Date _____ Page _____

Day-25

Selection Sort

Sorting:

⇒ Arranging the elements in correct order either in ascending or descending order.

⇒ Sorting helps in finding any element more fastly.

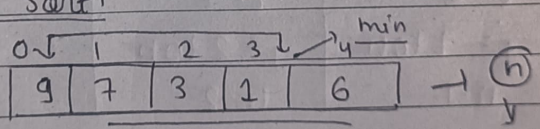
Ex:

3	8	1	2	5

Asending →

1	2	3	5	8

⇒ **Selection Sort:**



R 1 (0-4) 1 7 3 9 6 (min) — n-1 Round

R2 (1-4) 1 3 7 9 6

R3 (2-4) 1 3 6 9 7

R4 (3-4) 1 3 6 7 9

⇒ Code for find min. element index —

```
index=0;
for (int i=0; i<n; i++){
    if (arr[i] < arr[index])
        index= i;
}
```

Date _____ Page _____

⇒
```
for(i=0; i<n-1; i++){
    index = i;
    for(j=i+1; j<n; j++){
        if(arr[j] < arr[index])
            index = j;
    }
    swap(arr[index], arr[i]);
}
```

Space Complexity:
Auxilliary : O(1)
Total Space : O(n)

Note
arr[5], arr[2000]
⇒ both are constant.

Time Complexity:

⇒

i = 0	i = 1	----	i = n-2
j = 1 to n	j = 2 to n	----	j = n-1 to n
n-1	n-2		1

⇒ $(n-1) + (n-2) + \dots + 2 + 1$

⇒ $\frac{n \times n-1}{2} = \frac{n^2 - n}{2} = n^2$

⇒ $O(n^2)$ → ~~Best Case~~ Worst Case
$\Omega(n^2)$ → ~~W~~ Best Case
$\Theta(n^2)$ → Avg. Case