

Array

- What is Array ?
- Types of Array
 - 1 D array
 - Multi Dimensional Array
 - jagged Array
- Operations
 - Insertion
 - Deletion
- Searching
 - Linear Search
 - Binary Search (To Do).

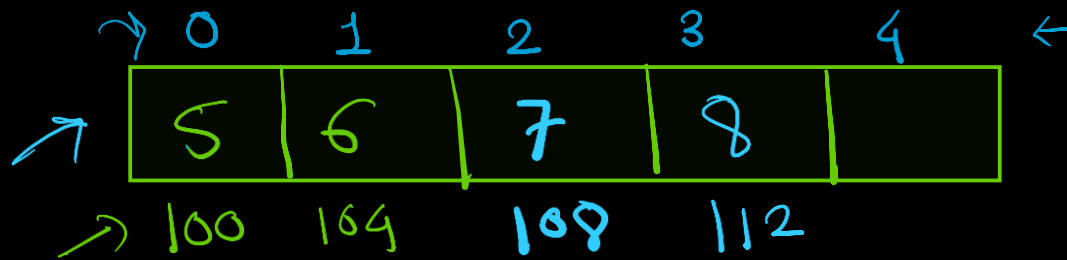
What is Array ?

1. It is a linear data structure
2. Array is collection of same data type

a. Either whole array fill with integer , float, or character we can't mix this data type in one array like $\rightarrow \text{arr} = \{1, 1.5, 0.5, 'a'\}$ this is not possible in array

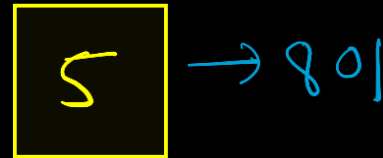
3. It store element in contiguous memory location

