Time & space complexity

1. time complexity of binary search

$$T(n) = T(n/2) + c$$

$$T(n/2) = T(n/4) + c$$

$$T(n/4) = T(n/8) + c$$

$$\therefore T(n) = T(n/8) + 3c$$

$$= T(n/2^3) + 3 \cdot c$$

$$\vdots$$

$$T(n/2^k) + kc$$

$$\frac{n}{2^k} = 1$$

$$n = 2^k$$

$$\log n = k \log 2$$

$$k = \log n$$

Hence the time complexity = (b) $O(\log n)$

2. Space complexity of the iterative version of binary search is (c) $O(1)$.

because no extra space is required.

3. (d) 500

because if we put 500 in place of n

$500^2 = 250000 < 10^8$

$\therefore$ This is the acceptable largest number.

Arijit
10 July 2024 6:19 pm

4. let length(array) = n

$1 \to n \longrightarrow O(n)$

$1 \to n \longrightarrow O(n)$

∴ Time complexity = $O(n * n) = O(n^2)$

Space complexity = O(1) [∵ we don't using extra space]

```
function Solve(array)
    for i from 1 to length(array)
        for j from 1 to length(array)
            print(array[i] + array[j])
```

5. $1 \to n \to O(n)$

The time complexity $= O(n)$

Space complexity $= O(1)$

[∵ we are using constant space]

```
function Solve(n)
    sum = 0
    for i from 1 to n
        sum = sum + i
    return sum
```

6. Time complexity $= O(n)$

Space complexity $= O(1)$

7. let exponent = n

$T(n) = T(n/2) + c$

$T(n/2) = T(n/4) + c$

⋮

$T(n) = T(n/2^k) + kc$

$\therefore 2^k = n$

$\therefore \log n = k \log 2 \approx k$

$\therefore O(\log n)$

Hence the time complexity will be $O(\log(\text{exponent}))$

space complexity* will be $O(\log(\text{exponent}))$

* using for call stack.

Arijit

10 July 2024 6:20 pm

8.

$1 \to 2 \to 4 \to 8 \to \cdots \to n$

$2^0 \to 2^1 \to 2^2 \to 2^3 \to \cdots \to 2^k$

$\therefore \quad n = 2^k$

$\log n = k \cdot \log 2$

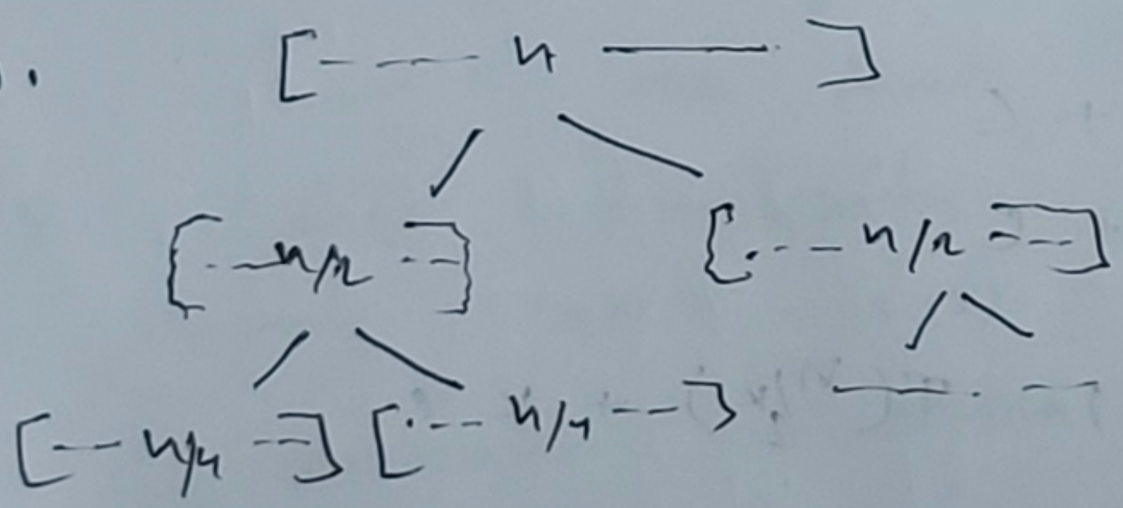
$\therefore$ Time complexity will be $O(\log n)$

space complexity " " $O(1)$.

9. $1 \to n \quad \cdots \quad O(n)$

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

$S.C = O(1)$

10. 

$\therefore T.C. = O(\log n)$

$S.C. = O(1)$

11. $1 \to n \to O(n)$

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

$S.C = O(n)$

$\to$ for call stack.

