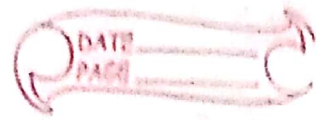


Name - Abhinav Bhatt
Branch - CSE



Roll no - 11911055

Explanations

Q3. Perform Analysis on time complexity -
lower order term.

Sol.

Case I

Best case Time complexity of
Insertion Sort.

It happens only when the elements
given are itself in sorted order.
as by the program we can see
there are two for loops

```
for (i=1; i<n; i++)  
{
```

```
    k = arr[i]
```

```
    for (j=i-1; j>=0 && k<arr[j];  
         j--)
```

Teacher's Sign.

Ex-

Elements-

0 1 2 3 4

$$K = arr[i] = 1.$$

$$\& arr[j] = arr[0] = 0.$$

So we can easily see that for ~~whole~~ all $i < 5$, inner most for loop will not be executed

Or we can say that the execution is only 1 time.

$$\begin{aligned} \text{Time complexity} &= O(n-1) \\ &\approx O(n) \end{aligned}$$

Q2-

Give suitable example
Also make program of each algorithms.

Quick Sort (Worst case)

10

20

30

40

50

Find element
greater than 10

Find element
smaller than 10
Teacher's Sign.

main (10) 20 30 40 50
i i

swap the main element with j^{th} element.

1 + n
Swap Comparisons

⇒ 20 30 40 50
do step 1 with 20 do step 2 with 20

20 30 40 50 ∞
j 1

do step 3

1 + (n-1)
Swap. comparisons

30 40 50
do step 1 with 30 do step 1 with 30

30 40 50
j i

Teacher's Sign.

do step 3

1 +

n-2

swap

comparisons

do

40

50

do step 1

with 10

do step 2 with 40.

40

50

i

j

Abhinav

1 +

n-3

swap

comparisons

50

If a list contains n element then
time complexity

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

$$= (1 + 1 + \dots + n \text{ times}) + (n + n + \dots + 1)$$

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

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

Teacher's Sign. _____

$$O(n^2)$$

$O(n^2)$ (worst case)

As it does not need an extra space
so it in-place algorithm.

Algo name	Case	Time
Quick sort	Best	$O(n \log n)$
	Avg	$O(n \log n)$
	Worst	$O(n^2)$
Merge sort	Best	$O(n \log n)$
	Avg	
	Worst	
Insertion sort	Best	$O(n)$
	Avg	$O(n^2)$
	Worst	$O(n^2)$
Bubble sort	Best	$O(n)$
	Avg	$O(n^2)$
	Worst	$O(n^2)$

Teacher's Sign. _____

Bubble sort

When

```
void Bubble sort (int arr[], int n)
{
    int i, j;
```

```
    for (i=0; i < n-1; i++)
```

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

```
        {
            if (arr[j] > arr[j+1])
                swap (arr[j], arr[j+1])
        }
    }
```

Worst case

when ~~i=0~~

~~i=0~~

j=0

j=1

j=2

.

:

j=n-2

inner loop run

n times

n-1 ..

n-2 "

2 times

Teacher's Sign. _____

$$\begin{aligned}
 \text{Total} &= n + n-1 + n-2 + \dots + 2 + 1 \\
 &= \frac{n(n+1)}{2} - 1 \\
 &\approx O(n^2)
 \end{aligned}$$

Best case $\approx O(n)$ (when element are already sorted)

Avg case $\approx O(n^2)$

Althina