4. We can easily Access element in array

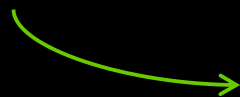


`int a = 5`

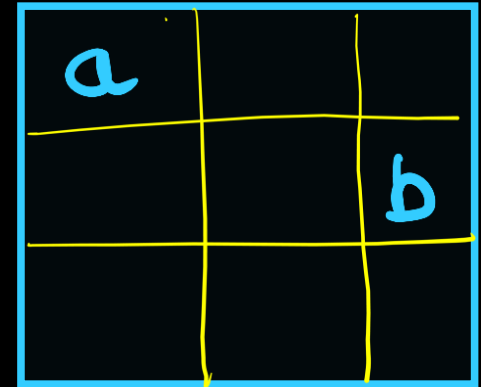
`int b = 6`

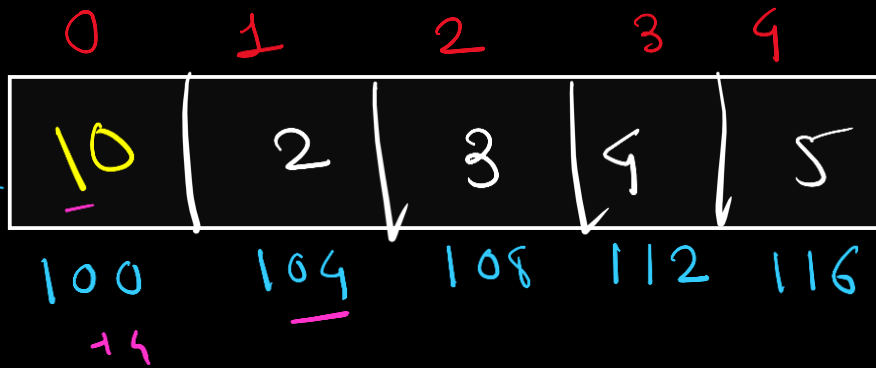


`a`

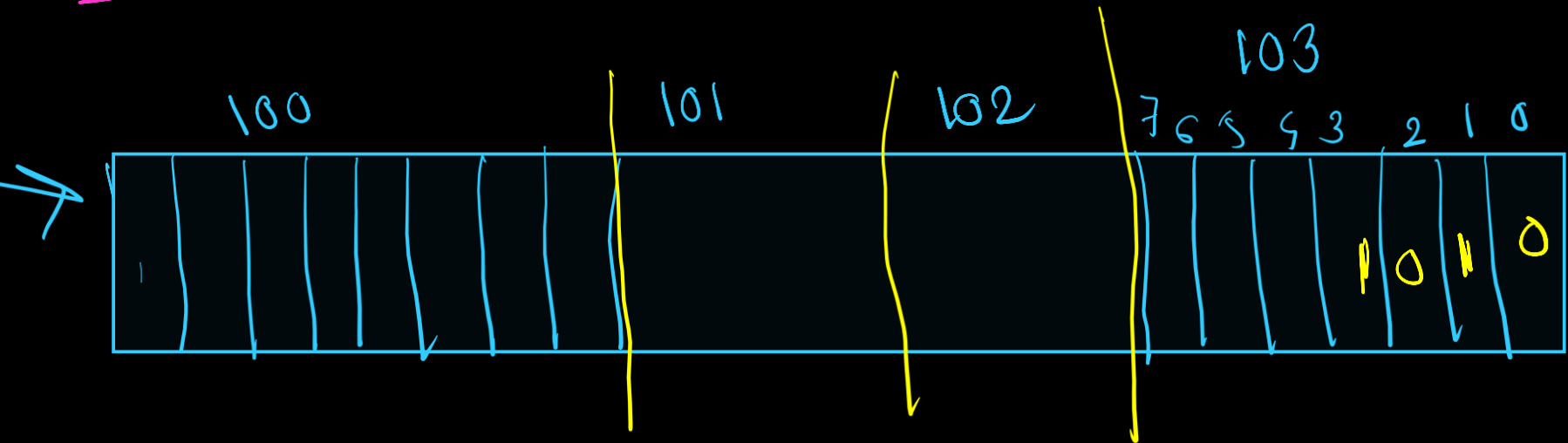


`b`



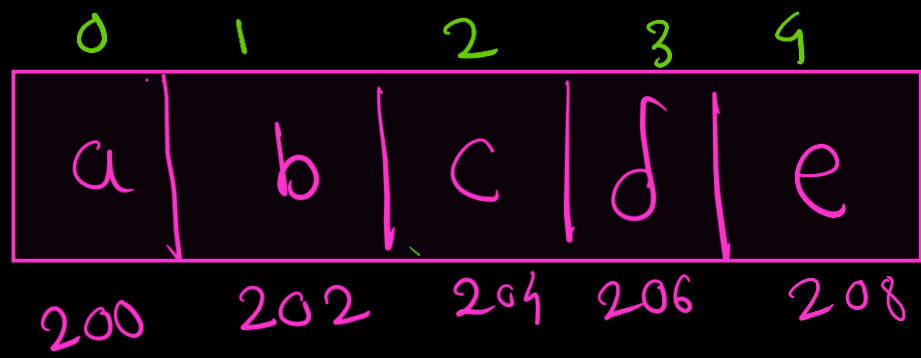


= size = 5



1 byte = 8 bits

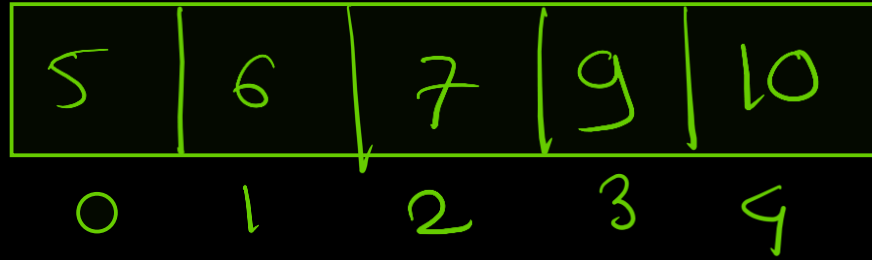
4 - 11 - = $8 * 4 = \underline{\underline{32}}$