Arijit

10 July 2024 6:20 pm

12.

let length(array) = n

∴ Time complexity = O(n * k)

Space Complexity = O(1)

13.

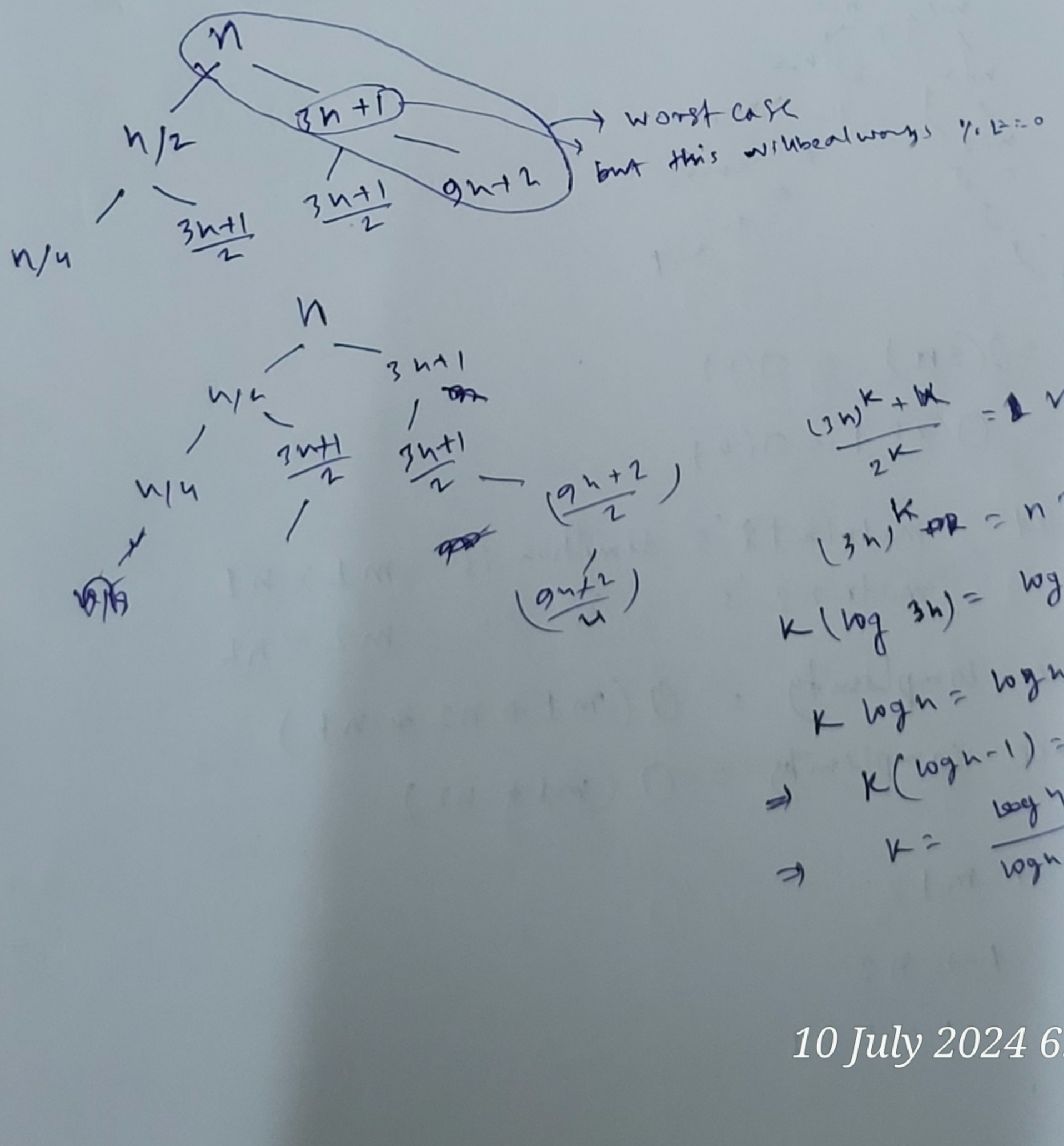
$n + (n-1) + (n-2) + (n-3) + \ldots + 1$

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

∴ Time complexity $= O(n^2)$

Space complexity $= O(1)$

14.



