Array:

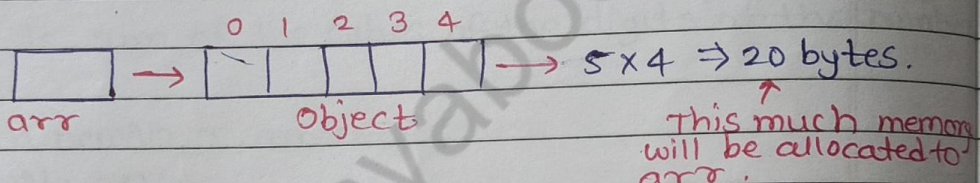
Array is collection of similar/same type of values in a single variable.

```
data-type[ ] ArrayName;
```

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

- arr → reference or Variable
- new → Allocate memory
- int [5] → object
- 5 → Length of array.

We can write array length in int, short, byte, char.



▷ Ways to make array.

1) `int [ ] marks;` // Decleration
   `marks = new int [5];` // Memory allocation

2) `int [ ] marks = new int [5];` // Decleration + memory allocation.

3) `int [ ] marks = {100, 90, 80, 70, 60};` // Decleration + Initalize

This is also valid syntax.

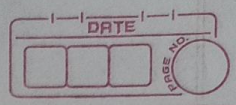
4) `int [ ] marks = new int [ ] {100, 90, 80, 70, 60};`

! Arrays index starts from 0 and goes upto (n-1) where n is length of an array.

0 to (n-1)

↑ Length of an array.

```
int[] marks = {100,60,90,80,50};
int[] arr   = {1,2,20,31,61,81};
```



▷ Accessing Array Elements:
Array elements can be accessed as follows
marks[0] = 100;
marks[1] = 60
marks[2] = 90
marks[3] = 80
marks[4] = 50

▷ Array length
Array have a length property which gives the length of the array.

marks.length; → It will give 5 bec array marks have 5 elements.

▷ Displaying an array using for loop.

```
for (int i=0; i<arr.length; i++) {
    System.out.println(arr[i]);
}
```

Array transversal.

This will print all the elements of array.

Quiz. Write a java program to print the elements of an array in reverse order.

⇒
```
for(int i = (arr.length-1); i>=0; i--){
    System.out.println(arr[i]);
}
```

inhance for loop

▷ for-each loop in Java.

Array elements can also be traversed as follows:

```
for(int element: marks)
{
    System.out.println(element);
}
```

↳ This will print all the element in array.

✱ Default Values of arrays :-

int[] arr ⇒ 0
char[] arr ⇒ " " (space)
String[] arr ⇒ null
double[] arr ⇒ 0.0
float[] arr ⇒ 0.0

! If int[] a, where a as an array if we System.out.print(a) then this will display info of array like

[I@7a81197d

one bracket bec it is an 1-d array. → int array

address in the memory.

! double x[] = {10, 20.23f, 'a', 4.89, 50}
Here implicit type casting will happen and every element will be cast to the double
x[2] ⇒ 97.00 ← Here a casted to double

! Size of array :
- We can give a size of array in byte, short, char and int.
- The size of array can't be negative.

Multidimentional Arrays.

Multidimentional Arrays are array of arrays each element of Multidimentional array is an array itself. Previous examples are of 1D array.

▲ Multidimentional 2-D arrays.

This is fix sequence [row][col];
first rows written the col are written.

A 2-D array can be created as
```
int newArr[][] = new int[2][3];
```
2 — Number of columns
↑ Number of rows

* We can add members in this array like
```
int newArr[0][0] = 101
int newArr[0][1] = 102
:
:
So on.
```

Indexing
▲ Locations of this 2-D array can be visulised as

	[0]	[1]	[2]
	Col 1	Col 2	Col 3
[0]	Row 1 (0,0)	(0,1)	(0,2)
[1]	Row 2 (1,0)	(1,1)	(1,2)

Eg. int x[ ][ ] = {{10,20,30,40,50},{60,70,80,90,100}}

← Col →

row	0	1	2	3	4
0 →	10	20	30	40	50
1 →	60	70	80	90	100

```
System.out.println(x[1][2]);
```

OP :- 80

▲ Method to print all the elements of that array.

```
for(int i=0; i<=1; i++){
  for(int j=0; j<=4; j++)
  {
    System.out.println(x[i][j]);
  }
}
```

✱ Possible syntax of multidimentional array.

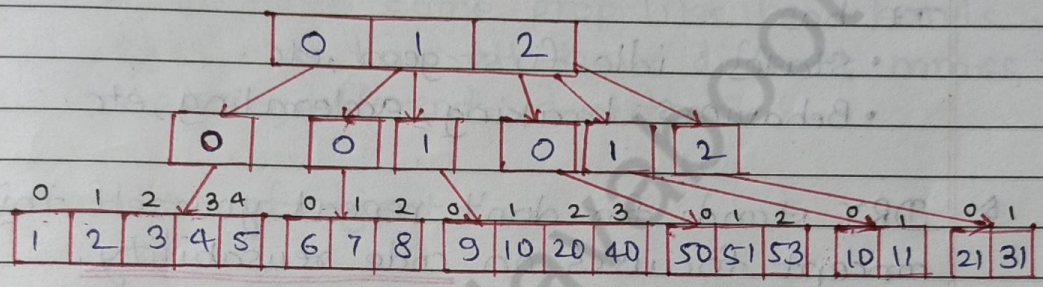
1) int x[ ][ ]
2) int [ ][ ] x
3) int [ ] x [ ]

✱ Jagged array in Java. (Array of Array)

```
int x[][][] = {
        {{10,20,30,40}},
        {{11,12,13},{45,46}},
        {{47,48,49,50}}
};
```

Length of this array will be 3.

* int x[][][] = {
{{1,2,3,4,5}}, //0
{{6,7,8},{9,10,20,40}}, //1
{{50,51,53},{10,11},{21,31}}}; //2
};



for(int i=0; i< int x[]

```
for(int i=0; i<x.length; i++)
  {
    for(int j=0; j<x[i].length; j++)
      {
        for(int k=0; k<x[i][j].length; k++)
          {
            System.out.println(x[i][j][k]);
          }
        System.out.println( );
      }
  }
```