Let the length of Array = n

(4) i = 1 j = 1, 2, 3, ... n

i = 2 j = 1, 2, 3, ... n

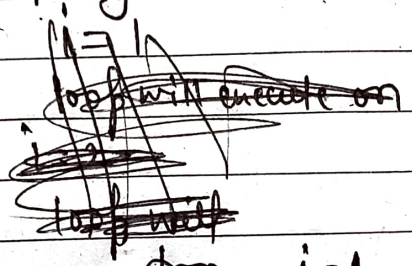
i = Ne (j = 1, ... n)

Since the loop are nested we multiply their complexities:

$$O(N) \times O(N) = O(N^2)$$

Space complexity (O(1)).

(5) As



(5) The loop runs from i=1 to n times. So, the time complexity will be O(n). Space complexity (O(1)).

(6) This is also runs from i=1 to n, So, the time complexity will be O(n). and space complexity O(1) because it only using a single loop variable (i) which uses constant space.

(3) Give time complexity $(O(n^2))$

i) 10 $(O(100) \Rightarrow O(10^2)$

So, the algorithm will perform almost 100 operation.

ii) 50 $(O(n^2)) = O(50^2)$

it will perform 2500 oper.

iii) 100 $(O(n^2)) = O(10^4)$

it will perfom 10000 oper.

~~iv)~~ 500 $(O(n^2)) = O((500)^2)$

it will perform 250000 oper so it will take long t

PAGE No.
DATE

② $O(\log n)$ → i have not study yet but i know because I saw kk vidios and from there I got to know but still i don't know how.

①

⑧
```
i = 1
while (i<n)
    i = i*2
```
∵ i is initialize from 1 and the cond^n is i should be less than n and i = i*2

so, i = 1, 2, 4, 8, 16, ...

$i = 2^0$	$i = 2^1$	$i = 2^2$	$i = 2^3$ .....

Power = 2^k

$2^k < n$

$k < \log_2 n$

(9) As the loop runs from 1 to n

Since the loop iterates n times so,

$T(C) = O(n)$.

(11) This loop is also runs from 1 to n

So, the loop iterates n times so,

$T(C) = O(n)$.

(12) As the outer loop is running from 1 to k and the inner loop is running from 1 to array.length and inner loop is independent of outer loop.

So, Time Complexity will be

as the loops are nested so,

$TC = O(k \cdot \text{length(array)})$

(15) As outer loop runs from i from 1 to length (array)

Let say length of array be n.

So, outer loop runs from 1 to n.

if $i=0$ $j = 1, 2, 3, \ldots n$

$i=1$ $j = 1, 2, 3, \ldots n$

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

PAGE No.
DATE

So,

$TC = O(N^2)$

(18) As the outer loop runs i from 1 to length[matrix1]

So, and inner loop runs j from 1 to length[matrix 2

and k from 1 to length (matrix 1[0])

So,

(TC) = length [matrix 1] * length [Matrix 2[0]] * length [Matrix 1[0]].

(19) As outer loop runs from 1 to n ;

and inner loop runs from 1 to i times;

then $T.C = \frac{n(n+1)}{2}$

$= O(n^2)$

Yaha multiply nahi kiya kiu ki n^2 bda h n se.

(22) As the outer loop run from 1 to length(array)

Let say length(array) = n ;

and inner loop runs from 1 to length(array) = means 1 to n

As they are nested;

So, $(TC) = O(n \times n)$

$= O(n^2)$

PAGE No.
DATE

(23.) As the outer loop runs from 1 to n times.
and inner loop runs from 1 to n times;
and they are nested loop;
So, TC = O(n x n)
= O(n²).

(24) As this loop runs from 1 to length(matrix)/2
Let length(matrix) → n
So,
TC = O(n/2) # SC = O(1)
and we can avoid constants
So, TC = O(n).
the only additional space used is for temporary variables.

(26) As this loop run from 1 to length(array) then
let say length(array) = n
So, TC = O(n) SC = O(1)
variable do not depend on the input size and remains constant.

(22) TC = O(n)
one input variable i and result they will take space. the space of (i) will be constant

(28) ∵ i runs from 1 to n
and i = i * 2
so, we have already solved.
So, TC = O(log n)
SC = O(1).
but the space of result will increase. as loop execute - it will go to n.
So, SC = O(n)

(12) i j ∵ j = j+1 and i = i+1

i		j	
0		(i=i+1)(j=i+1)! . . .	(n-1) times
1	&	1 2 2	(n-2) "
2		2 3 3	(n-3) "
⋮		⋮	⋮
(n-1)		1	

TC = Total iteration = 1+2+3+ . . . . (n-1)

↳ sum of natural nums

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

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

$TC = O(n^2)$

(16) iteration of outer loop = n

" " inner " = n

So,

$TC = O(n+n)$ → here we are doing add n because loops are not nested.

$= O(2n)$

$= O(n)$ we can avoid constant

(20) ~~outer loop = i=~~

As outer loop runs from 0 to $(2^n - 1)$

$O(2^n)$

and

another loop runs from 0 to (n-1)

$O(n)$

$TC = O(n+2^n)$

$= O(2^n)$

$SC = O(n)$

↓

∵ n = length(array)

↓

array size

(25) As outer loop run from 1 to length(matrix)
Let length(matrix) → n

and inner loop run from 1 to n or length(matrix)
No. of iteration → n * n;
So, T(= O(n2)

and transpose is taking space ⇒ n
row = n
S(= O(n*n)
= O(n2)

(29)

i	j
1	1
2	2
⋮	⋮
n	n

Total iteration = n * n
T(= O(n2)
S(= O(1)

(30)

Bonus

(1) As outer loop runs from 0 to $(n!-1)$
and inner loop from n to 1

total iteration: $n! * n$ → this small in comparison to $n!$ so, we can avoid it.

$TC = O(n!)$

(2) The size of duplicate will grow.

So, $SC = O(n)$.

(3) outer loop runs from 0 to $(2^n - 1)$;
and inner from 0 to $(n-1)$

$T.C = O(2^n * n)$ → inner

$= O(2^n)$

(4) same as (3)

=