

12

* It is a collection of elements of same data type.

Syntax :

`<data-type>[] <variable-name> = new <data-type>[size];`

Eg:- `int[] rollno = new int[5];`
(Cos)

```
int[] rollno = { 23, 12, 45, 66, 10};
```

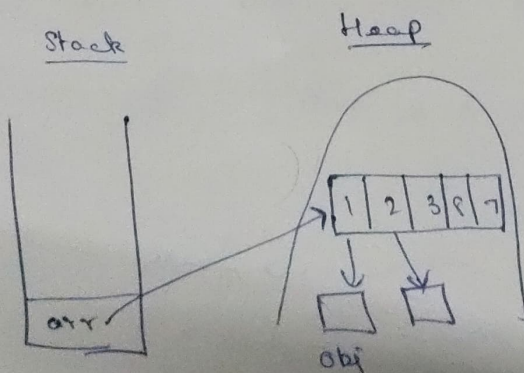
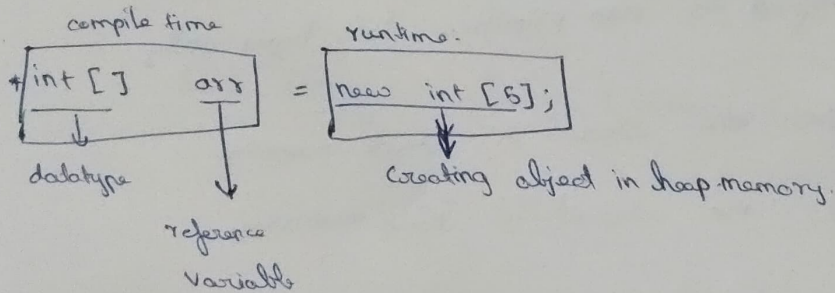
* The data-type reference what type of data stored in the array.

* All element should be of same data type.

Note :

* int [] rollno ; // It is declaration of array. rollno is getting defined in the stack.

+ roll no = ^{new}~~int~~ int[5]; // actually here object is being created in the
 ↓
 initialisation.
 memory (heap)



* In C/C++, array is continuous. But in java, there is no

- * In Java, heap objects are not continuous.

* It is Heap memory, the objects are created at runtime.

80 It is Dynamically allocate memory.

- * Index starts from 0.

Note:

int[] arr = new int [size];
 ↓
 used to
 means creating object.

* If you declared an array and didn't initialise values by default it will contain zero in integer and null in string.

* null can assigned to non primitive data types only.

+ Primitive types are stored in stack memory.

* Object, String are stored in heap memory.

* In string we didn't initialise value there will no object to point in heap memory so it is null.

* To get size of an array `<array-name>.length`

* You can iterate array using for loop.

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

* Or else you can use enhanced for loop.

```
for (int num: arr) { // for every element in array, print the  
    System.out.println(num); // here num represents element  
} // of the array.
```

* Enhanced for loop also called for Each loop.

* If you access element more than the size. it gives you ~~index~~ "ArrayIndexOutOfBoundsException" Error.

* You can also use this to print an array ~~of~~.

```
System.out.println(Arrays.toString(<array-name>));
```

35:00

* Strings are immutable (no change) and arrays are mutable when passed through function.