Arrays

What is an array?

An array is an object that can store many values of the same type.

An array could be a list of ints or Strings or any other reference types you create.

An array store a fixed number of elements of the same type.

int[] scores

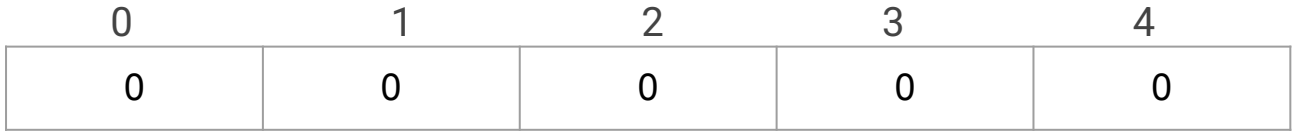
0	1	2	3	4
90	87	70	84	98

Making an array with default values

Square brackets [] with the number of elements in the array

int[] scores = new int[5];

Empty square brackets []

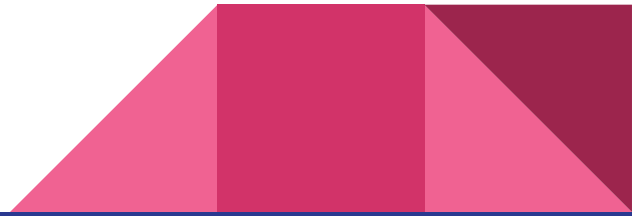


Making an array with default value

int[] scores = new int[5];

When you create an array in this manner, Java assigns default values to each item.

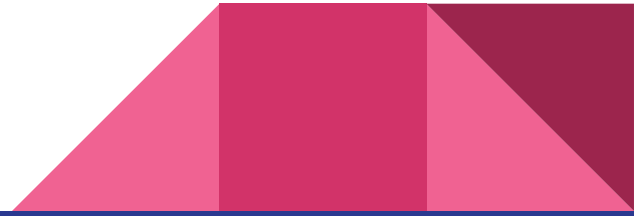
Type	Default Value
int	0
double	0.0
boolean	false
Objects n	null



Making an Array with Initial Values

int[] scores = {90, 87, 79, 84, 98};

0	1	2	3	4
90	87	70	84	98



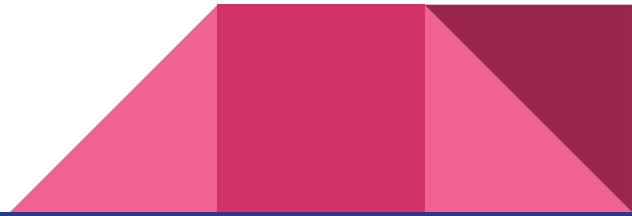
Making an array of any type

Array with default values:

Type[] variableName = new Type[numElements];

Array with initial value:

Type[] variableName = {initial values list};



Getting/Setting value at an index

int[] scores = {90, 87, 79, 84, 98};

Getting:

int myScore = score[1]; ⇒ 87

Setting:

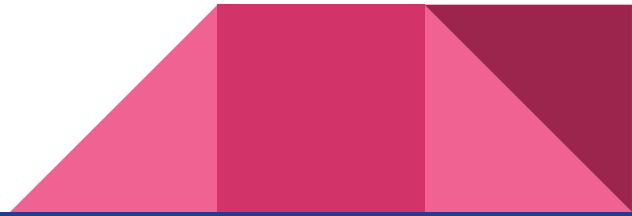
score[1] = 92 ⇒

0	1	2	3	4
90	92	79	84	98

Getting the array length

int[] scores = {90, 87, 79, 84, 98};

