Arrays

Arrays

- an array is a sequence of variable that can store value of one particular data type

- Eg. int Num [1, 2, 5, 6]

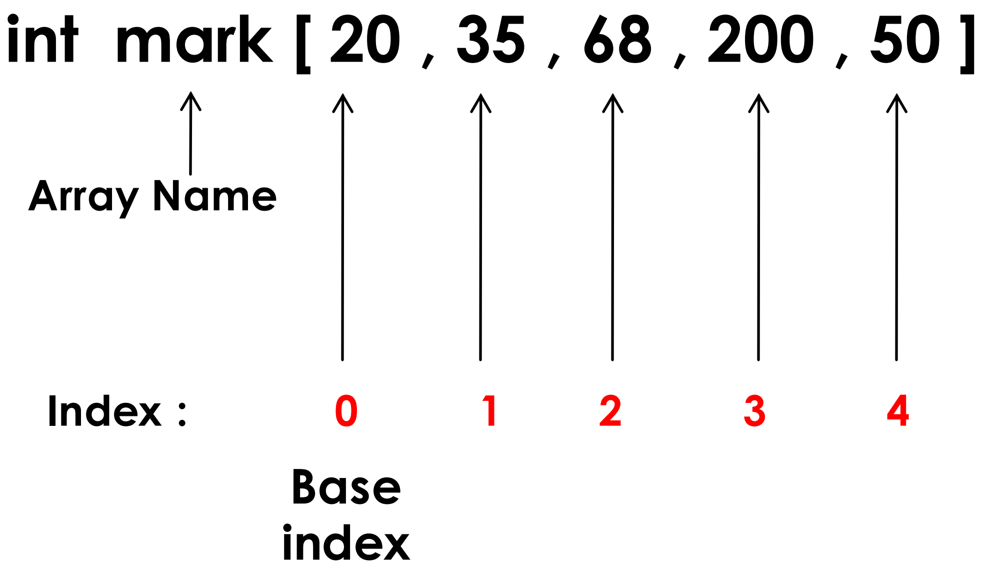
  char Name [“Arjun” , “Appu” , “vishnu”]

Arrays And Needs

- The array stores a fixed-size sequential collection of elements of the same data type.
- It can be access more than one data using a single variable.
- A specific element in an array is accessed by an index

  array_name [index]

- All array have 0 is the index of 1st element. Called "base index"
- If an array have "N" elements, then the last element index is "N – 1"



int mark [20 , 35 , 68 , 200 , 50]

Array Name

Index :   0   1   2   3   4

Base index

N = 5 ,

then last index   N -1 = 5 – 1

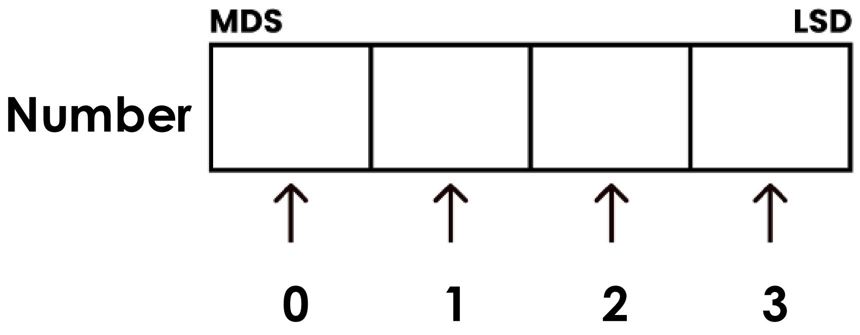
= 4

Defining Array

- Array must declare before using it
- Maximum number of elements an array can hold depends upon the size of an array
- Syntax :

```
DataType   ArrayName  [  ArraySize ];


       int  Number [ 4 ];
```



MDS

LSD

Number

0

1

2

3

Memory Allocation for Array

- Memory allocation done with the help of data type
- Character Type array have 1 bytes of memory
- Integer Type array have 2 bytes of memory
- Floating Point Type array have bytes of memory

- Example :

int Number [4];

↓ ↓

2 byte x 4 = 8 byte

Array Initialization

- **It is the process of storing initial values into the array memory**
- **Syntax :**

```
int item[ 3 ]   =   { 10 , 5 , 25 } ;
```

10	5	25

```
int item[ 5 ]   =   { 10 , 5 , 25 } ;
```

10	5	25		

Accessing the elements of an array

- It is the way of storing elements in to an array
- An element is accessed by indexing the array name; it is done by placing the index of the element with in square brackets [ ].

- Int item[3] = 10 ;

Accessing methods

1) int a[10] = {10, 11, 34, 55, 26 } ;

2) int a[ ] = { 34, 55, 26 } ;

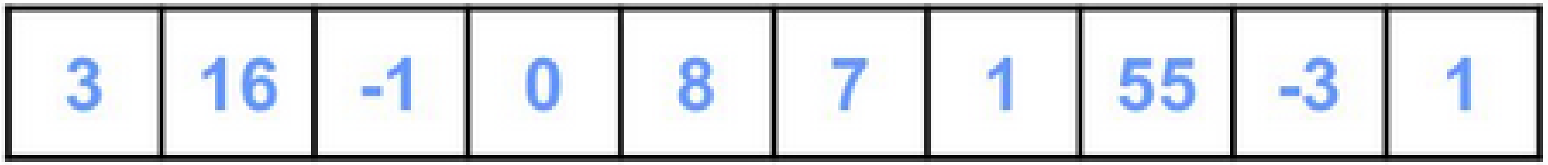
3) int a[0] = 10, a[1] = 11 , a[2] = 34 ;

