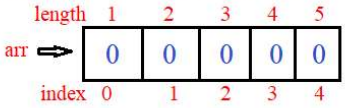
Understanding Java Array: An Array collects muliple elements of similiar datatypes in a continous block of memory

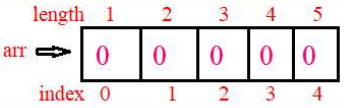
```
int arr[];
arr=new int[5];
```



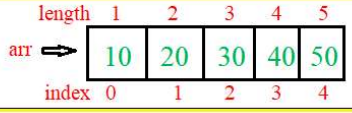
Drawbacks

1) Array length is fixed.
2) Array allows only homogeneous data
3) Array does not have any method support

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

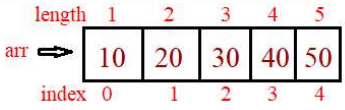



```
int arr[]={10,20,30,40,50};
```




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

Anonymous array



length:
It is a final variable which is used to get the length of an Array.

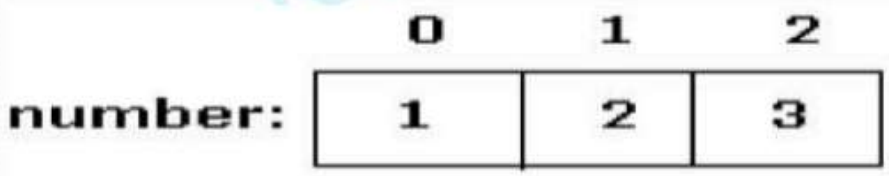
```
String empNames[]=new String[100];
syso(empNames.length); //100
syso(empNames.length()); //C.E
```

length():
It is a final **method** which is used to get the length of a String.

```
String s="Java";
syso(s.length); //C.E
syso(s.lemgth()); //4
```

Understanding Arrays

- An array is an indexed collection of fixed number of homogeneous data elements.
- An array stores multiple data items of the same data type, in a continuous block of memory, divided into a number of slots.



- The main advantage of arrays is we can represent multiple values with the same name so that readability of the code will be improved.
- The main disadvantage of arrays is its fixed length.
- It means once we created an array there is no chance of increasing or decreasing the size based on our requirement that is to use arrays compulsory, we should know the size in advance which may not possible always.
- We can resolve this problem by using **collections**.

How to declare an Array?

- To declare an array, write the data type, followed by a set of square brackets[], followed by the identifier name.

```
int []rollNumber; //valid
int rollNumber[];//valid
```

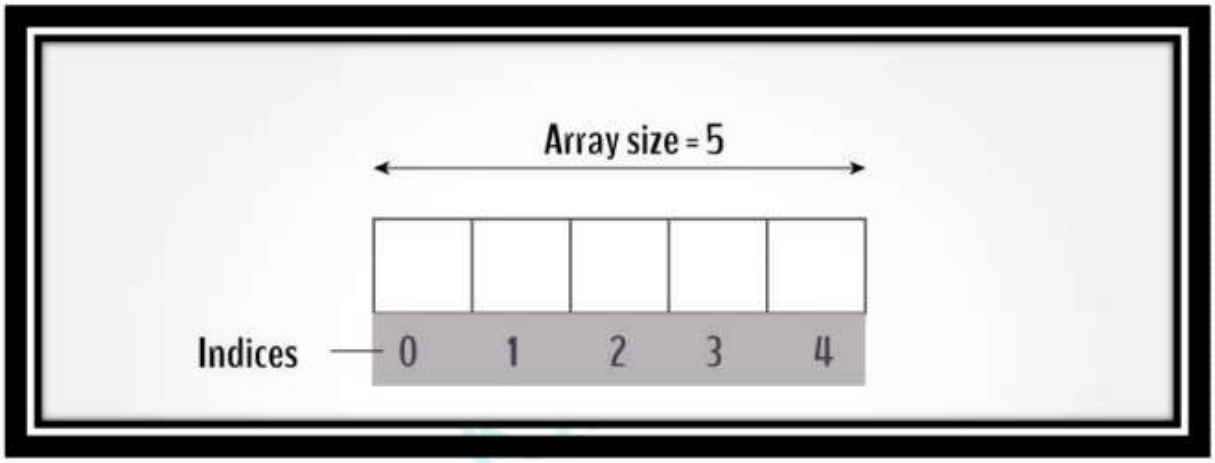
- At the time of declaration we can't specify the size of an array.

```
int []rollNumber;
int [5]rollNumber; //error
```

