Date: / / Page no:

- **SORTING**

Process of arranging data into meaningful order so that you can analyze the data more efficiently.

Bubble-Sort

- Bubble Sort works by repeatedly swapping the adjacent element if they are in the wrong order.

- It is not suitable for large data because it is having quite high worst-time complexity.

{ worst case ⇒ $O(n^2)$ }

* Working :-

array :- | 7 | 11 | 9 | 2 | 17 | 4 | → (0,1) Compare

7 NC 11 9 2 17 4

(i=0) 1st pass :- 7 11 9 2 17 4 → (1,2) Compare

swap

7 9 11 2 17 4 → (2,3) Compare

swap

Date: / / Page no:

7, 9, 2, 11, 17, 4 → (3,4) Compare

NC

7, 9, 2, 11, 17, 4 → (4,5) Compare

swap

7, 9, 2, 11, 4, 17

(1st pass completed)

(i=1) 2nd pass :-

| 7 | 9 | 2 | 11 | 4 | 17 | → (1,2) Compare

swap

7 2 9 11 4 17 → (2,3) Compare

NC

7 2 9 11 4 17 → (3,4) compare

swap

7 2 9 4 11 17 → (4,5) Compare

NC

7, 2, 9, 4, 11, 17

(2nd pass completed)

Since 0 and 1 are at NC position so we have not compared it here.

(i=2) 3rd pass :-

0	1	2	3	4	5
7	2	9	4	11	17

→ (0,1) Compare

swap

swap

~~7, 2, 4, 9, 11, 17~~

2, 7, 9, 4, 11, 17 → (1,2) Compare

NC

2, 7, 9, 4, 11, 17 → (2,3) Compare

swap

2, 7, 4, 9, 11, 17 → (3,4) Compare

NC

No need

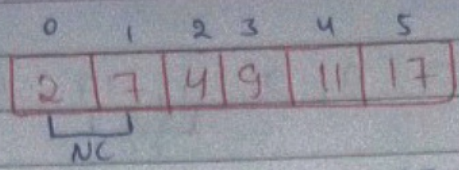
x

x 2, 7, 4, 9, 11, 17 → (4,5) Compare

NC

Date: / / Page no:

(i=3) 4th pass :-



→ (0,1) Compare

2, 7, 4, 9, 11, 17 → (1,2) Compare

swap

2, 4, 7, 9, 11, 17

(4th pass completed)

(i=4) 5th Pass :-

2 4 7 9 11 17

NC

2,	4,	7,	9,	11	17

(array sorted)

* Here (i=0,1,2..) is the value which we are decreasing from the array length because at each pass we get the max element in the end. So we do not need to compare it.

It is also known as sinking and exchange sort.

It is a stable array means order of elements remain same.

* Code :-

Q:- W.A.P. to sort the array in ascending order using bubble sort.

```
import java.util.*;
public class bubbleSort
{
  public static void main (String[] args)
  {
    int[] arr = {4, 1, 6, 9, 3.4};
    bubble (arr);
    Syso (Arrays.toString (arr));
  }

Static void bubble (int[] arr)
{
  for (int i = 0; i < arr.length ; i++)
  {
    for (int j = 1 ; j < arr.length - i ; j++)
    {
      if (arr[j] > < arr[j-1])
      {
        int temp = arr[j];
        arr[j] = arr[j-1];
        arr[j-1] = temp;
      }
    }
  }
}
```