4) int a[10];

```
for ( int i = 0; i < 10 ;  i ++)
      {
            cin>> a[ i ] ;
      }
```

Arrays Types

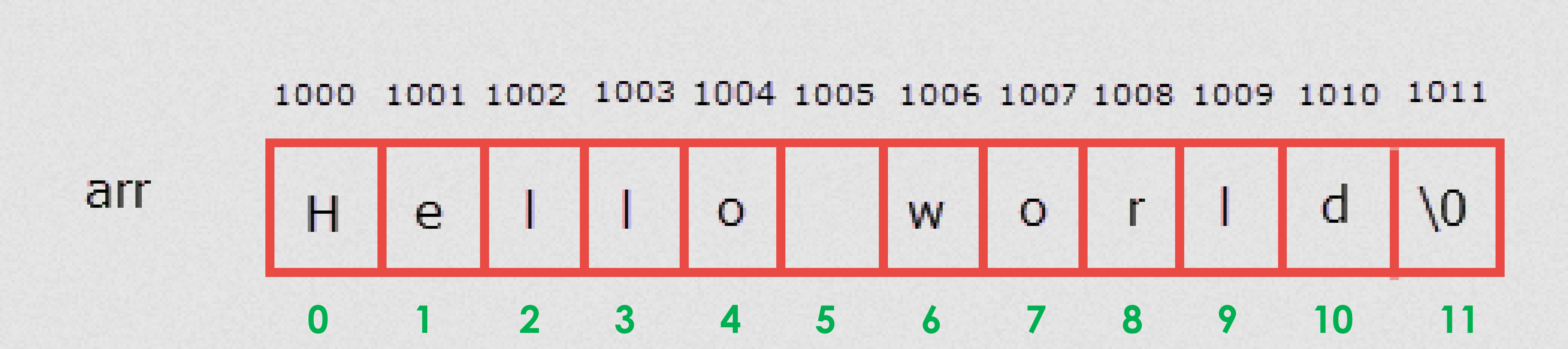
Int array



3	16	-1	0	8	7	1	55	-3	1
0	1	2	3	4	5	6	7	8	9

Int Num[ ] = { 3, 16, -1, 0, 8, 7, 1, 55, -3, 1 };

Char array



	1000	1001	1002	1003	1004	1005	1006	1007	1008	1009	1010	1011
arr	H	e	l	l	o		w	o	r	l	d	\0
	0	1	2	3	4	5	6	7	8	9	10	11

char **text[ ] = { 'H', 'e', 'l', 'l', 'o', '\0' };**

char **text[ ] = “ Hello ” ;**

Arrays Operations

Simple Arrays Operations

- **The operations performed on array**
  1) **Traversal**
  2) **Sorting**
     - **Selection Sorting**
     - **Bubble Sorting**
  3) **Searching**
     - **Linear Search**
     - **Binary Search**
  4) **Insertion**
  5) **Deletion**
  6) **Merging**

Traversal

- Traversal means accessing each elements at least once.
- Use this operation to check the correctness of insertion, deletion ,etc..
- Displaying all the elements of an array is an example.

Sorting

- Process of arranging the elements of the array in some logical order
- 2 algorithm used

  1) Selection Sorting
  2) Bubble Sorting

1) Selection Sorting

- Simple sorting techniques
- To sort an array in ascending order,
- the selection sort algorithm starts by finding the minimum value in the array and move it to the 1st position.
- At the same time, the element at the 1st position is shifted to the position of smallest element.
- This step repeats.