char
arr

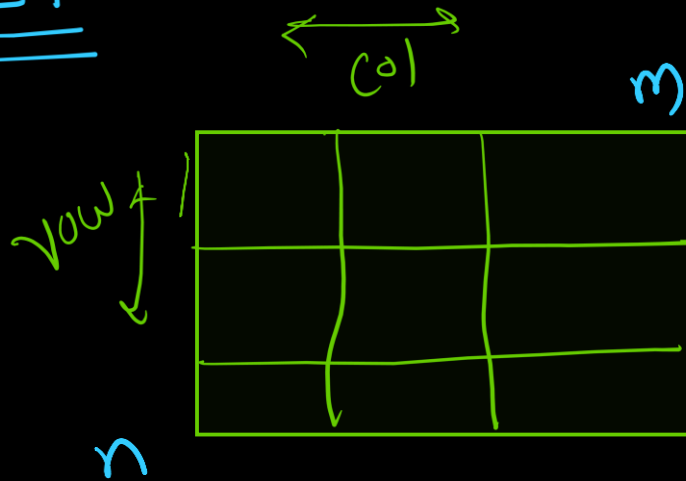
Types of Array

1. 1 D array

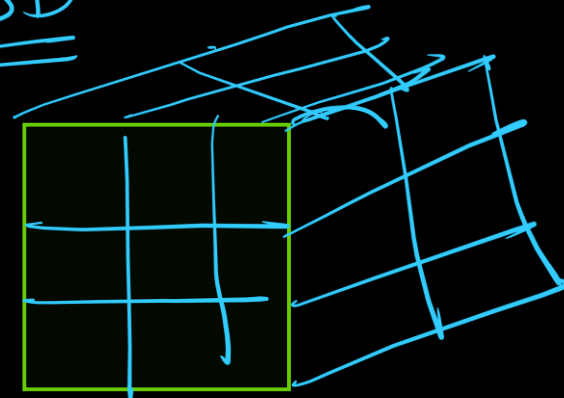


multidimensional Array

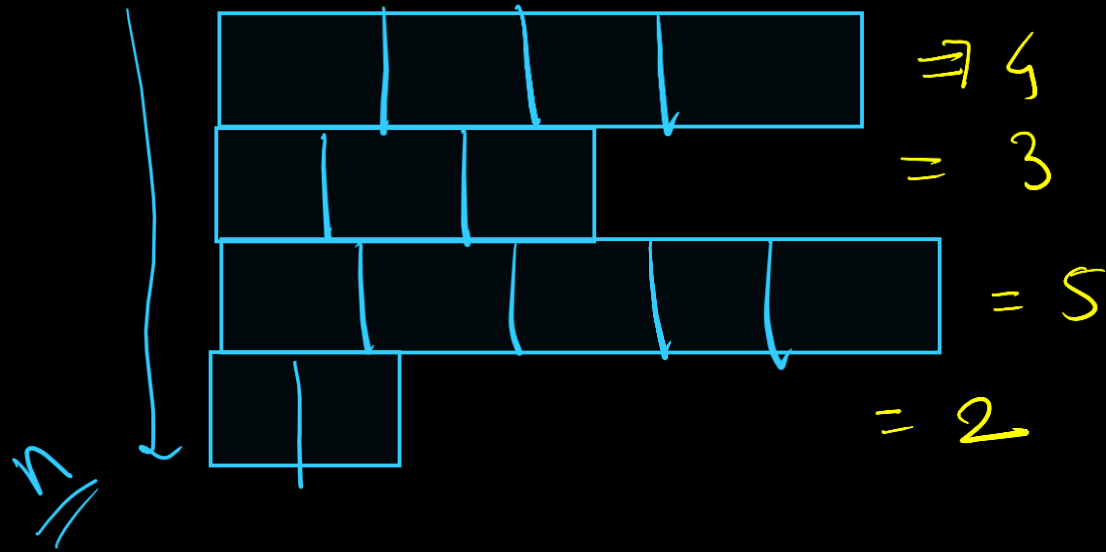
2D



3D



jagged Array



array =