How to Instantiate an array?

- To instantiate (or create) an array, write the new keyword and the datatype of the array, followed by the square brackets containing the number of elements you want the array to have.
- Every array in java is an object hence we can create by using new keyword.

```
int []rollNumber;
rollNumber=new int[5]; //valid 1st way
        (or)
int []rollNumber=new int[5];// valid 2nd way
        (or)
int []rollNumber=new int[]{10,20,30,40,50};// valid 3rd way
        (or)
int []rollNumber={10,20,30,40,50}; //valid 4th way
```



- The length of an array starts with '1'
- The index position of an array starts with '0'

2 types of packages

1.pre-defined (more than 5000)

2.user-defined

Array is a class which is present in java.lang package, which is by default imported in every java program.

8 wrapper classes are also present in java.lang package.

System class is also present in java.lang package.

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

Declaration of an array `int arr[];`

Declearation means creating an arry

always go with left side of equals to

```
int arr[];
```

here type of an arry is int

name of an array is arr
(we can write any name)

[] array dimension means
array representation

if we are not writting [] it
is not consider as an arry

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

Instantiation of an array

reserving some memory
blocks or else
reserving som memory area
so we can pass the data later

We should use the new key word in instantiation

Every array is an object.

Whenever we are instantiating an array every memory of that array will be filled with the default values of the data type of that array.

Initialization means assigning some values to that array

```
int arr[]={10,20,30,40,50};
```

Index positions of an array are used to pass the values into an array and retrieve the values from an array.

length:
It is a final variable which is used to get the length of an Array.

```
String empNames[]=new String[100];
syso(empNames.length); //100
syso(empNames.length()); //C.E
```

length():
It is a final **method** which is used to get the length of a String.

```
String s="Java";
syso(s.length); //C.E
syso(s.lemgth()); //4
```

Here final variable in the sense

Here, Final is that it is fixed

```
7         System.out.println("Implementing Java Array\n");
8
9         int arr1[];
10        arr1=new int[5];
11
12        int arr2[]=new int[3];
13
14        int arr3[]= {11,22,33,44};
15
16        int arr4[]=new int[] {100,200,300,400,500,600};
17
18        String s="Java";
19
20        System.out.println("arr1 : "+arr1.length());
21        System.out.println("arr2 : "+arr2.length());
22        System.out.println("arr3 : "+arr3.length());
23        System.out.println("arr4 : "+arr4.length());
24        System.out.println("s : "+s.length);
25    }
26 }
```

```
    {
        System.out.println("Implementing Java Array\n");

        int arr1[];
        arr1=new int[5];

        int arr2[]=new int[3];

        int arr3[]= {11,22,33,44};

        int arr4[]=new int[] {100,200,300,400,500,600};

        String s="Java";

        System.out.println("arr1 : "+arr1.length);
        System.out.println("arr2 : "+arr2.length);
        System.out.println("arr3 : "+arr3.length);
        System.out.println("arr4 : "+arr4.length);
        System.out.println("s : "+s.length());
    }
    public static void main(String[] args)
    {
        new ClassA().meth1();
    }
}
```

```
Implementing Java Array

arr1 : 5
arr2 : 3
arr3 : 4
arr4 : 6
s : 4
```

```
package com.pack1;

public class ClassB
{
    public static void main(String[] args)
    {
        ClassB bobj=new ClassB();
        System.out.println(bobj);
    }
}
```

```
com.pack1.ClassB@3feba861
```

