Tutorial-1

Q.1)

→ Space complexity of Binary search algorithm:

① Iterative - O(1)

② Recursive - O(log n) due to recursive call stack is used.

Q.2)

→ Space complexity for Bubble sort:

① Space complexity is O(1) if array not sorted.

② Space complexity is O(n) if array is already-sorted.

Q.3) 1]

	frequency count
1) for (i = 1 to n) do	→ n
2) for (j = 1 to i) do	→ n(n+1)/2
3) for (k = 1 to j) do	→ n(n+1)(n+2)/6
x = x + 1	

$$f(n) = n + \frac{n(n+1)}{2} + \frac{n(n+1)(n+2)}{6}$$

2]

	f(n)
i = 1	— 1
while (i <= n) do	— n
{	
x = x + 1;	— n
i = i + 1;	— n
}	
	f(n) = 3n + 1

$$f(n) = 3n + 1$$

Page No.
Date

Q.6) calculate Time and space complexity :-

A] Algo Transpose (a,n)
{
for i=1 to n-1 do ⟶ n-1
for j=i+1 to n do ⟶ n-1 x (n-1)
{
t = a[i,j] ; a[i,j] = a[j,i] ; a[j,i] = t;
}
}

Time complexity = $O(n^2)$
space complexity =
additional space is O(1),
Auxilary space used $O(n^2)$ i.e. for space used for matrix.

B] Algo Mult (a,b,c,n)
{
for i=1 to n do —— n
for j=1 to n do —— n x n
{
c[i,j] = 0;
for k=1 to n do —— n x n x n
c[i,j] = c[i,j] + a[i,k] * b[k,j];
}
}

$f(n) = n^3 + n^2 + n$

Time complexity = $O(n^3)$
space complexity = O(1) [additional]
= $O(n^2)$ [Auxillary space]