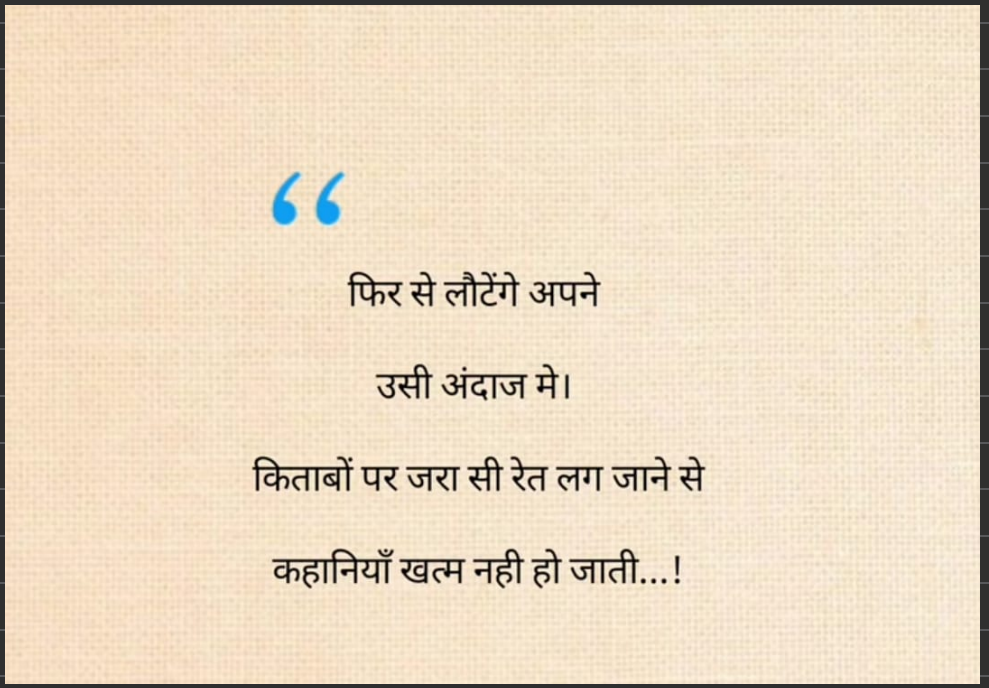
Sorting Algorithms



What is sorting

Arranging the numbers in either increasing or decreasing order.

Ascending → 1 1 2 3 4 4

descending → 5 5 4 3 2 1 1

increasing → 1 2 3 4 5 6

decreasing → 6 5 4 3 2 1

Stable Sorting

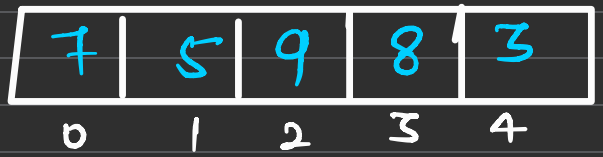
Relative position of duplicate numbers are maintained

$1 \quad 2_a \quad 4 \quad 3 \quad 2_b \quad 6 \rightarrow 1 \quad 2_a \quad 2_b \quad 3 \quad 4$

In-place Sorting

No need of extra space to sort the elements.

Bubble Sort



- mainly compares two adj elements
  if they are incorrectly placed → swap

$x > y$ (incorrect)

N = 5

pass 1

7 5 9 8 3
5 7 9 8 3
5 7 9 8 3
5 7 8 9 3
5 7 8 3 9

(A)

pass 2

5 7 8 3 | 9
5 7 8 3 9
5 7 8 3 9
5 7 3 8 9

pass 3

5 7 3 | 8 9
5 7 3 8 9
5 3 7 8 9

pass 4

5 3 | 7 8 9
3 5 7 8 9

, E / E
H / H
B / B
V, / V
R / R
, P / P

Size of array = N (5)

No. of passes = N-1 (4)

size	No. of passes	Comparision
5	pass 1	4
	pass 2	3
	pass 3	2
	pass 4	1

(N - pass)

```
void bubblesort (arr, N)
{
    for (pass=1 ; pass <= N-1 ; pass++)
    {
        for (i=0 ; i<(N-pass); i++)
        {
            if (a[i] > a[i+1])
            {
                swap (a[i], a[i+1])
            }
        }
    }
}
```

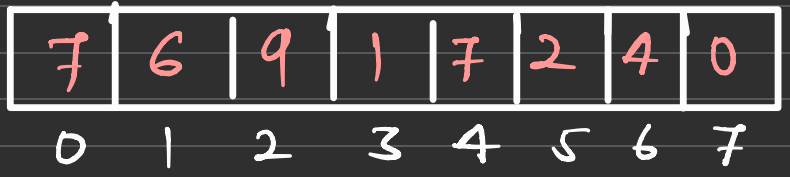
$O(N-1)$

$O(N-1)$

$O(N-1) \times O(N-1)$

$O(N^2 - N - N + 1)$

$O(N^2)$



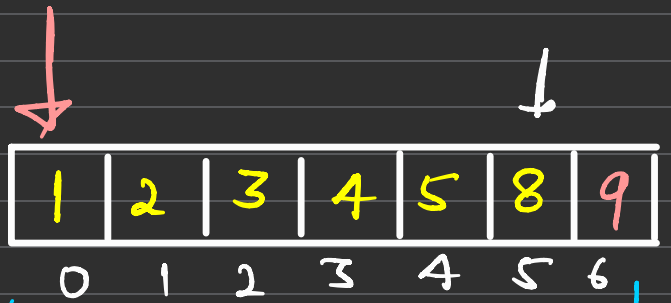
N = 8

Find the index of smallest No. ?

```
int small = a[0]
int index = 0

for (i = 1 ; i < N ; i++)
{
    if (a[i] < small)
    {
        small = a[i]
        index = i
    }
}
```

Selection Sort



smallNo = ~~8~~

minIndex = ~~6~~

(N-1) 1 > Correct position

```
void selectionsort (arr, N)
{
O(N)  for (i=0 ; i<N ; i++)
      {
        smallNo = a[i]
        minIndex = i
O(N)    for (j = i+1 ; j<n ; j++)
        {
          if (a[i] < smallNo)
          {
            smallNo = a[i]
            minIndex = i
          }
        }
        swap (arr[i], a[minIndex])
      }
}
```

O(N X N)

O(N²)

N = 7

9 3 8 4 5 1 2

Count the inversion ? 6 + 2 + 4 + 2 + 2

inversion : pair such that they are not in correct order , a(i) > a(i+k)

K = 0, 1, 2, 3.... N-1)

Insertion Sort