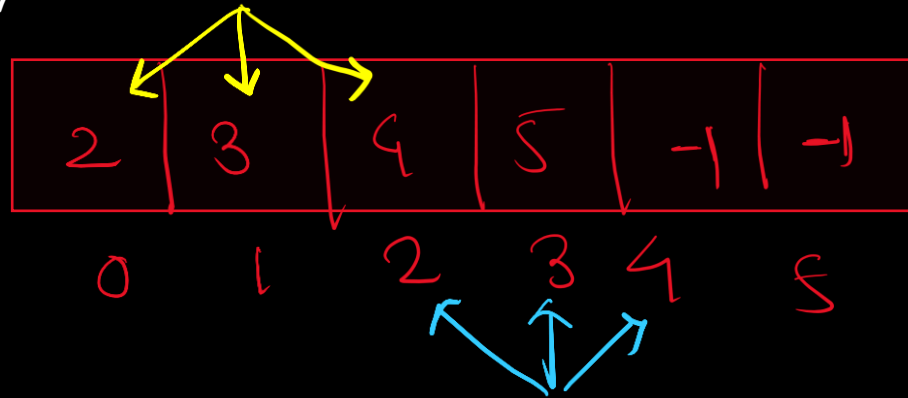
make	it	simple	java	dsa
------	----	--------	------	-----



jagged Array



Insertion in Array



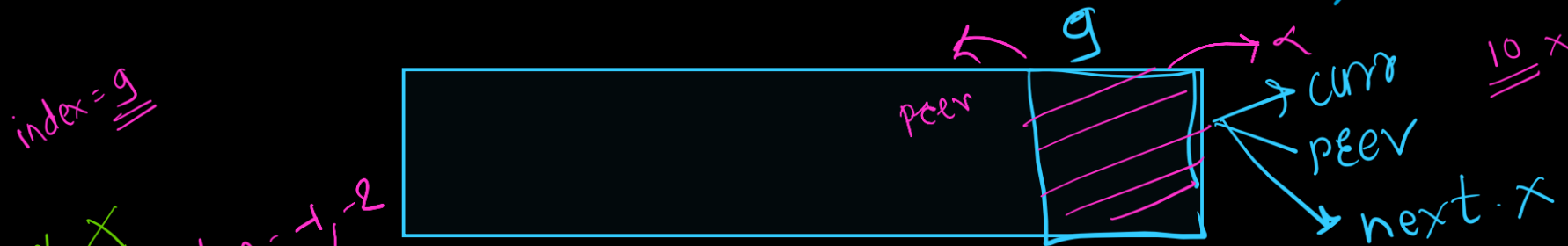
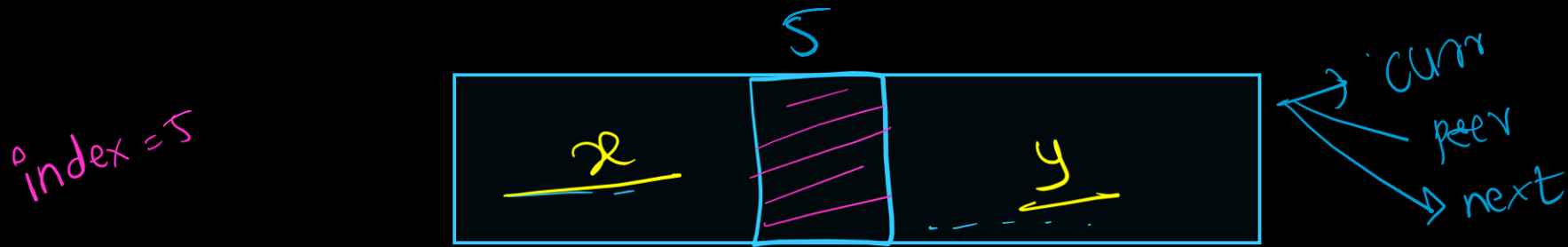
1) You have given a value.

