JUNE						2024
30						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
S	M	T	W	T	F	S

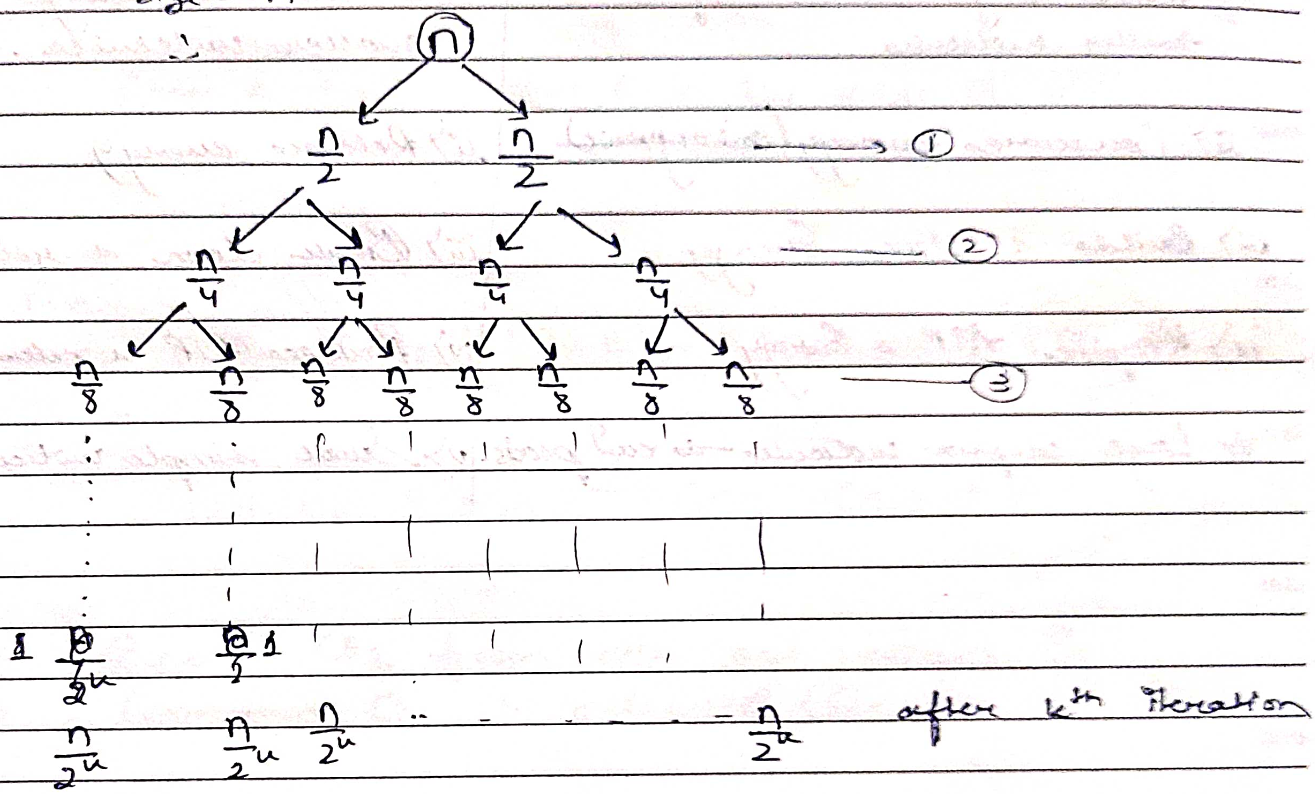
MCQs

09.00

1) b) O(log n)

10.00

Size = n



11.00

12.00

1.00

2.00

3.00

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

4.00

$n = 2^k$

$\log_2 n = k \log_2 2$

5.00

$k = \log_2 n$

6.00

$\therefore$ Time Complexity = $O(\log n)$

Sunday 30

09.00 2) c) O(1) ∵ low, mid, high → The only variables used no extra variable is used to store.

10.00 no. of variables required does not depend on the size of the array.

11.00 3) $O(n^2)$

→ (1) 500 $500^2 = 250000 < 10^8$

12.00 4) function Solve (array)

for i from 1 to length (array) → n

01.00 for j from 1 to length (array) → n

print (array[i] + array[i])

02.00 length (array) → n

~~I(n)~~ = $O(n^2)$

03.00 Time Complexity

Space Complexity = O(1)

04.00 5) function Solve (array)

for i from 1 to length (array) → n

05.00 if arr[i] % 2 == 0

print (arr[i])

else

06.00 print (arr[i] * 2)

T.C = O(n)

Remember S.C = O(1)

02 JULY Tuesday

Week 27 | 184-182 2024

JULY 20

S	M	T	W	T	F
	1	2	3	4	5
7	8	9	10	11	12
14	15	16	17	18	19
21	22	23	24	25	26
28	29	30	31		

09.00

7) function Solve (base, exp)

if exp == 0 → O(1)

return 1

or

half = pow (base, floor (exp/2))

