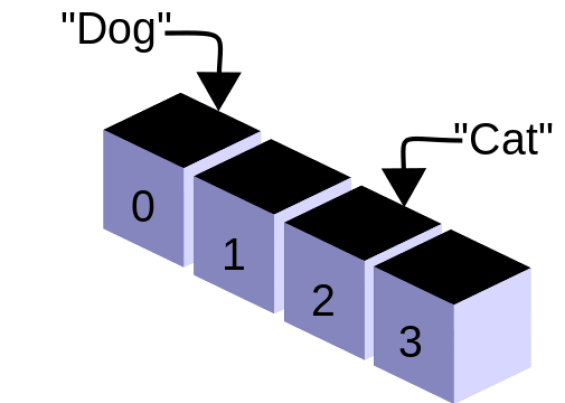
false false false false false

Reference Array default values:

null null null null null

**6. What is a 1D array with an example?**

=> 1D array or **single dimensional array** stores a list of variables of the same data type. It is possible to access each variable using the index.



In java language, int[] numbers; declares an array called numbers. Then, we can allocate memory for that array using ‘new’ keyword as follows.

numbers= new int[10];

This array is capable of storing 10 integer values.

We can combine the above two statements together and write as follows.

int numbers = new int[10];

Below is an **example** of assigning values to the array.

numbers ={1,2,3,4,5,6,7,8,9,10};

The starting index of an array is 0. Therefore, the element in the 0th index is 1. The element in the 1st index is 2. The element in the 2nd index is 3, etc. The index of the final element is 9.

If the programmer wants to store number 50 on the 2nd index, he can write the statement as follows.

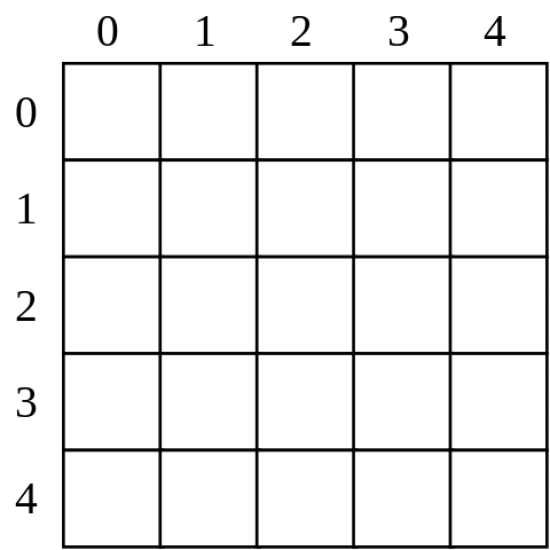
numbers[2] = 50;

One dimensional arrays can store multiple values of the same primitive type such as int, float, long, String, etc. or objects of the same class.

**7. Write a program on a 2D array?**

=> 2D array or **multi-dimensional array** stores data in a format consisting of rows and columns.

2D array or **multi-dimensional array** stores data in a format consisting of rows and columns.



For example, int[][] numbers; declares a 2D arrays.

numbers = new int [2][3];

 The above statement allocates memory for a 2D array of 2 rows and 3 columns.

We can combine the above two statements together and write the statement as follows.

int[][] numbers = new int[2][3];

Below is an **example** of assigning values to the 2D array.

int[][] numbers = { {10,20,30}, {50,60,70}};

Similar to a 1D array, the starting index of the 2D array is also 0. This array has two rows and three columns. The indexes of the rows are 0 and 1 while the indexes of columns are 0, 1 and 2. The element 10 is in the 0th row 0th column position. Number 20 is in the 0th row, 1st column position. Number 70 is in 1st row, 2nd column position.

numbers[1][2] = 50;

Above statement assigns number 50 to 1st row, 2nd column position.