M T W T F S S

Page No.:

Date:

YOUVA

Selection Sort

Selection Sort is works on placing smallest element to its correct position until every element is not is on its correct position.

↳ Smallest element lekar, usko right jagah par rakhna hai.

arr[] = 64, 25, 12, 22, 11 — n = 5

0 1 2 3 4

min

Note:
Swap smallest element in the array to its 1st position. And second smallest to its 2nd position and so on

Round 1: 64 25 12 22 11

i=0

swap

Round 2: 11 25 12 22 64

swap

i=1

Round 3: 11 12 25 22 64

i=2

swap

Round 4: 11 12 22 25 64

No swapping.

i=3

Total Round = (n-1)

Note:
Stable Algorithm: If the two element have the same value and appear in the input in a particular order a stable sorting algorithm will always arrange them in same order.

7 May 2023, 8:58 pm

M T W T F S S

Page No.: YOUVA

Date:

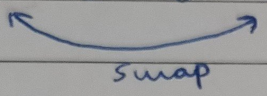
Selection Sort is not stable because it does not guarantee the relative order of equal element in the input list after sorting.

For Example:

5, 3, 5, 2 (indices: 0, 1, 2, 3) $n = 4$

R1: 5 3 $\bar{5}$ 2

i=0



R2: 2 3 $\bar{5}$ 5 (5 is coming after $\bar{5}$ but in original $\bar{5}$ is after 5)

This situation is not stable.

→ When to use selection sort:

→ For small list (100-200)
→ When stability not required
→ Memory efficient

→ When Not to use selection sort:

→ for large list
→ Unstable

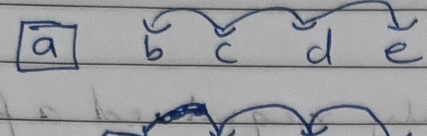
M T W T F S S

Page No.: YOUVA

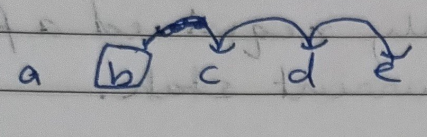
Date:

Time Complexity:

size = n



4 (n-1)



3 (n-2)

1 1

$= (n-1) + (n-2) + \dots\dots\dots + 2 + 1$

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

$= \frac{n^2 - n}{2}$

Time Complexity = ~~(n^2)~~ $O(n^2)$

Best Case (Already Sorted) = $O(n^2)$

Worst Case (Totally Unsorted) = $O(n^2)$

During Sorted arr we still have to go for all rounds.

7 May 2023, 8:58 pm

Space Complexity:

Space Complexity = O(1)

This is because this only required a few variable to keep track of current state.

Code:

```
void selection(int *arr, int size)
{
    for (int = 0; i<size; i++)
    {
        int min = i;
        for (int j = i; j<size; j++)
        {
            if (arr[j] < arr[min])
            {
                min=j;
            }
        swap(arr[min], arr[i]);
    }
}
```