$\frac{(3n)^k + k}{2^k} = n$

$(3n)^k = n\,2^k$

$k(\log 3n) = \log n + k \log 2$

$k \log n = \log n + k$

$\Rightarrow k(\log n - 1) = \log n$

$\Rightarrow k = \frac{\log n}{\log n - 1}$

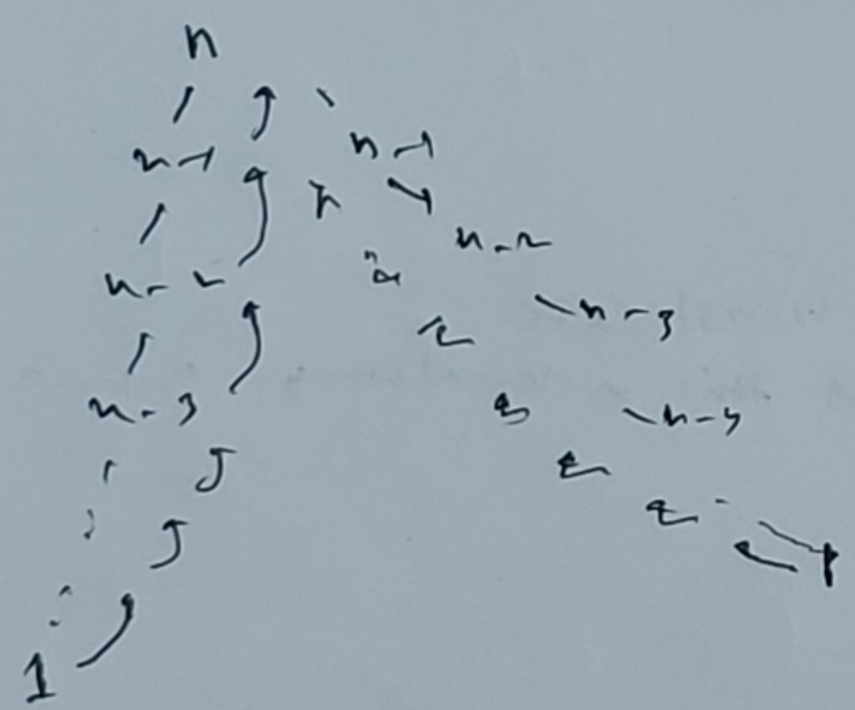
Arijit
10 July 2024 6:20 pm

15. Time complexity = $O(n^2)$
Space complexity = $O(1)$

16. $1 \rightarrow n$ ---- $O(n)$
$+1 \rightarrow n$ ---- $O(n)$

∴ Time complexity = $O(n) + O(n)$
$= O(2n)$ $= O(n)$
Space complexity $= O(1)$

17.



T.C $= O(2n) = O(n)$
S.C $= O(2n) = O(n)$

18. let matrix1's dimation is $m1 \times n1$
matrix2's " " $m2 \times n2$

∴ Time complexity $= O(m1 * n2 * n1)$
Space complexity $= O(m1 * n2)$

$1 \rightarrow m1$
$1 \rightarrow n2$
$1 \rightarrow n1$

Arijit
10 July 2024 6:20 pm

26. T.C = O(length(array))

S.C = O(1)

27. let, length(array) = n

let, no of unique element = m

∴ Time complexity = O(n)

space complexity = O(m)

28. T.C = O(log n)

~~T.C = O(1)~~

S.C = O(1)

29. $1 \longrightarrow n$

$i \rightarrow 1$

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

∴ T.C = $O(n^2)$

S.C = O(1)

30. let length(array) = n

$1 \rightarrow n-k \longrightarrow O(n-k)$

$i+1 \rightarrow i+k \longrightarrow O(k)$

∴ T.C = O((n-k)*k)

S.C = O(1)

Arijit

10 July 2024 6:21 pm

for j $\Rightarrow$ $n + (n-1) + (n-2) \cdots (1)$

$= \frac{n(n-1)}{2} \simeq n^2$

for i $\Rightarrow$ $n^2 + (n^2-1) + (n^2-2) \cdots 1$

$= \frac{n(n-1)(2n+1)}{6} \simeq n^3$

$\therefore$ T.C $= O(n^3)$

S.C $= O(1)$

20. T.C $= O(2^n * n)$

S.C $= O(n)$

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

S.C $= O(n)$

22. T.C $= O(n^2)$

S.C $= O(1)$

23. T.C $= O(n^2)$

S.C $= O(1)$

24. Time complexity $= O(n/2) \simeq O(n)$

Space complexity $= O(1)$

25. let matrix is $n \times m$

$1 \rightarrow n$

$1 \rightarrow m$

$\therefore$ Time complexity $= O(n \times m)$

Space complexity $= O(n \times m)$

Arijit

10 July 2024 6:21 pm

Bonaus MCQs

2. a) O(n)

* if n is the number of duplicate then T.C = O(n)

3. T.C = O(2^n * n)

4. SC = O(2^n * n)

Arijit
10 July 2024 6:21 pm