- a) insert at that value
- b) insert after the value
- c) insert before the value

2) You have given an index.

- a) insert at that index.
- b) insert after the index.
- c) insert before the index

2) You have given an index.



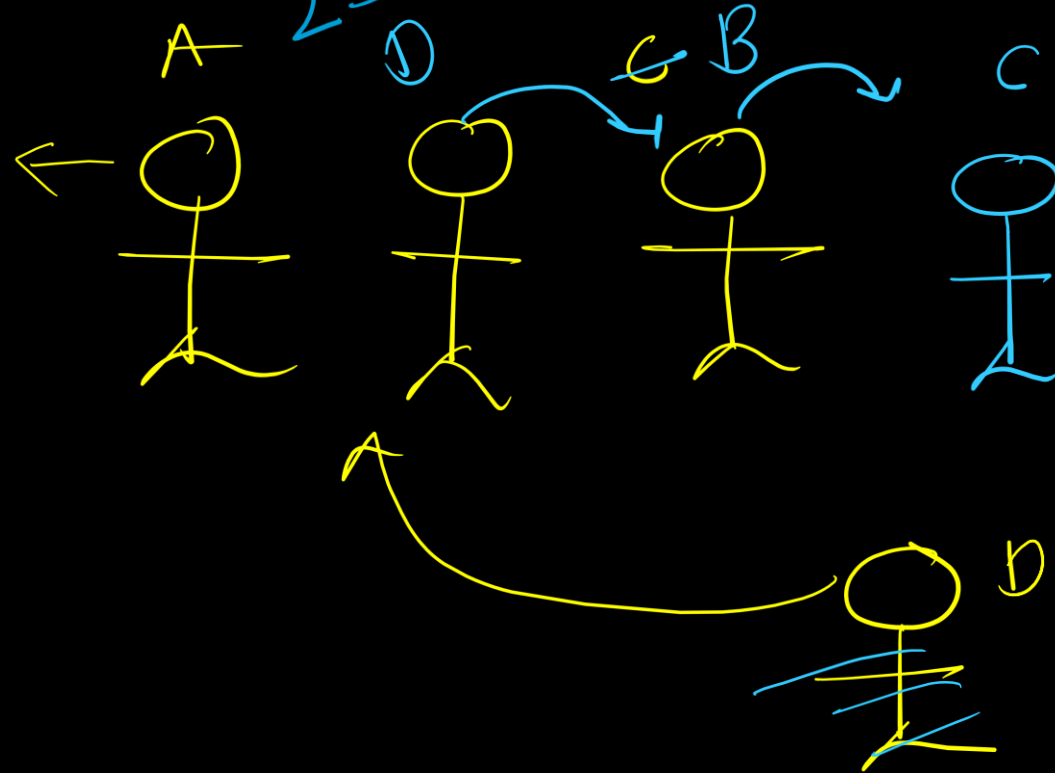
array =

1	3	4	5	<u>-1</u>	<u>-1</u>	<u>-1</u>	<u>-1</u>
0	1	2	3	4	5	6	7

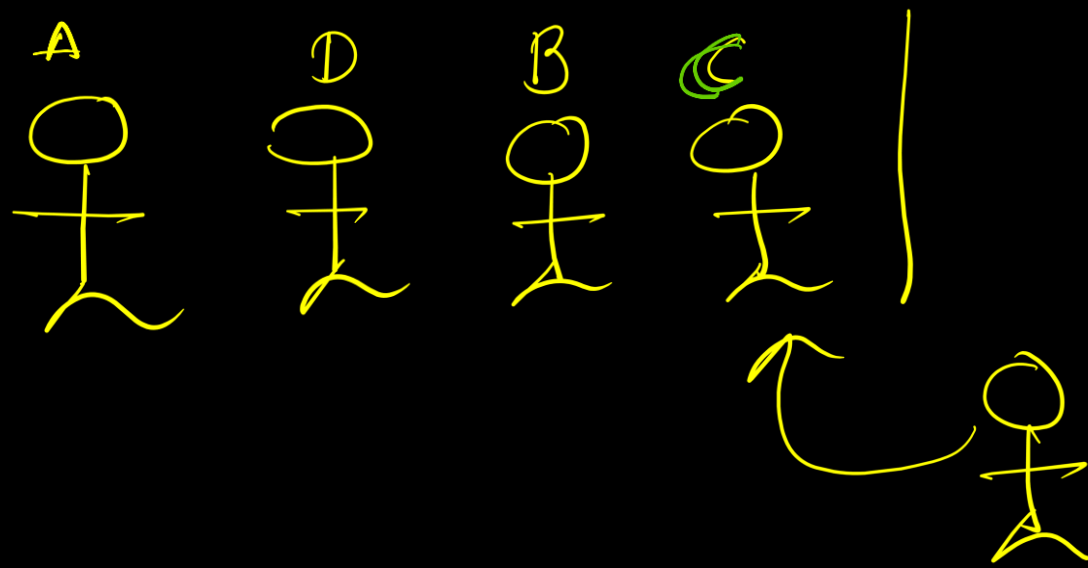
→ 8

Actual size

8



Cur size = 4



```
if (currSize < actualSize){  
    ----- perform some operatn  
}  
else{  
    we can't insert Array is full  
}
```

0	1	2	3	4	5	6	7
1	3	4	5	-1	-1	-1	1

Diagram showing an array with indices 0 to 7. The values are 1, 3, 4, 5, -1, -1, -1, 1. Arrows indicate a shift of elements from index 1 to 2, 2 to 3, and 3 to 4.

actual size = 8
 cur size = 4

Insert 2 at index 1

1	3	3	4	5	-1	-1	1
0	1	2	3	4	5	6	7

Diagram showing the array after inserting the value 2 at index 1. The values are 1, 3, 3, 4, 5, -1, -1, 1. The indices are 0 to 7.

$arr[1] \rightarrow arr[1+1]$
 $arr[2] \rightarrow arr[2+1]$
 $arr[3] \rightarrow arr[3+1]$

Assign

$\rightarrow arr[1+1] = arr[1]$
 $arr[2+1] = arr[2]$
 $arr[3+1] = arr[3]$

arr[index] = value
↓

arr[1] = 2

0	1	2	3	4	5	6	7
1	2	3	4	5	-1	-1	1

$$\text{currSize} = \underline{\underline{\text{currSize} + 1}} = 4 + 1 = \underline{\underline{5}}$$

```
if (currsize < actualsize) {
```

```
    if (index < 0 || index >= actualsize) {  
        print "index is invalid";
```

```
    }
```

```
    else {
```

```
        // shift element
```