when ever you are going to print object the compiler internallay goes to toString method of object class

```
package com.pack1;

public class ClassB
{
    public static void main(String[] args)
    {
        ClassB bobj=new ClassB();
        System.out.println(bobj);
        System.out.println(bobj.toString());
    }
}
```

```
com.pack1.ClassB@3feba861
com.pack1.ClassB@3feba861
```

```
8
9       int arr1[];
10      arr1=new int[5];
11
12      int arr2[]=new int[3];
13
14      int arr3[]= {11,22,33,44};
15
16      int arr4[]=new int[] {100,200,300,400,500,600};
17
18      String s="Java";
19
20      System.out.println("arr1 : "+arr1.length);
21      System.out.println("arr2 : "+arr2.length);
22      System.out.println("arr3 : "+arr3.length);
23      System.out.println("arr4 : "+arr4.length);
24      System.out.println("s : "+s.length());
25
26      System.out.println("\narr1 : "+arr1);
27      System.out.println("arr2 : "+arr2);
28      System.out.println("arr3 : "+arr3);
29      System.out.println("arr4 : "+arr4);
30
31
32  }
```

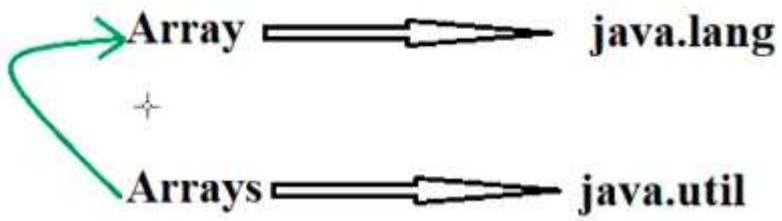
```
Implementing Java Array

arr1 : 5
arr2 : 3
arr3 : 4
arr4 : 6
s : 4

arr1 : [I@12edcd21
arr2 : [I@52cc8049
arr3 : [I@5b6f7412
arr4 : [I@27973e9b
```

here arr1 mean objct ,means when ever you are creating an array mean you are creating an object

As array have no method support

Arrays class will have methods to manupliate the data in **array**.

So to manupliate the data in array , we need to take care of the arrays class which is present in **java.util package** need to **import**



We are actually calling toString() from arrays class. Means we are inherit toString() from object class.

Here method is calling directly with the help of the class name is static method.

Here toString() is a static method which is parameterized

Here toString() is going to print all the elements in arr1.

```
System.out.println("\narr1 : "+Arrays.toString(arr1)); //[0,0,0,0,0]
```

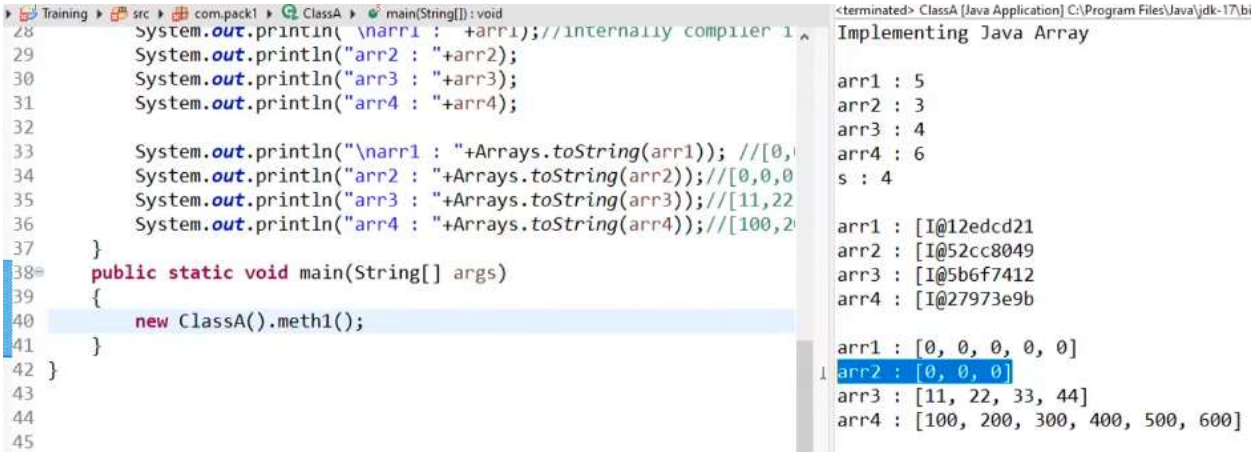
In the above line arrays class is actually overriding the toString method