10	3	5	4	2	8	15

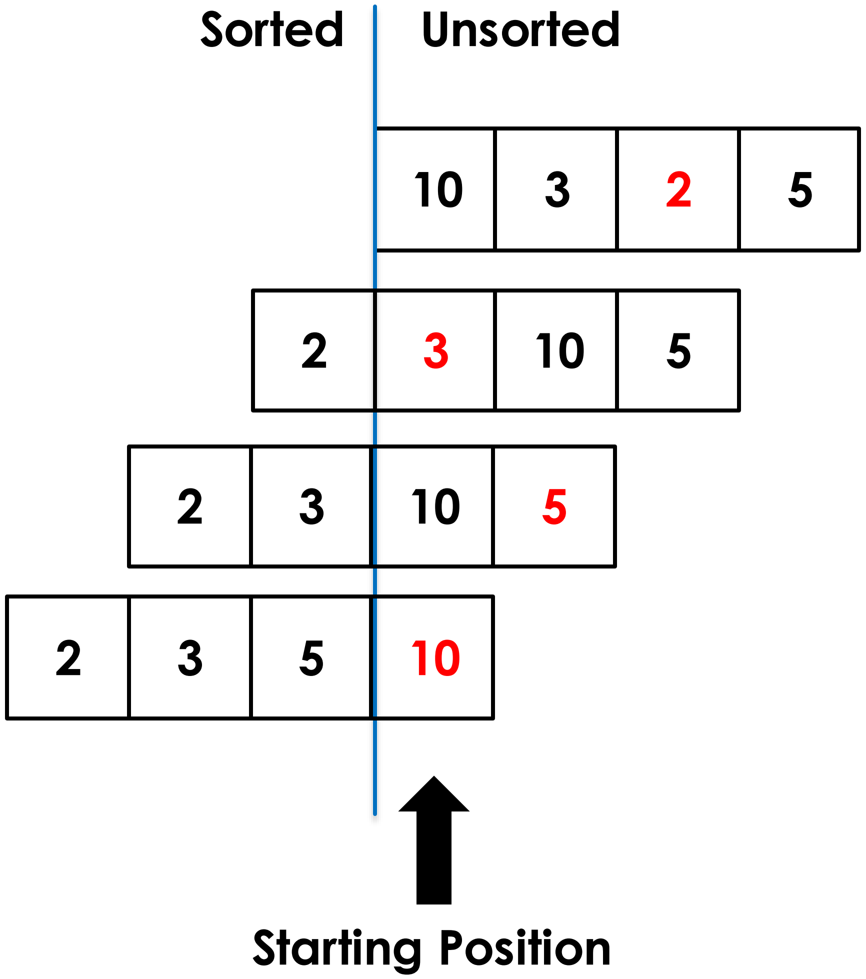
2	3	5	4	10	8	15

working

- Logic : Array is considered in to 2 parts
  - Unsorted part
  - Sorted Part
- Selection : 1. Select the lowest element in the remaining array
- Swapping : 2. Bring it to the starting position
- Counter Shift : 3. change the counter for unsorted array by 1