10.00

if exp % 2 == 0

return half * half

else

11.00

return half * half * base

~~T(n) = T(exp)~~

12.00

~~T.C~~ = ~~O(log n)~~

T.C = O(log exp)

01.00

S.C =

$T(exp) = T(\frac{exp}{2}) + C$

$T(\frac{exp}{2}) = T(\frac{exp}{4}) + C$

02.00

$T(\frac{exp}{2^k}) + C$

$2^k = exp$

03.00

O(log exp)

04.00

8) func SumCal (n)

sum = 0

05.00

i = 1

while (i < n)

sum = sum + 1 → O(n)

06.00

i = i * 2

return sum

iteration stops ~~when~~ after kth iteration

09.00 $i = 2^k$

1st iteration $i = 2$
2nd $i = 4$
3rd $i = 8$
!
k^{th} $i = 2^k$

10.00 $2^k \geq n$

$k = \log_2 n$

11.00

$O(\log n) \rightarrow T.C$

12.00 $O(1) \rightarrow S.C$

01.00 9) func. Solve (n)

```
if n == 0
    return 1
```
} O(1)

02.00 res = 1

```
i = 1 → n
    res = res * i
```
} ~~n~~

03.00 return res

T.C = O(n)

04.00 S.C = O(1)

10) func. Solve (arr)

05.00

```
l = 0
r = length(arr) - 1
```
} O(1)

→ T.C = O(log n)
S.C = O(1)

→ O(log n)

```
while (l <= r)
    mid = floor((l+r)/2);
    if arr[mid] == target
        return mid
    elsif arr[mid] < target
        l = mid + 1
    else
        r = mid - 1
```
} O(1)

06.00

return -1

Remember

04 JULY Thursday

Week 27 | 186-180 2024

11) func. solve(n)

09.00 if n<=1

return 1 } O(1)

Sum = 0 → O(1)

10.00 for i = 1 → n → O(n)

Sum = Sum + Solve(i-1)

return Sum

11.00

T.C = O(n)

(2) S.C =

12.00

12) func. solve(arr, k)

i = 1 → k ——→ k

01.00 j = 1 → length(array) ——→ n

print(arr[j]) → O(n)

2.00 length(arr) = n

T.C = O(k x n)

3.00 S.C = O(1)

13) func. Solve(arr)

.00 i = 0

while i < length(arr) ——→ n

j = i+1

00 while j < length(arr) ——→ n-1

if arr[i] + arr[j] == target → O(1)

return true

00 j = j+1

i = i+1

return false.

Remember

AUGUST 2024

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

JULY 05 Friday 2024 /// Week27 | 187-179

i = 0 {1, ... n-1}
i = 1 {2, 3, ... n-1}
.
.
i = n-1

Loop n=1 to n

$n \cdot ((n-1)+(n-2)+(n-3)+\dots+3+2+1)$

$n \cdot (n(n-1)+(n-2)+(n-3)+\dots$ $\frac{n(n-1)}{2}$

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

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

$T.C = O\left(\frac{n^2}{2}\right)$

$S.C = O(1)$

14) function Solve(n)

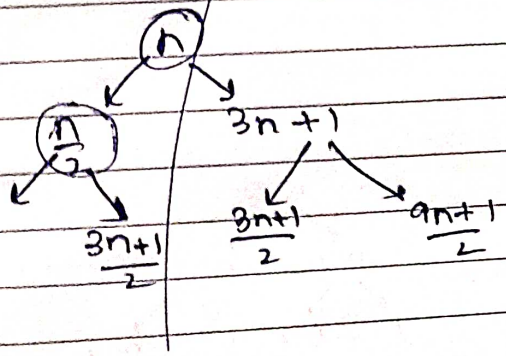
if n == 1
return 1
if n%2 →

even → solve($\frac{n}{2}$) → $O(\log n)$

odd → $(3*n+1) \longrightarrow O(n)$

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

$S.C = O$



15) func. Solve(arr)

for i = 1 → n

j = 1 → i

print(arr[j])

i = 1 j = {1}

i = 2 j = {1, 2}

i = 3 j = {1, 2, 3}

.

.

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

$1+2+3+\dots+n$

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

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

$S.C = O(1)$

06 JULY Saturday

JULY 2024

S	M	T	W	T	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

16) func. solve(arr)

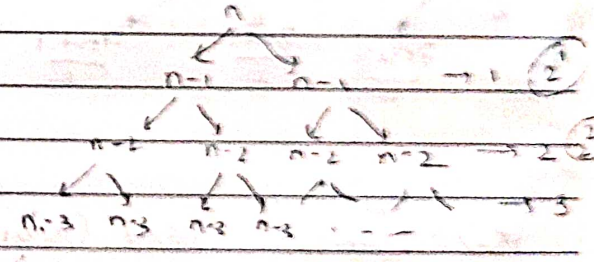
```
res = 0
i = 1 → n                    O(n)
    res = res + arr[i]
j = 1 → n                    O(n)
    res = res - arr[i]
return res
```

T.C = O(n)
S.C = O(1)

17) func. solve(n)

```
if n<=1                      } O(1)
    return 1
a = Solve(n-1)
b = Solve(n-1)
return a+b
```

