

D.S Assignment

①

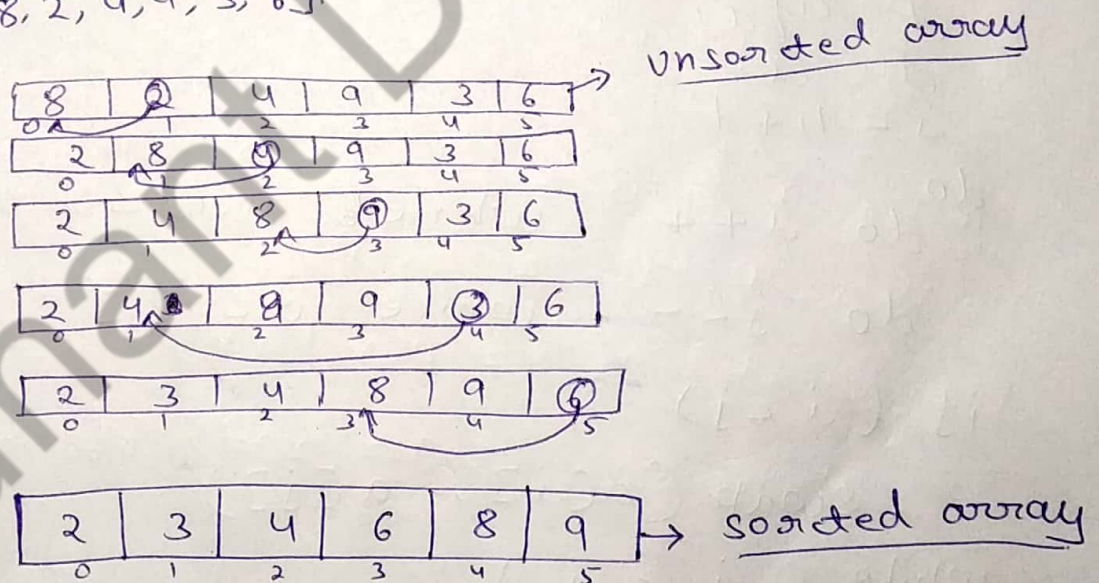
Q1. Analysis on time complexity of insertion sort algorithm.

Let a is an array with n elements

1. read a
2. repeat step 3 to 8 for $i = 1$ to $n-1$
3. $temp = a[i]$
4. $j = i - 1$
5. repeat step 6 to 7 while $temp < a[j]$ and $j >= 0$
6. $a[j+1] = a[j]$
7. $j = j - 1$ // end of 5 loop
8. $a[j+1] = temp$ // end of 2 loop
9. exit

operation followed by insertion sort:

let $a = [8, 2, 4, 9, 3, 6]$



time complexity = $O(n)$. (best case)

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

Time complexity can be reduced by ① & ②

(1) Binary search :- By using a sorted array & using binary search can reduce the complexity to $O(\log n)$.

(2) By using linked list : the complexity of insertion become worst and complexity reduced to $O(n)$.

Q2. Quicksort algorithm

quick_sort($a[l], l, h$)

if ($l < h$)

• $j = \text{partition}(a, l, h)$

quick_sort($a, l, j-1$)

quick_sort($a, j+1, h$)

// end of if.

partition(a, l, h)

$x = a[l]$

$i = l$

$j = h+1$

do

do $i++$, while $a[i] < x$ and $i \leq h$

do $j--$, while $x < a[j]$

if ($i < j$)

swap($a[i], a[j]$)

while ($i < j$)

$a[i] = a[j]$

$a[j] = x$

return j

exit

the complexity of best case of quick sort is $T(n) = O(n \log n)$.

And the complexity of worst case of quicksort is $T(n) = O(n^2)$.

Bubble Sort Algorithm

```
start
for (i = 0 to n)
  for (j = 0 to n-1)
    if (a[j] > a[j+1])
      temp = a[j]
      a[j] = a[j+1]
      a[j+1] = temp
```

end:

The complexity of best case of bubble sort is $T(n) = O(n^2)$.
and worst case $T(n) = O(n^2)$.

Complexity comparison of Quicksort, bubblesort, merge sort and insertion sort:-

Quicksort: $T(n) = O(n \log n)$ (best case) & $O(n^2)$ (worst case).

Bubble sort: $T(n) = O(n^2)$ (best & worst case).

Mergesort: $T(n) = O(n \log n)$ (best case).

Insertion sort: $T(n) = O(n^2)$ (worst) & $O(n)$ (best case).