10	3	5	4	2	8	15

2	3	5	4	10	8	15



Sorted

Unsorted

10 3 2 5

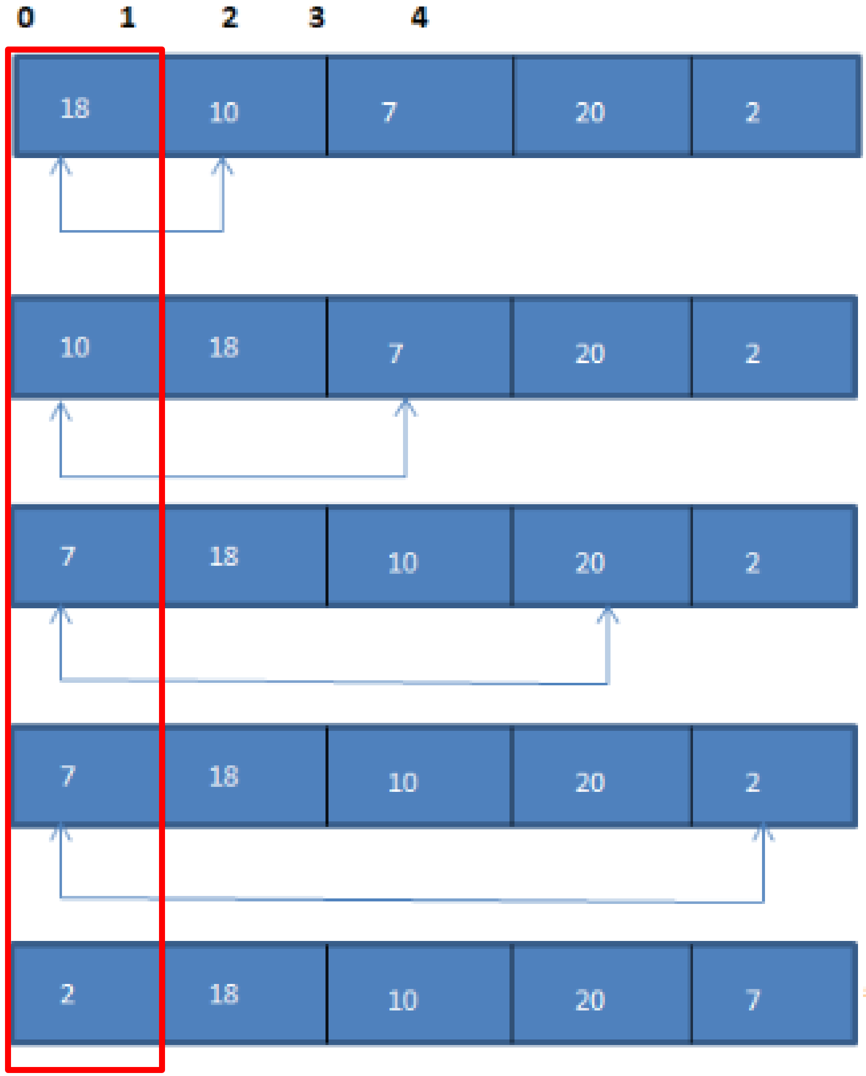
2 3 10 5

2 3 10 5

2 3 5 10

Starting Position

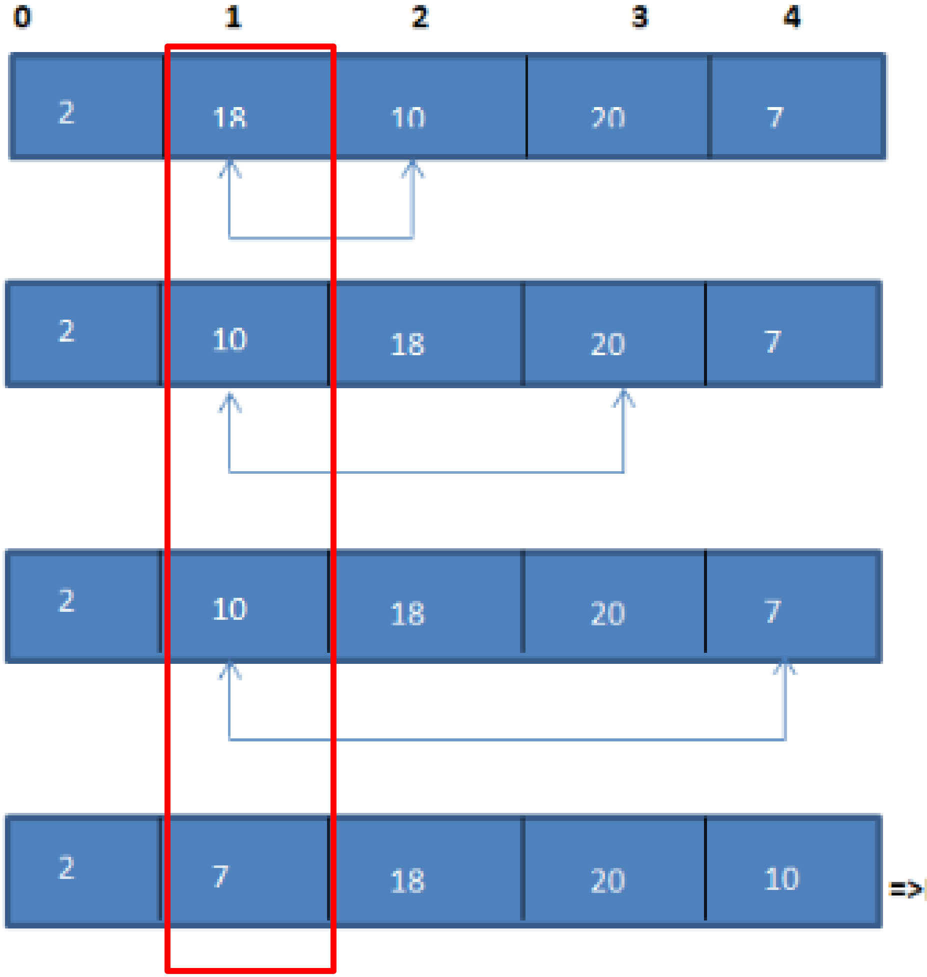
Pass 1:



0	1	2	3	4
18	10	7	20	2
10	18	7	20	2
7	18	10	20	2
7	18	10	20	2
2	18	10	20	7

=>smallest element at position 0

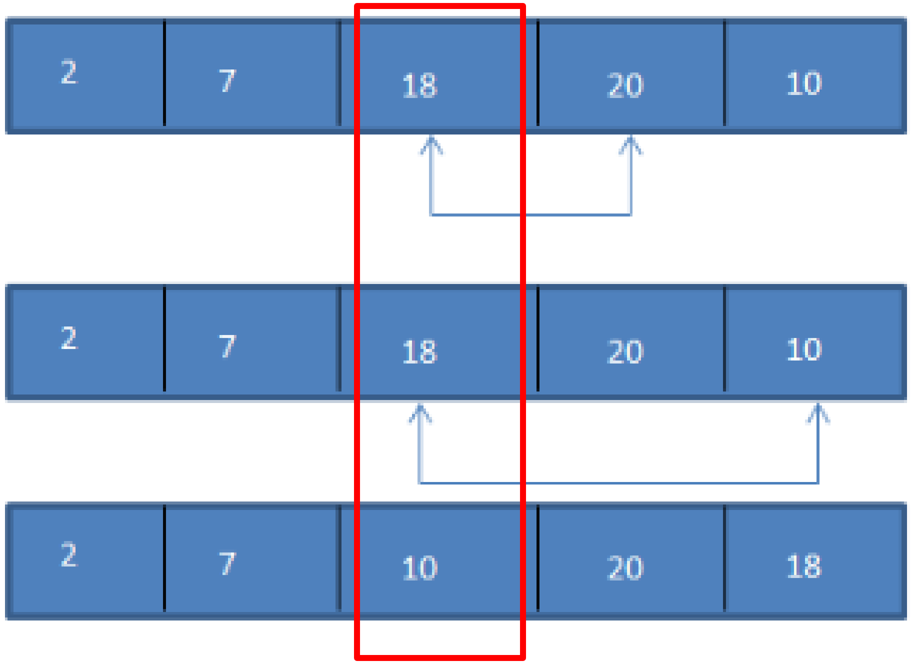
Pass 2:



0	1	2	3	4
2	18	10	20	7
2	10	18	20	7
2	10	18	20	7
2	7	18	20	10

=>Next smallest element at
position 1

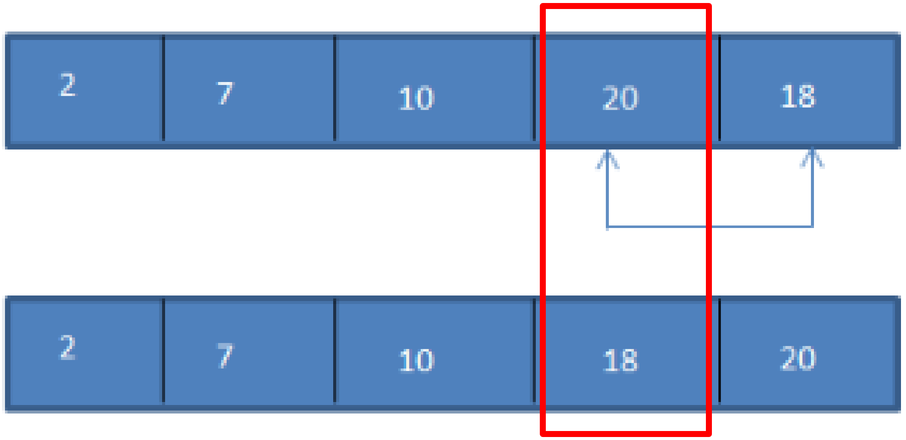
Pass 3:



