Name - Adhyan Dhyani
Class - 4th Sem, CSE
Roll no - 20-H.

Date. ________
Page No. ________

DAA - Tutorial - 2

Q1

Ans 1

$j=1 \quad i=1$

$j=2 \quad i=1+2=3$

$j=3 \quad i=3+3=1+2+3$

$\vdots$

$j=k \quad i=1+2+3+\cdots k$

as $i<n$

Sum of k consecutive integers $= \frac{k(k+1)}{2}$

$\therefore \frac{k(k+1)}{2} < n$

$\frac{k^2+k}{2} < n$

Afters removing const.

$k^2 < n \Rightarrow k < \sqrt{n}$

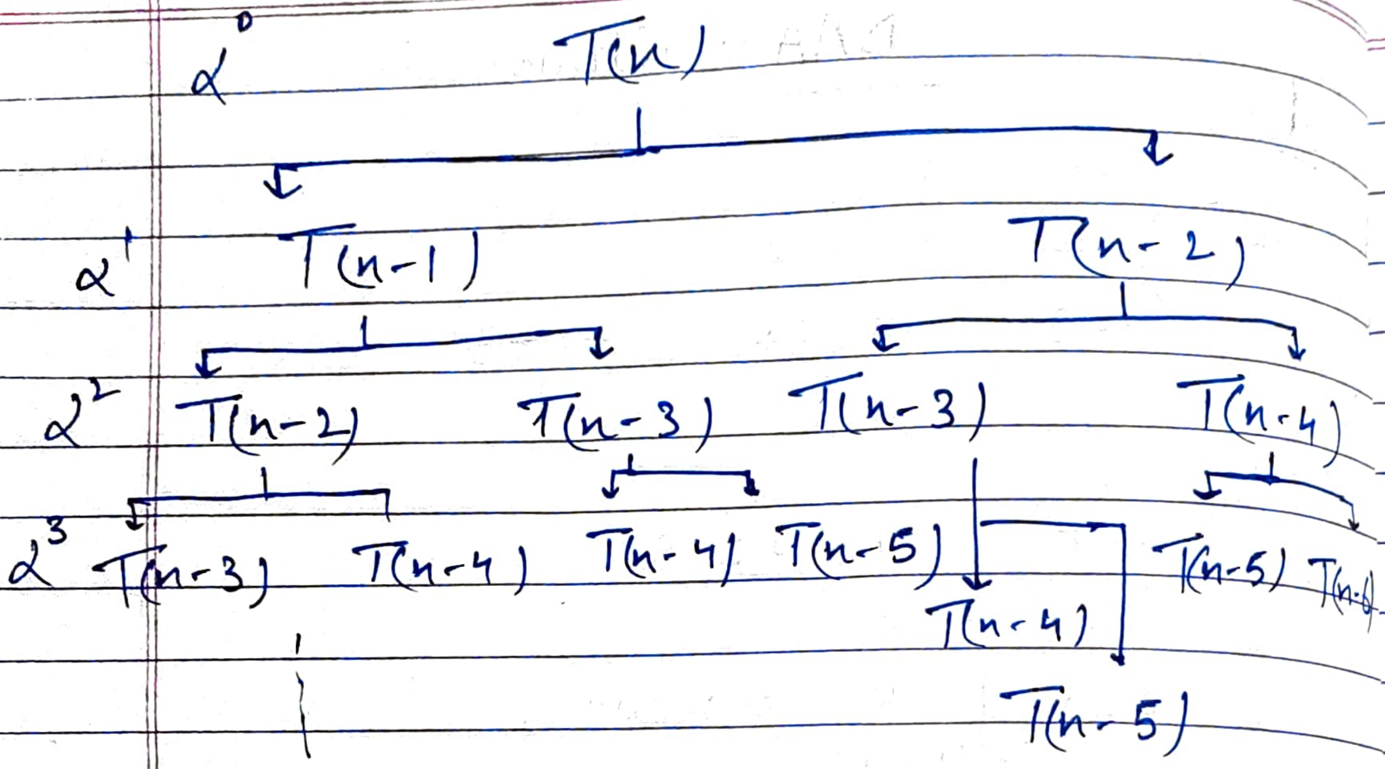
$\therefore T(n) = O(\sqrt{n})$ Ans

Q2

Ans2

Recurrence relation of fibonacci series

$T(n) = T(n-1) + T(n-2) + 1$



2^n $\quad T(n) = 2^0 + 2^1 + 2^2 \cdots\cdots 2^n$

$$S = \frac{a(r^n - 1)}{r-1}$$

$$a = 1 \quad r = 2 \Rightarrow \frac{1(2^n - 1)}{1} = 2^n - 1$$

$T(n) = O(2^n)$ Ans

space complexity = $O(n)$.

Q4

Sol^n

$$T(n) = 2T(n/2) + cn^2$$

Using master's

$$a = 2, \quad b = 2$$

$$n^{\log_b a}$$

$$n^{\log_2 2} = n^1$$

$$f(n) > n^2$$

$$T(n) = f(n)$$

$$\Rightarrow \quad O(n^2) \text{ Ans}$$

Q5

Sol^n

i	j	
1	1, 2, 3, . . .	n times
2	1 . . ——— .	$n/2$ times
3	1, 2	$n/3$ times
⋮	⋮	
n	1 time	

$$T(n) = n + n/_2 + n/_3 + n/_4 \cdots 1$$

$$= n\left(1 + 1/_2 + 1/_3 \cdots\cdots 1/_n\right)$$

$$\underline{T(n) \rightarrow} = \underline{n(\log n)}$$

Q6
Sol^n

$$T(n) = 2, 2^k, 2^{k^2} \cdots 2^{k^{\log k(\log n)}}$$

we know $2^{k^{\log k(\log n)}} = 2^{\log n} = n$

∴ Total iteration will be :-

$$= \log k(\log(n))$$

$$T(n) = O(\log k(\log n))$$ Ans.

