Date - 16/04/2020

Sorting :→ Decorate the given unsorted data either in ascending order or descending order

↳ Bubble sort → first of all, lets take an example.

I/P: 15 16 8 4 3, no of data = 5

Pass 1: 15 16 8 4 3, swapping is not required

15 16 8 4 3, swapping is required

15 8 16 4 3, swapping is required

15 8 4 16 3, swapping is required

15 8 4 3 16 → output of first pass.

↓ this become Input of the second pass.

I/P: 15 6 8 4 3

Pass 1: 15 8 4 3 | 16, no. of comparisions = 4

Pass 2: 8 4 3 | 15 16, no. of comparisions = 3

Pass 3: 4 3 | 8 15 16, no. of comparisions = 2

Pass 4: 3 | 4 8 15, 16, no. of comparisions = 1

O/P: 3 4 8 15 16 → Sorted data.

↳ In bubble sort technique, each pair of adjacent element is compared and elements are swapped if they are not in order.

↳ In general, bubble sort takes n-1 passes to sort the given n number of unsorted data.

↳ after the completion of first pass, largest element of given array of data will reach at its own position/Rank.

↳ output of the first pass become the input of the second pass, after completion of second pass second largest element will reach at its on position and so on.

Analysis of bubble sort:

So, next our goal is how we compute the time complexity in different cases.

Time complexity: The number of machine instructions which a program executes during its running time is called time complexity.

Note → one machine instructions count one unit of time.

In sorting algorithm:

Time complexity ∝ Total No. of comparisions.

Time complexity $\propto$ Total no. of comparisions.

There are three cases.

- Best Case
- Worst case
- Average case

- Best case: Input: - Already sorted data

No. of data = n.

Required Number of Passes = 1.

No. of comparisions = n-1

Time complexity = $O(n)$, $n \to \infty$

- Worst case: given data: increasing order.

I/P: 4, 8, 10, 12, 18, 20

O/P: 20 18 12 10 8 4

No. of data = n

Number of passes = n-1

no. of data = n
no. of comparisons.

Pass 1 : n-1

Pass 2 : n-2

Pass 3 : n-3

- - - - .
- - - - .
- - - .

Pass n-2 : n-(n-2) = 2

Pass n-1 : n-(n-1) = 1

Pass 1 + Pass 2 + Pass 3 + ---- + Pass n-2 + Pass n-1

= n-1 + n-2 + n-3 + ---- + 2 + 1

or

1 + 2 + 3 + ---- + n-3 + n-2 + n-1

As we know that

$$1+2+3+\cdots+n-2+n-1+n = \frac{n(n+1)}{2} \longrightarrow (1)$$

Put n = n-1 in equation (1)

$$= \frac{(n-1)(n-1+1)}{2} = \frac{n(n-1)}{2}$$

Total No. of comparisons $= \frac{n^2}{2} - \frac{n}{2}$

Time complexity = $O(n^2)$, n is too large

$n \to \infty$.

$n^2 >> n$:

$n \to \infty$