2	7	18	20	10
2	7	18	20	10
2	7	10	20	18

=>Next smallest element at position 2

Pass 4:



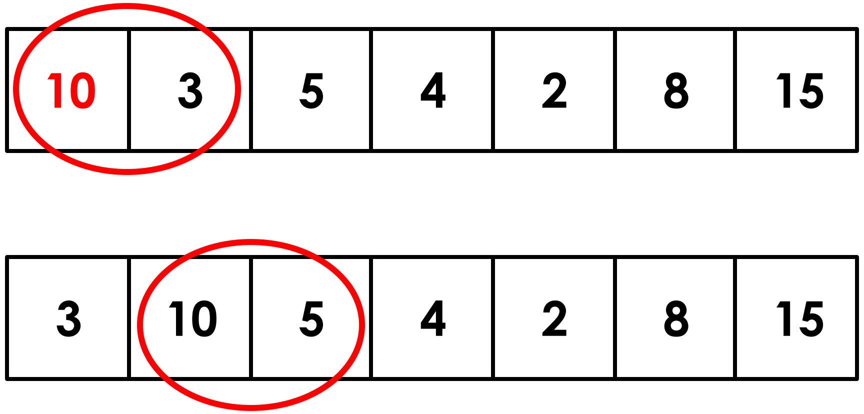
2	7	10	20	18

2	7	10	18	20

=>Sorted array

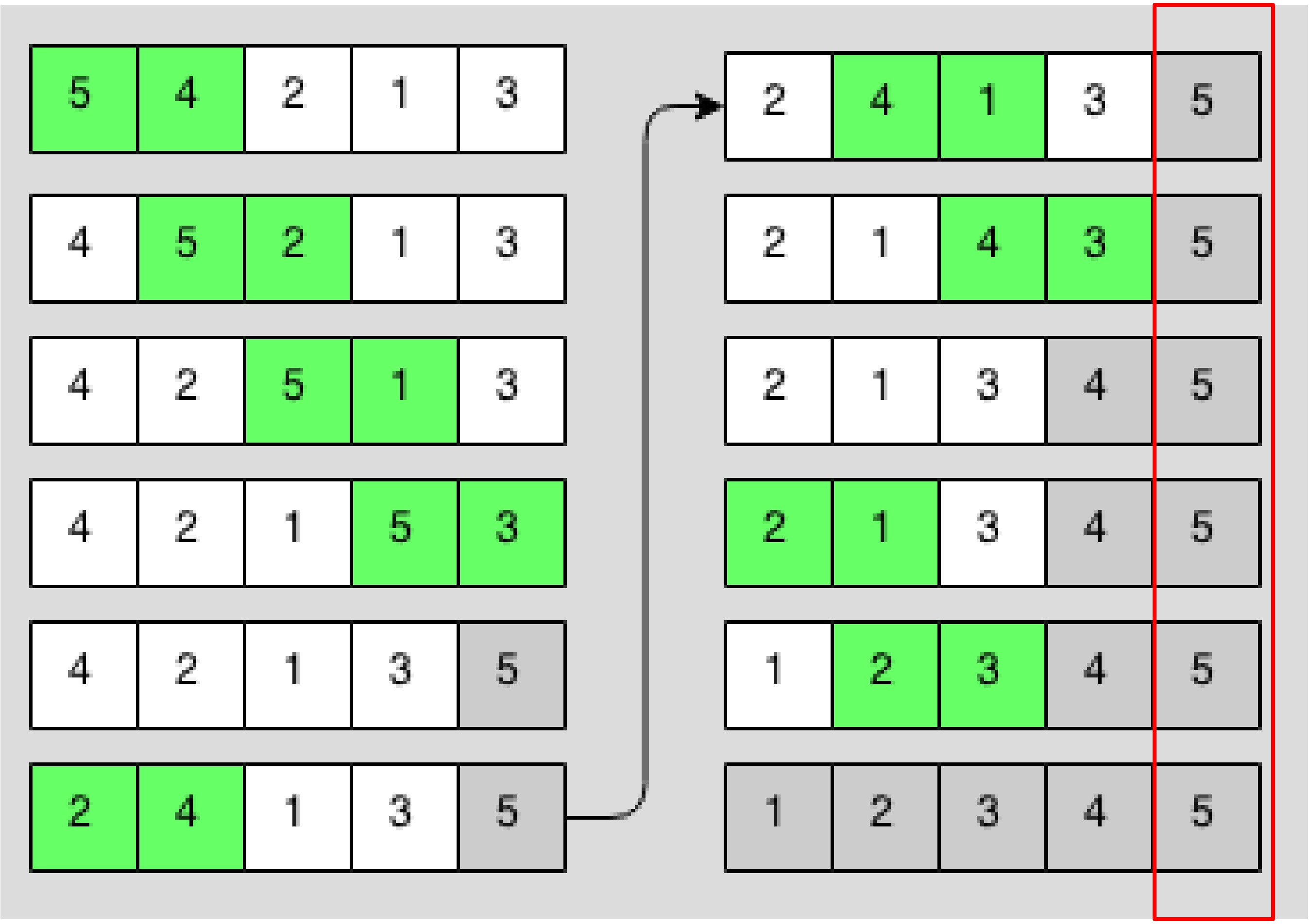
2) Bubble Sorting

- Simple sorting techniques
- It working by repeatedly swapping the adjacent elements if they are in wrong order.
- It shift largest element at the last position