Q8
Sol^n a) $100 < \log(\log n) < \log(n) < \log^2 n < \sqrt{n} < n < n\log n < n^2 < 2^n < 4^n < 2^{2n} < \log(n!) < n!$

b) $1 < \log(\log(n)) < \sqrt{\log n} < \log n < \log 2n$
$< 2\log n < n < 2n < 4n < n\log n < n^2$
$< \log(n!) < n! < 2(2^n)$.

c) $96 < \log_8(n) < \log_2(n) < 5n < n\log_6(n)$
$< n\log_2 n < n! < \log n! < 8^{2n}$.

Q3
Sol^n i) $O(\log n)$

→ Binary search
→ Finding largest / smallest number in binary search tree

ii) $O(n\log n)$

→ Merge sort
→ leap sort
→ quick sort

iii) $\log(\log n)$

```
for ( int i=2 ; i<=n ; i = pow (i,e))
  {
     // O(1) expressions
  }
```

iv) n^3

```
for (int i=0 ; i<n ; i++ )
  {
     for (int j=0 ; j<n ; j++ )
     { for (int k=0 , j<k ; k++ )
          {
             // O(1) expression
          }
     }
  }
```

Q7

Sol^n When the given array is already sorted and you are talking the 1st or the last element as key element then it is worst case of quicksort.

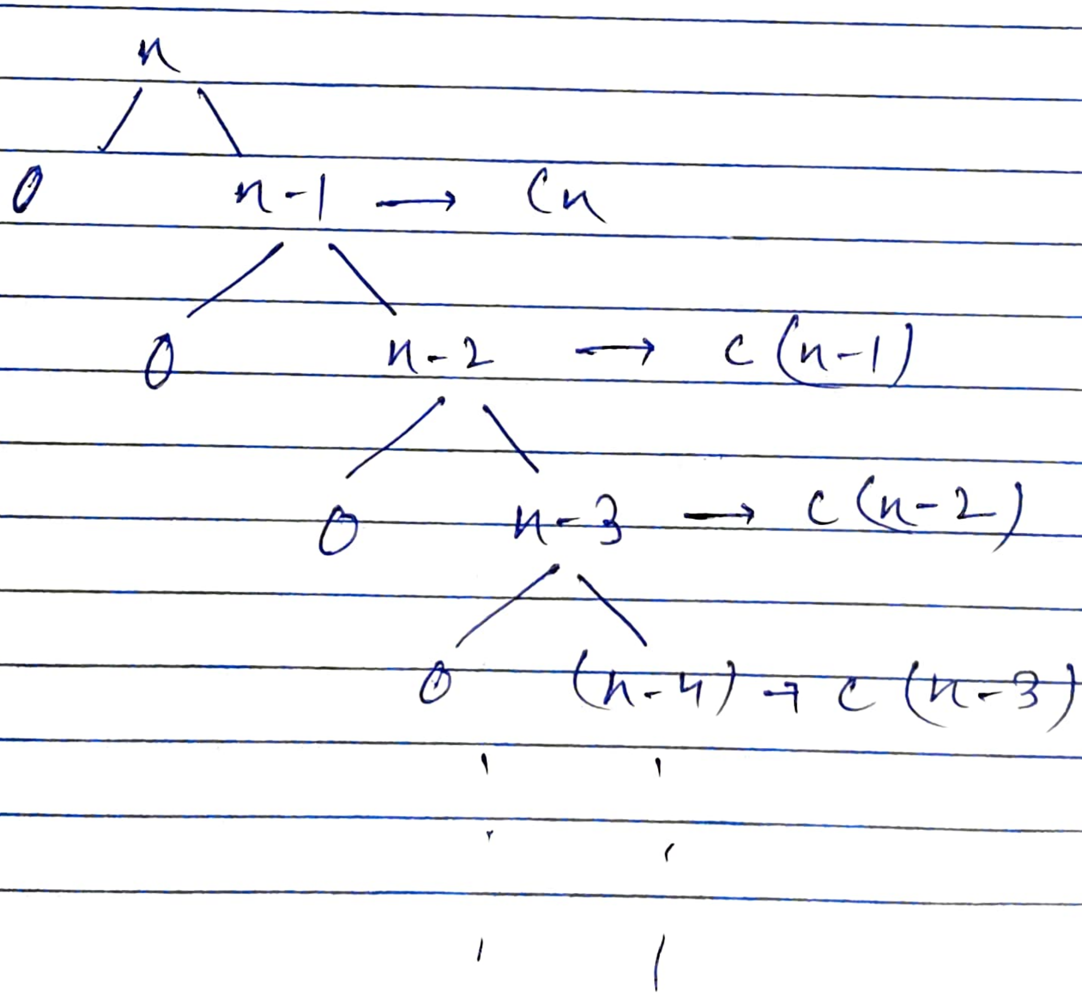
$$T(n) = T(n_1) + T(n_2) + cn$$

For worst case $\qquad$ c = constant

$n_1 = 0 \qquad n_2 = n-1$

$$T(n) = T(0) + T(n-1) + cn$$

$$T(n) = T(n-1) + cn$$



$$c\left[n+(n-1)+(n-2)+(n-3)+\cdots 1\right]$$

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

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

Time complexity $= O(n^2)$.