08 : Sorting

* Sorting refers to the operation of arranging data in some gun order, like ↑sing ↓ing with numerical data / alphabeetically, with char data.
* 4 types →

① Insertion Sort =

* It is a simple sorting algo which sorts the array by shifting elemnts one.
* Has 1 of the simplest implemntation.
* It is efficient for smaller data sets bt very inefficient for largr lists.
* It is adaptive (reduces its total num of stps if gun a partially sorted list).
* Its space complexity is less like bubble.

* (Al) ⇒ / prgm =

```
for(i=1 ; i<n ; i++) {
        temp = a[i];
        j = i-1;
    while (j>=0 && a[j] > temp) {
        a[j+1] = a[j];
        j=j-1;
    }
    a[j+1] = temp;
}
```

0 1 2 3 4

9 3 2 5 4 ⇒ 2 9 3 5 4

2 3 9 5 4 ⇒ 2 3 5 9 4 ⇒ 2 3 4 5 9

② Selection Sort =

* Here, Smallest element is exchanged with the 1st element of unsorted list of elements.
* 2nd Smallest element is exchanged with the 2nd element of unsorted list of elements & so on until all the elements are sorted.

* (A) Pgm =

```
for(i=0; i<limit-1; i++) {
    for(j=i+1; j<limit; j++) {
        if(A[i] > A[j])
        {
            temp = A[i];
            A[i] = A[j];
            A[j] = temp;
        }
    }
}
```

eg → | 25 | 17 | 31 | 13 | 2 |

no shifting only exchange

1st Iteration

| 25 | 17 | 31 | 13 | 2 | (compare 0th with all. element → exchange)
|---|---|---|---|---|
| 17 | 25 | 31 | 13 | 2 | ✓
| 17 | 25 | 31 | 13 | 2 | → 13 | 25 | 31 | 17 | 2
| 2 | 25 | 31 | 17 | 13 |

2nd .i

| 2 | 25 | 31 | 17 | 13 | (compare 1st with 2,3,4)
|---|---|---|---|---|
| 2 | 25 | 31 | 17 | 13 |
| 2 | 17 | 31 | 25 | 13 |
| 2 | 13 | 31 | 25 | 17 |

3rd .i

| 2 | 13 | 31 | 25 | 17 | (compare 2nd with 3,4)
|---|---|---|---|---|
| 2 | 13 | 25 | 31 | 17 |
| 2 | 13 | 17 | 31 | 25 |

4th .i

| 2 | 13 | 17 | 31 | 25 | (compare 3rd with 4, till these all values are sorted)
|---|---|---|---|---|
| 2 | 13 | 17 | 25 | 31 | ✓

* eg for insertion Sort →

25	17	31	13	2

1st .i (shifting)

| 25 | 17 | 31 | 13 | 2 | (compare 1st with 0th)
|---|---|---|---|---|
| 17 | 25 | 31 | 13 | 2 |

2nd .i

| 17 | 25 | 31 | 13 | 2 | ✓ (compare 2nd with 0,1)
|---|---|---|---|---|
| → 17 | 25 | 31 | 13 | 2 | ✓

3rd .i

| 17 | 25 | 31 | 13 | 2 | (compare 3rd with 0,1,2)
|---|---|---|---|---|
| 13 | 17 | 25 | 31 | 2 |
| → 13 | 17 | 25 | 31 | 2 ✓
| → 13 | 17 | 25 | 31 | 2 ✓

4th .i

| 13 | 17 | 25 | 31 | 2 | (compare 4th with 0,1,2,3)
|---|---|---|---|---|
| 2 | 13 | 17 | 25 | 31 | ✓
| → 2 | 13 | 17 | 25 | 31 | ✓
| 2 | 13 | 17 | 25 | 31 | ✓

| 2 | 13 | 17 | 25 | 31 | ✓
|---|---|---|---|---|

③ Exchange Sort / Bubble Sort =

* It is the oldest & simplest sorting (al) in use.
* It works by comparing each item in the list with the item next to it. & swapping these 2 items if they are in wrong order. This will repeated until no swaps are needed (i.e) sorted.

* (Al) (prgm =

```
for( i=0 ; i<elements-1 ; i++) {
    for ( j=0 ; j<elements-i-1 ; j++) {
        if (a[j] > a [j+1])
        {
            temp = a[j] ;
            a[j] = a [j+1] ;
            a[j+1] = temp ;
        }
    }
}
```

* eg →

84	69	76	86	94	91

* 84	69	76	86	94	91	
* 69	84	76	86	94	91	
* 69	76	84	86	94	91	✓
* 69	76	84	86	94	91	✓
* 69	76	84	86	94	91	
* 69	76	84	86	91	94	

④ Quick Sort =

* most popular sorting technique.
* As the name suggests the Q.sort is the fastest known sorting (al)
* It has the bst avg time performance.
* It works by partitioning the array to be sorted & each partition in turn sorted recursively, → partition exchange sort

[* divide & conquer breaks a pbm into sub pbms that are similar to the original pbm, recursively solves the sub pbms, & finally combines the solⁿ to the sub pbms to solve the original pbm.]

* eg →

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

→ pivot (last element of the list)

ആദ്യം 4-നെ [illegible] crt position-ൽ [illegible] then partition for that 4-നെ [illegible] element [illegible] compare [illegible] → ≤ 4 ≥ [illegible]

7	1	3	5	2	6	4

→ pivot

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

left of 4   right of 4

2-നെ crt loc-ൽ [illegible]

1	2	3	4	7	5	6

→ pivot (6)

=> | 1 | 2 | 3 | 4 | 5 | 6 | 7 | Sorted

1<2, 3>2

5<6, 7>6

Name	Best	Avg	worst	Stable	Method
B. Sort	$O(n^2)$	$O(n^2)$	$O(n^2)$	Yes	exchanging
S. Sort	$O(n^2)$	"	"	No	Selection
I. Sort	$O(n)$	"	"	Yes	Ins
Q. Sort	$O(n \log n)$	$O(n \log n)$	"	No	partition