10	3	5	4	2	8	15

3	10	5	4	2	8	15



5 4 2 1 3

4 5 2 1 3

4 2 5 1 3

4 2 1 5 3

4 2 1 3 5

2 4 1 3 5

2 4 1 3 5

2 1 4 3 5

2 1 3 4 5

2 1 3 4 5

1 2 3 4 5

1 2 3 4 5

Sorted part

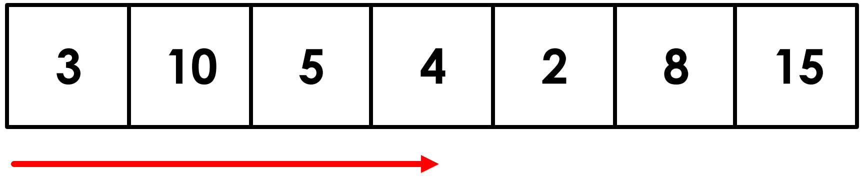
Largest element

Searching

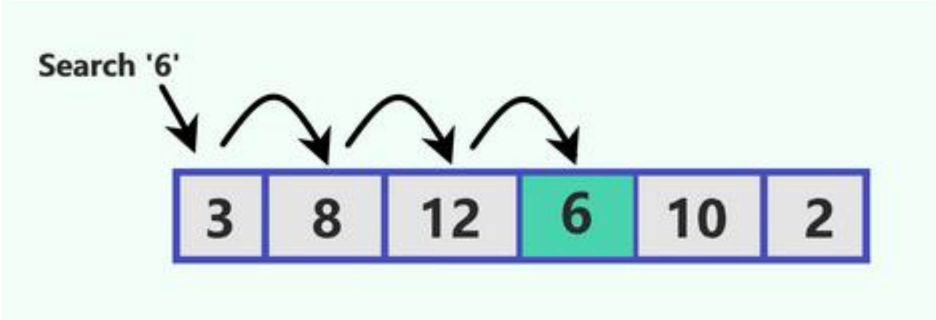
- Process of finding the location of the given elements in an array.
- 2 algorithm used
  1) Linear Search
  2) Binary Search

1) Linear Search

- Also called Sequential search.
- It is a method for finding a particular value in a list
- Linear search consist of checking each element in the list, once at a time
- Start from 1st element ;
- Ends when the value was founded

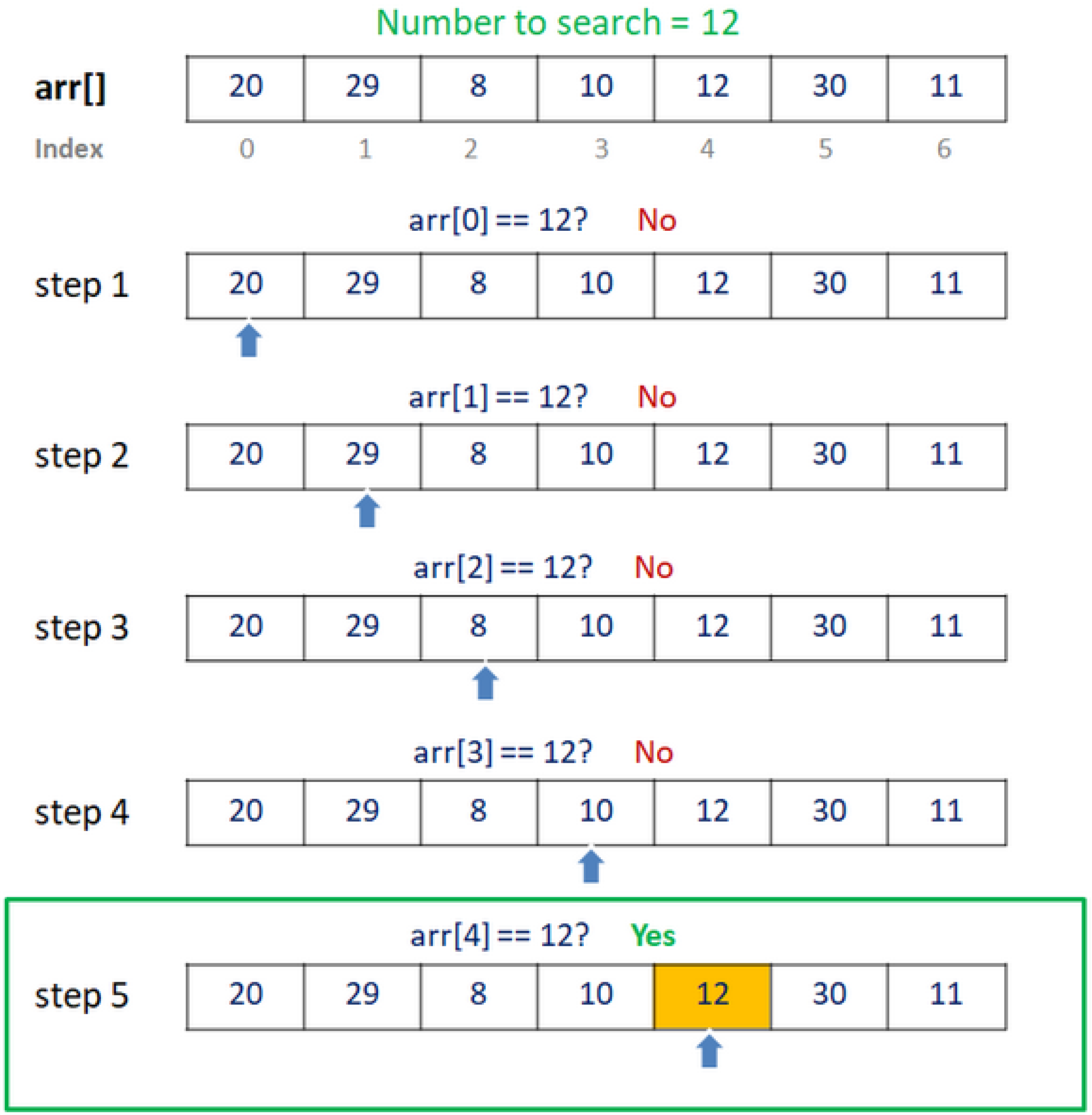


3	10	5	4	2	8	15



Search '6'