What is meant by overriding?

This is actually one of the OOPS concepts, this is actually polymorphism, method name will be same implementation will be different.

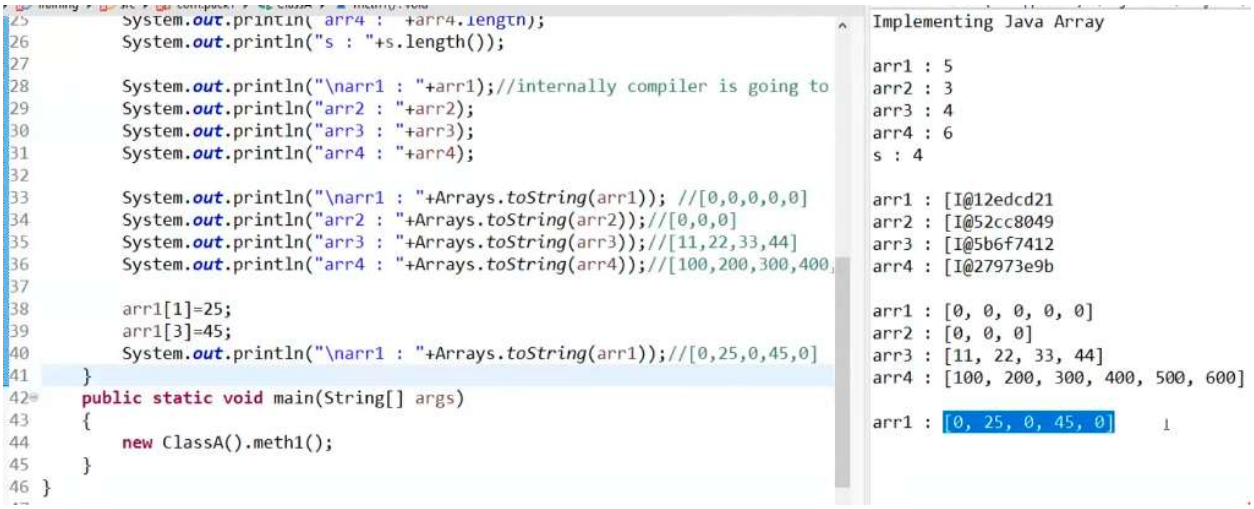
Basically, polymorphism means it is an ability to do different, different tasks with the same identity.