T(n) = 2T(n-1) + O(1)
T(n-1) = 2T(n-2) + O(1)
T(n-2) = 2T(n-3) + O(1)



=

$\therefore 2^1 + 2^2 + 2^3 + \ldots + 2^{n-1}$

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

$O(2^n)$ → T.C

S.C → O(n)

(Sunday 07)

$1 = (n-(n-1))$

$(n-(n-1))$

Remember

AUGUST 2024

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

JULY
Monday
08
2024 /// Week 28 | 190-176

09.00 18)

```
func. solve (m1, m2)
    for i = 1 → len(m1)
        for j = 1 → length(m2[0])
            sum = 0
            for k = 1 → length(m1[0])
                sum = sum + m1[i][k] * m2[k][j]
            res[i][j] = sum
return res.
```

10.00

11.00

12.00

$T.C = O(\text{length}(m1) \times \text{length}(m2[0]) \times \text{length}(m1[0]))$

01.00

$S.C = O(\text{length}(m1) \times \text{length}(m2[0]))$

↳ resultant matrix dimensions

02.00

19)

```
func. solve(n)
    i = 1 → n
        j = 1 → i
            k = 1 to j
                print(i, j, k)
```

03.00

04.00

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

05.00

$$\sum_{i=1}^{n} \frac{i^2+i}{2} = \frac{1}{2}\sum_{i=1}^{n} i^2 + \frac{1}{2}\sum_{i=1}^{n} i$$

06.00

$$\frac{1}{2}\cdot\frac{n(n+1)(2n+1)}{6} + \frac{n(n+1)}{4} = O(n^3) \rightarrow T.C$$

Remember

09 JULY Tuesday

Week 28 | 191-175 2024

JULY

S	M	T	W	T
7	8	9	10	
14	15	16	17	
21	22	23	24	
28	29	30	31	

S.C = O(1)

09.00

~~20~~ 21) func. Solve (n)

if n == 0

return 0

10.00

return n + Solve (n-1)

11.00

T.C = O(n)

S.C = O(n)

12.00

22) func. Solve (arr)

i = 1 → length (arr)

01.00

j = 1 → length (arr)

if arr[i] < arr[j]

Swap (arr[i], arr[j])

02.00

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

S.C = O(1)

03.00

23) func. Solve (n)

for i = 1 → n

04.00

j = 1 → n

if i == j

print (i, j)

05.00

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

S.C = O(1)

06.00

AUGUST 2024

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

JULY
Wednesday
10

24) func. solve (arr)
for i = 1 → n/2 → O(n)
swap (arr[i], arr[n-i])

T.C = O(n)
S.C = O(1)

25) func. Solve [matrix]
T. = []
i = 1 → n (no. of rows)
row = []
j = 1 → n[0] (no. of cols) → m times
row. append (matrix[i][j])
T. append (row)

return T

T.C = O(n×m)
S.C = O(n×m)

26) func. Solve (arr)
c = 0
i = 1 → n
if arr[i] > 0
c = c+1

return c

T.C = O(n)
S.C = O(1)

Remember

11 JULY Thursday

Week 28 | 193-173 2024

JULY 2024

S	M	T	W	T	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

09.00 27) func. Solve (arr)
res = []
i = 1 → n → m elements not present in arr[i] & res
if not res.contains (arr[i])
10.00 res.append (arr[i])
return res.

11.00 T.C = O(n)
S.C = O(m)

12.00 28) func. solve (n)
i = 1
(i < n)
01.00 i = i * 2

S.C = O(1)

i = 1, 2, 4, 8, 16, ... 2^k

$2^k \geq n$
$2^k > n$
$\log n = k$
$O(\log n)$ → T.C

03.00 29) func. Solve (arr)
i = 1 → n
j = i
04.00 (j > 0 & arr[j-1] > arr[j])
swap (arr[j], arr[j-1])
j = j - 1

05.00 i = 1, j = 1
i = 2, j = 2, 1
i = 3, j = 3, 2, 1
06.00 ⋮
i = n, j = n, n-1, ... 3, 2, 1

S.C = O(1)

$(n)+(n-1)+(n-2)+\dots 3+2+1 = \frac{n(n+1)}{2} = \frac{n^2}{2} + \frac{n}{2} = O(n^2)$ → T.C

Remember

AUGUST 2024

				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31
S	M	T	W	T	F	S

JULY 12
Friday
2024 /// Week 28 | 194-172

09.00

30) func. Solve (arr, k)
i = 1 → n-k → n-k
maxi = arr[i]
for j = i+1 → i+k → k time
if (arr[j] > maxi)
maxi = arr[j]
return (maxi)

10.00

11.00

O((n-k) x k)
O(nk - k²)

n > k

12.00

∴ O(nk) → T.C
S.C = O(1)

01.00

1) (c) O(n!)

02.00

2) (a) O(n)

03.00 3) O(n * 2^(n-1)) = ~~O(n *~~ O(2^n * n)

4) ~~O(n * 2^n)~~ O(2^n)

04.00

05.00

06.00