

## 08: Sorting

\* Sorting refers to the operation of arranging data in some given order, like arranging numerical data / alphabetically, with char data.

\* 4 types →

### ① Insertion Sort =

- \* It is a simple sorting algorithm which sorts the array by shifting elements one.
- \* Has one of the simplest implementations.
- \* It is efficient for smaller data sets but very inefficient for larger lists.
- \* It is adaptive (reduces its total no. of steps if given a partially sorted list).
- \* Its space complexity is less like bubble sort.

### \* (AI) = | 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;
}
```

• 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 =

\* Hence, smallest element is exchanged with the 1<sup>st</sup> element of unsorted list  
 \* elements  
 \* and smallest element is exchanged with the 2<sup>nd</sup> element of unsorted list or  
 elements i.e. so on until all the elements are sorted.

(P)  $\int \cos x =$

```

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

```

|    |    |    |    | negative |
|----|----|----|----|----------|
| 25 | 17 | 31 | 13 | 2        |
| 17 | 25 | 31 | 13 | 2        |
| 17 | 25 | 31 | 13 | 2        |
| 2  | 25 | 31 | 17 | 13       |

(Combine 5th with all pigment — no change)

3rd - 1st

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

2 | 13 | 31 | 25 | 17

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

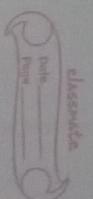
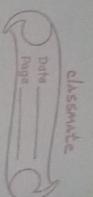
1st - 9  
25 11 3 3 2  
Skating  
39 yrs 6 m

| 2 <sup>nd</sup> - 1 |    |
|---------------------|----|
| 17                  | 25 |
| 31                  | 13 |
| 2                   | ✓  |

combase 2<sup>nd</sup> w/bg 0, 1

• 547

13 17 25 31 12  
2 13 17 25 (31) ✓ combine with 0, 1, 2, 3



### 3 Exchange Sort / Bubble Sort =

- \* It is oldest & simple sorting (all) in use.
- \* It works by comparing each item in the list with the item next to it. If swapping these 2 items if they are in wrong order. They will swapped until no swaps are needed i.e. sorted.

#### 4) Program =

```
for(i=0; i<elements-1; i++) {
    for(j=0; j<elements-i; 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 |

### 4 Quick Sort =

- \* most popular sorting technique as the name suggests the Q.Sort is the fastest known sorting (all)
- \* It has the best 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 prob into sub problems that are similar to the original problem, recursively solves the sub problems, finally combines the soln to the sub problems to solve the original prob.]

\* eg → 

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 7 | 1 | 3 | 5 | 2 | 6 | 4 |
|---|---|---|---|---|---|---|

 → pivot (last element)

on 4-th pos. swap 2nd & last position  
for that 4-th pos. elem move compare  
answ →  $\leq_{12}$  arrangement.

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 7 | 1 | 3 | 5 | 2 | 6 | 4 |
|---|---|---|---|---|---|---|

|   |   |     |   |   |   |   |
|---|---|-----|---|---|---|---|
| 1 | 3 | (2) | 4 | 7 | 5 | 6 |
|---|---|-----|---|---|---|---|

left 94 right 91

2-arr. arr. loc-min. among.

5 < 6

✓

|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 7 | 5 | 6 |
|---|---|---|---|---|---|---|

✓

✓

✓

✓

122  
Sorted

322

classmate

Date \_\_\_\_\_  
Page \_\_\_\_\_

| Name    | Best          | Avg           | Worst    | Space | Method     |
|---------|---------------|---------------|----------|-------|------------|
| B. Sort | $O(n^2)$      | $O(n^2)$      | $O(n^2)$ | 4S    | Exchanging |
| S. Sort | $O(n^2)$      | "             | "        | No    | Selection  |
| T. Sort | $O(n)$        | "             | "        | 4S    | Ing        |
| Q. Sort | $O(n \log n)$ | $O(n \log n)$ | "        | No    | partition  |