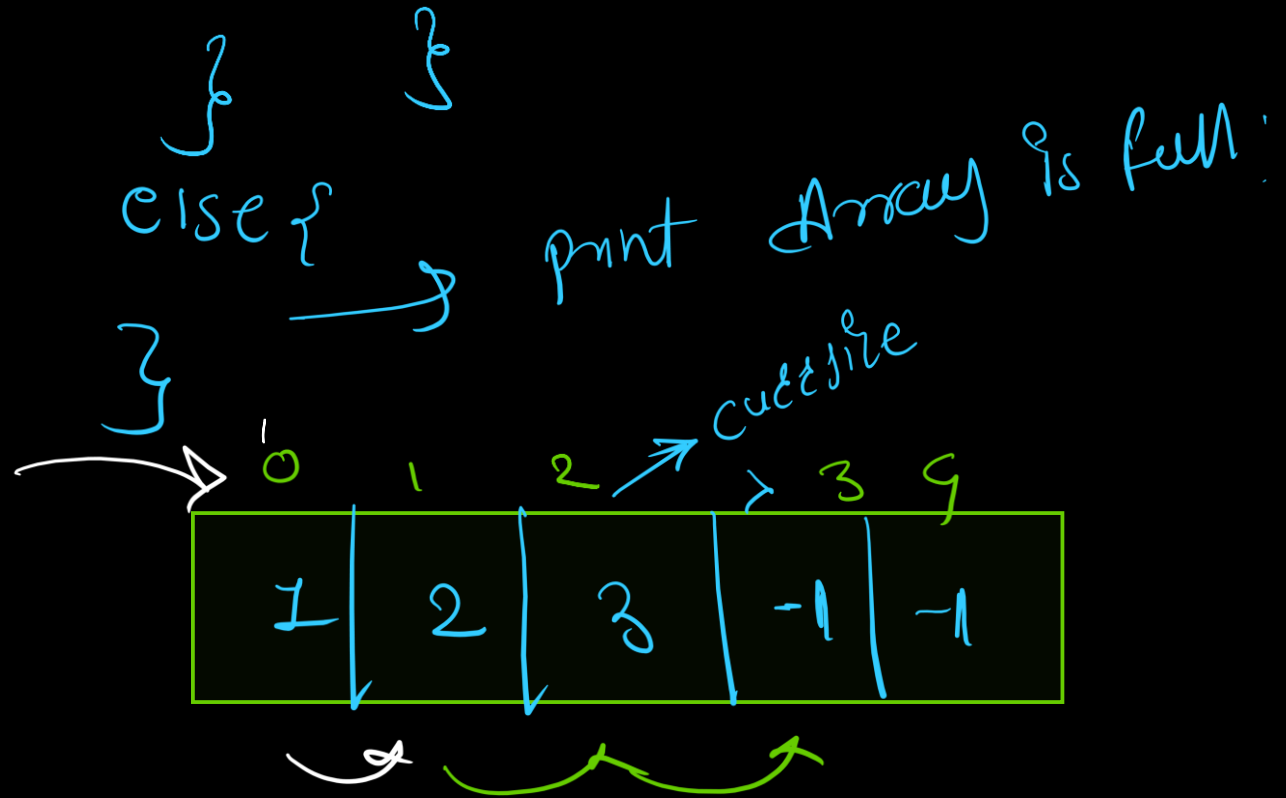
```
        for (i = currsize; i >= index; i--) {
```

```
            arr[i+1] = arr[i];
```

```
        }
```

```
        arr[index] = value;
```

```
        currsize = currsize + 1;
```



```

for (i = current; i < index; i++) {

```

```

    arr[i+1] = arr[i];

```

```

}

```

DRY RUN. →

0	1	2	3	4	5	6	7	8
1	2	4	5	6	7	8	-1	-1

$i = \text{currSize} = 6$

0	1	2	3	4	5	6	7	8
1	2	4	5	6	7	8	8	-1

$\text{arr}[7] = \text{arr}[6]$

$i = 5$

0	1	2	3	4	5	6	7	8
1	2	4	5	6	7	7	8	-1

$\text{arr}[6] = \text{arr}[5]$

$i = 4$

0	1	2	3	4	5	-	-	
1	2	4	5	6	6	7	8	-1

$\text{arr}[5] = \text{arr}[4]$

= Actual size = 9
currSize = 7.

value = 8, index = 2

$i=3$

0	1	2	3	4	5	6	7	8
1	2	4	5	5	6	7	8	-1

$$\text{arr}[4] = \text{arr}[3]$$

$i=2$

0	1	2	3	4	5	6	7	8
1	2	4	4	5	6	7	8	-1

$$\text{arr}[3] = \text{arr}[2]$$

End the loop =

$\text{arr}[\text{index}] = \text{value}$

1	2	3	4	5	6	7	8	-1
0	1	2	3	4	5	6	7	8

$$\text{currSize} = \text{currSize} + 1 = \underline{\underline{\text{currSize} = 8}}$$