3	8	12	6	10	2



Number to search = 12

arr[]	20	29	8	10	12	30	11
Index	0	1	2	3	4	5	6

arr[0] == 12? No

step 1	20	29	8	10	12	30	11

arr[1] == 12? No

step 2	20	29	8	10	12	30	11

arr[2] == 12? No

step 3	20	29	8	10	12	30	11

arr[3] == 12? No

step 4	20	29	8	10	12	30	11

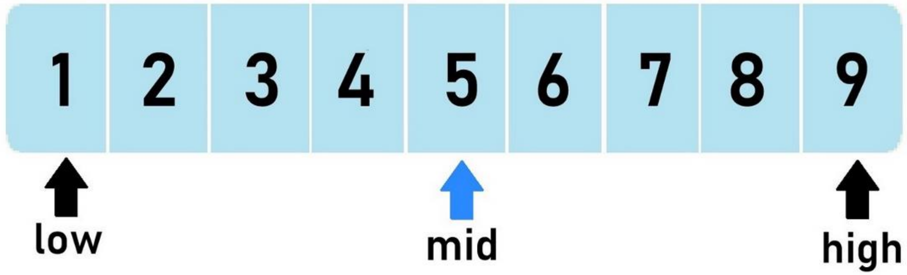
arr[4] == 12? Yes

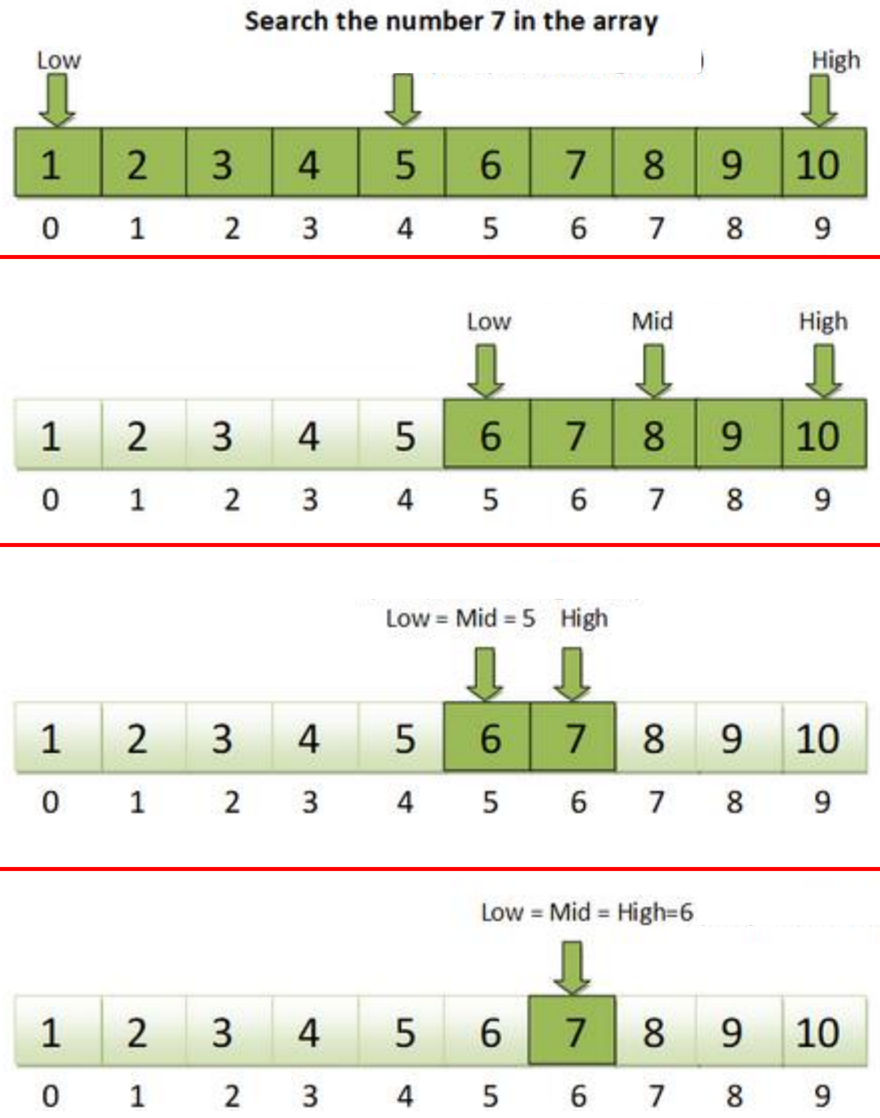
step 5	20	29	8	10	12	30	11

2) Binary Search

- **Used in large arrays**
- **More efficient search**
- **Fast searching than others**
- **It works only in sorted arrays**