Need to import java.util package to use arrays class



```
Training ▸ src ▸ com.pack1 ▸ ClassA ▸ main(String[]) : void

        System.out.println("\narr1 : "+arr1);//internally compiler i
        System.out.println("arr2 : "+arr2);
        System.out.println("arr3 : "+arr3);
        System.out.println("arr4 : "+arr4);

        System.out.println("\narr1 : "+Arrays.toString(arr1)); //[0,
        System.out.println("arr2 : "+Arrays.toString(arr2));//[0,0,0
        System.out.println("arr3 : "+Arrays.toString(arr3));//[11,22
        System.out.println("arr4 : "+Arrays.toString(arr4));//[100,2
    }
    public static void main(String[] args)
    {
        new ClassA().meth1();
    }
}
```

```
<terminated> ClassA [Java Application] C:\Program Files\Java\jdk-17\bi
Implementing Java Array

arr1 : 5
arr2 : 3
arr3 : 4
arr4 : 6
s : 4

arr1 : [I@12edcd21
arr2 : [I@52cc8049
arr3 : [I@5b6f7412
arr4 : [I@27973e9b

arr1 : [0, 0, 0, 0, 0]
arr2 : [0, 0, 0]
arr3 : [11, 22, 33, 44]
arr4 : [100, 200, 300, 400, 500, 600]
```



```
        System.out.println("arr4 : "+arr4.length);
        System.out.println("s : "+s.length());

        System.out.println("\narr1 : "+arr1);//internally compiler is going to
        System.out.println("arr2 : "+arr2);
        System.out.println("arr3 : "+arr3);
        System.out.println("arr4 : "+arr4);

        System.out.println("\narr1 : "+Arrays.toString(arr1)); //[0,0,0,0,0]
        System.out.println("arr2 : "+Arrays.toString(arr2));//[0,0,0]
        System.out.println("arr3 : "+Arrays.toString(arr3));//[11,22,33,44]
        System.out.println("arr4 : "+Arrays.toString(arr4));//[100,200,300,400,

        arr1[1]=25;
        arr1[3]=45;
        System.out.println("\narr1 : "+Arrays.toString(arr1));//[0,25,0,45,0]
    }
    public static void main(String[] args)
    {
        new ClassA().meth1();
    }
}
```

```
Implementing Java Array

arr1 : 5
arr2 : 3
arr3 : 4
arr4 : 6
s : 4

arr1 : [I@12edcd21
arr2 : [I@52cc8049
arr3 : [I@5b6f7412
arr4 : [I@27973e9b

arr1 : [0, 0, 0, 0, 0]
arr2 : [0, 0, 0]
arr3 : [11, 22, 33, 44]
arr4 : [100, 200, 300, 400, 500, 600]

arr1 : [0, 25, 0, 45, 0]
```

```java
import java.util.Arrays;

public class ClassA
{
    void meth1()
    {
        System.out.println("Implementing Java Array\n");

        int arr1[];
        arr1=new int[5];

        int arr2[]=new int[3];

        int arr3[]= {11,22,33,44};

        int arr4[]=new int[] {100,200,300,400,500,600};

        String s="Java";

        System.out.println("arr1 : "+arr1.length);
        System.out.println("arr2 : "+arr2.length);
        System.out.println("arr3 : "+arr3.length);
        System.out.println("arr4 : "+arr4.length);
        System.out.println("s : "+s.length());

```

```java
        System.out.println("\narr1 : "+arr1);//internally compiler is going to call toString() present in Object Class
        System.out.println("arr2 : "+arr2);
        System.out.println("arr3 : "+arr3);
        System.out.println("arr4 : "+arr4);
```

```java

        System.out.println("\narr1 : "+Arrays.toString(arr1)); //[0,0,0,0,0]
        System.out.println("arr2 : "+Arrays.toString(arr2));//[0,0,0]
        System.out.println("arr3 : "+Arrays.toString(arr3));//[11,22,33,44]
        System.out.println("arr4 : "+Arrays.toString(arr4));//[100,200,300,400,500,600]

        arr1[1]=25;
        arr1[3]=45;
        System.out.println("\narr1 : "+Arrays.toString(arr1));//[0,25,0,45,0]

        System.out.println("\nReteriving the data from Array");

```

```
43
44          System.out.println(arr1[0]);
45          System.out.println(arr1[arr1.length-2]);
46          System.out.println(arr3[arr1.length-3]);
47          //System.out.println(arr4[arr4.length]); // It generates AIOB-Exception
48          System.out.println(arr4[arr4.length-1]);
49      }
50      public static void main(String[] args)
51      {
52          new ClassA().meth1();
53      }
54  }
```