int len = scores.length; //no parenthesis ⇒ 5



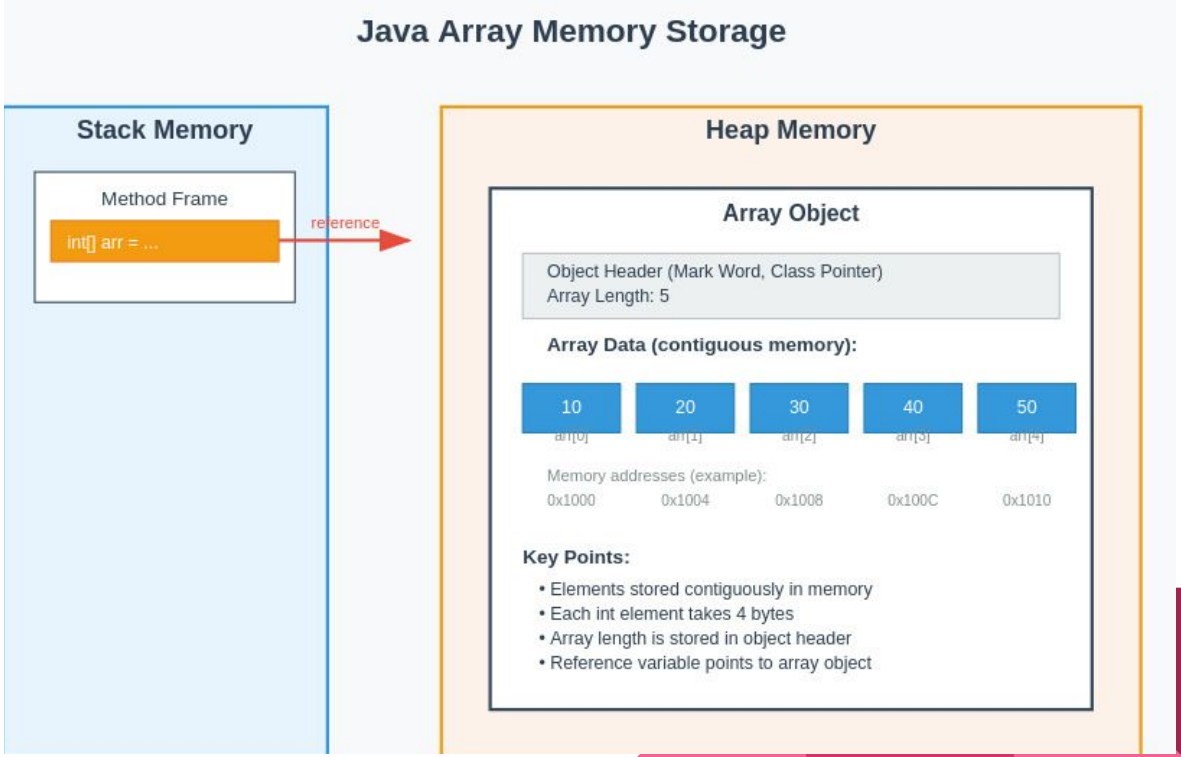
Last Index of Array

int lastIndex = array.length - 1;

Trying to access a value outside the index value list, you get an **ArrayIndexOutOfBoundsException**

Arrays are reference types

Store an address not a value.



Arrays are reference types

When you make a copy of an array, the pointer (arrow) is copied, no the content.

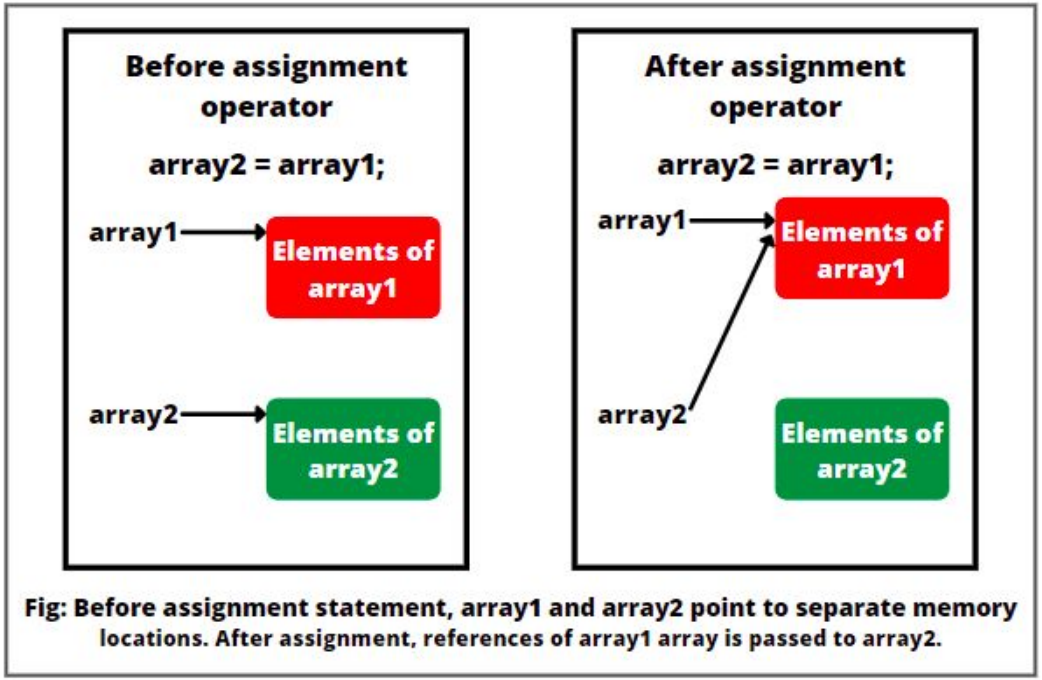


Fig: Before assignment statement, array1 and array2 point to separate memory locations. After assignment, references of array1 array is passed to array2.

Example

```
int[] arr1 = {1, 2, 3, 4, 5};

int[] arr2 = {10, 20, 30, 40, 50};

arr1 = arr2;

arr1[0] =100;
```

What is stored in arr1 and arr2 after after executing that code?

```
arr1 => { 100, 20, 30, 40, 50 }

arr2 => { 100, 20, 30, 40, 50 }
```