- **Search a sorted array by repeatedly dividing the search interval in half**
- **Begin with an interval covering the whole array**
- **If the search value is less than the middle item, narrow the interval to the lower half. Otherwise narrow it to the upper half.**
- **To calculate middle index**
- mid = (low + high) / 2;   (Integer part only taken)
- **(0 + 9) / 2 = 4.5 = 4**





Mid = (0 + 9) / 2
Mid = 4.5
Mid = 4

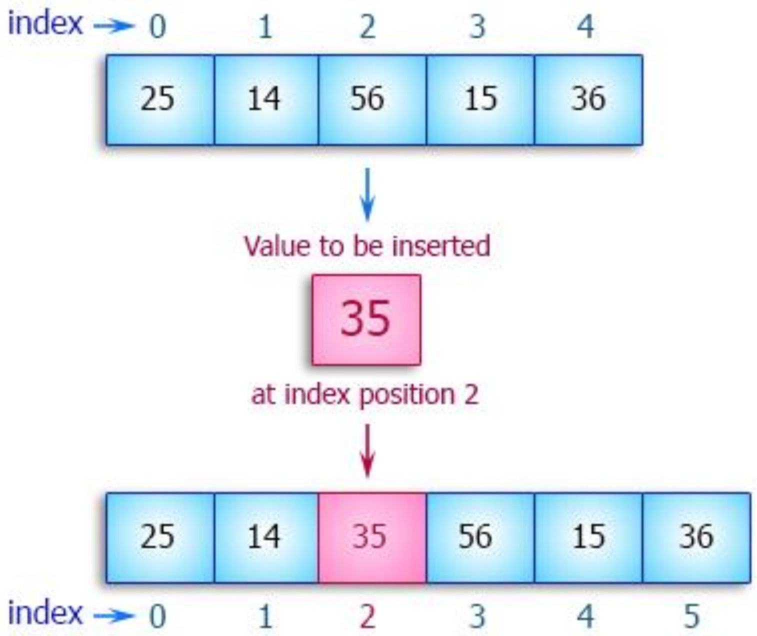
Mid = (5 + 9) / 2
Mid = 7

Mid = (5 + 6) / 2
Mid = 5.5
Mid = 5

Low = High

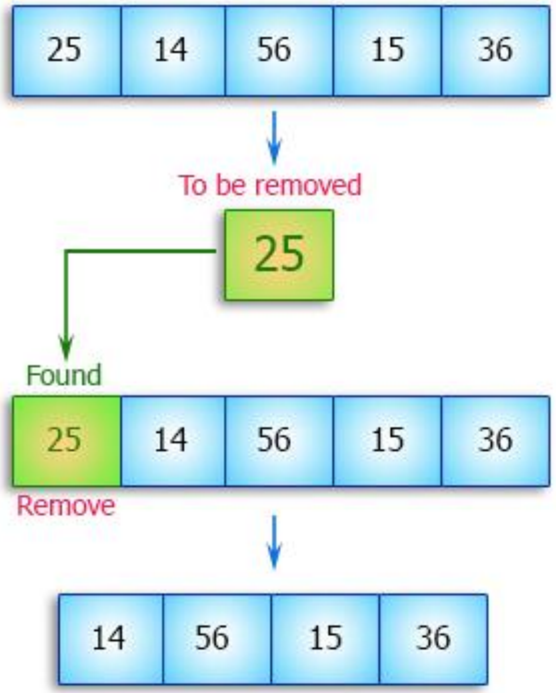
Insertion

- It is the operation to insert one or more data elements into an array
- Based on the requirements, new data can be added at beginning, end or any given index



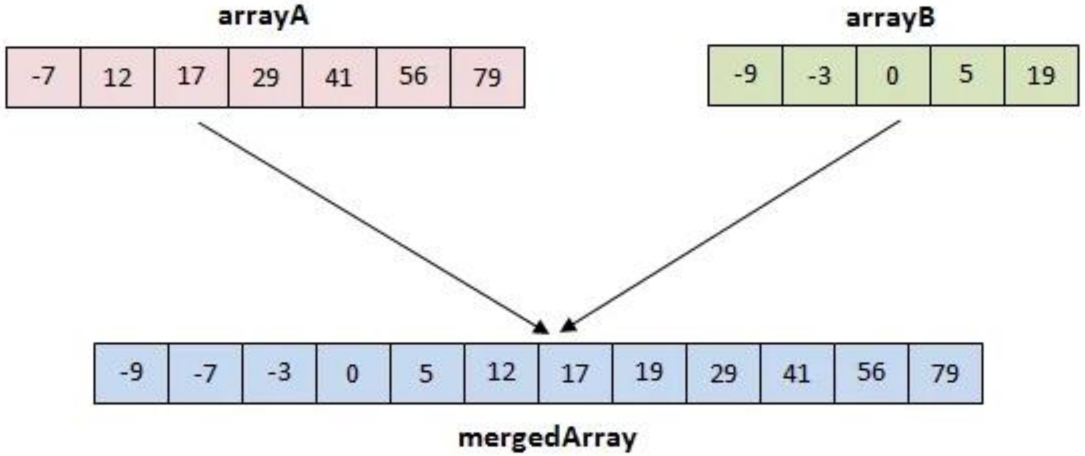
Deletion

- It is the operation to remove one or more data elements from an array
- Based on the requirements, data can be removed at beginning, end or any given index



Merging

- It is the operation to adding or concatenate one array element with another array element.



arrayA

-7	12	17	29	41	56	79

arrayB

-9	-3	0	5	19

-9	-7	-3	0	5	12	17	19	29	41	56	79

mergedArray