

TUTORIAL-2 → DAA

1.) void fun(int n)

{ int j=1; i=0;

while(i < n);

i = i + j;

j++;

}

Aish Goyal

Section F

23

Time complexity → $O(\sqrt{n})$.

1st time = $i = 1$

2nd time = $i = 3$ ($i = 1 + 2$).

3rd time $i = 6$ ($i = 1 + 2 + 3$).

!

n^{th} time = $i = \frac{i(i+1)}{2} = x^2 < n$

$x = \sqrt{n}$.

2.)

let $T(0) = 1$.

* $\text{fib}(n) = \text{fib}(n-1) + \text{fib}(n-2)$

$\text{fib}(n)$:

if $n \leq 1$

return 1

return $\text{fib}(n-1) + \text{fib}(n-2)$.

Time complexity :-

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

$$= 2T(n-2) + C.$$

$$T(n-2) = 2 * (T(n-2-2) + C) + C.$$

$$= 2 * (2T(n-2) + C) + C$$

$$= 4T(n-2) + 3C.$$

$$T(n-4) = 2 * (4T(n-2) + 3C) + C.$$

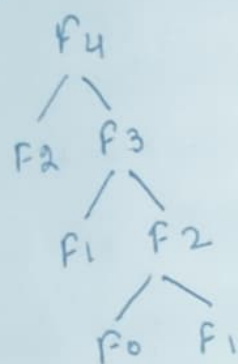
$$= 8T(n-3) + 7C$$

$$= 2^K * T(n-K) + (2^K - 1)C$$

$$n - K = 0 \Rightarrow n = K \Rightarrow K = n$$

$$\begin{aligned} T(n) &= 2^n * T(0) + (2^n - 1)C \\ &= 2^n * 1 + 2^n C - C \\ &= 2^n(1 + C) - C \\ &= 2^n \\ &= O(2^n). \end{aligned}$$

Space Complexity:- Space is proportional to the maximum depth of the recursion tree.



Hence the space complexity of Fibonacci recursive is $O(N)$.

③ Merge Sort - $n \log n$.

for time complexity: - n^3

We can use three nested loops

for (int i = 0; i < n; i++)

{ for (int j = 0; j < n; j++)

{ for (int k = 0; k < n; k++)

{ some $O(1)$ expressions

}

}

→ for time complexity - $\log(\log n)$.

for (int i = 2; i < n; i = power(i, j))

{

" some $O(1)$ expression

}

where K is constant.

for time complexity, $n \log n$

int fun (int n)

{ for (i=1; i <= n; i++)

{ for (j=1; j <= n; j+=i

{ some $O(1)$ expression

}

}

Q:-4.

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

using master's method

$$T(n) = aT(n/b) + f(n)$$

$$a \geq 1, b \geq 1, c = \log_b a$$

$$c = \log_2^2 = 1$$

$$f(n) > n^c$$

$$T(n) = \Theta(f(n))$$

$$\Rightarrow \Theta(n^2)$$

Q:-5

for $i=1 \rightarrow j=1, 2, 3, 4, \dots, n$ (run for n times)

for $i=2 \rightarrow j=1, 3, 5, \dots$ (run for $n/2$ times)

for $i=3 \rightarrow j=1, 4, 7, \dots$ (run for $n/3$ times)

$$T(n) = n + n/2 + n/3 + n/4 + \dots$$

$$n(1 + 1/2 + 1/3 + 1/4 + \dots)$$

$$n \int_1^n \frac{1}{x} dx \Rightarrow n \int_1^n \frac{1}{x} dx \Rightarrow \log x \Big|_1^n$$

$$T.C = n \log n$$

Q:-6.

for first iteration $i=2$

second iteration $i=2^k$

third iteration $i=(2^k)^k = 2^{k^2}$

i th iteration $i=2^k$ loop ends at $2^k = n$

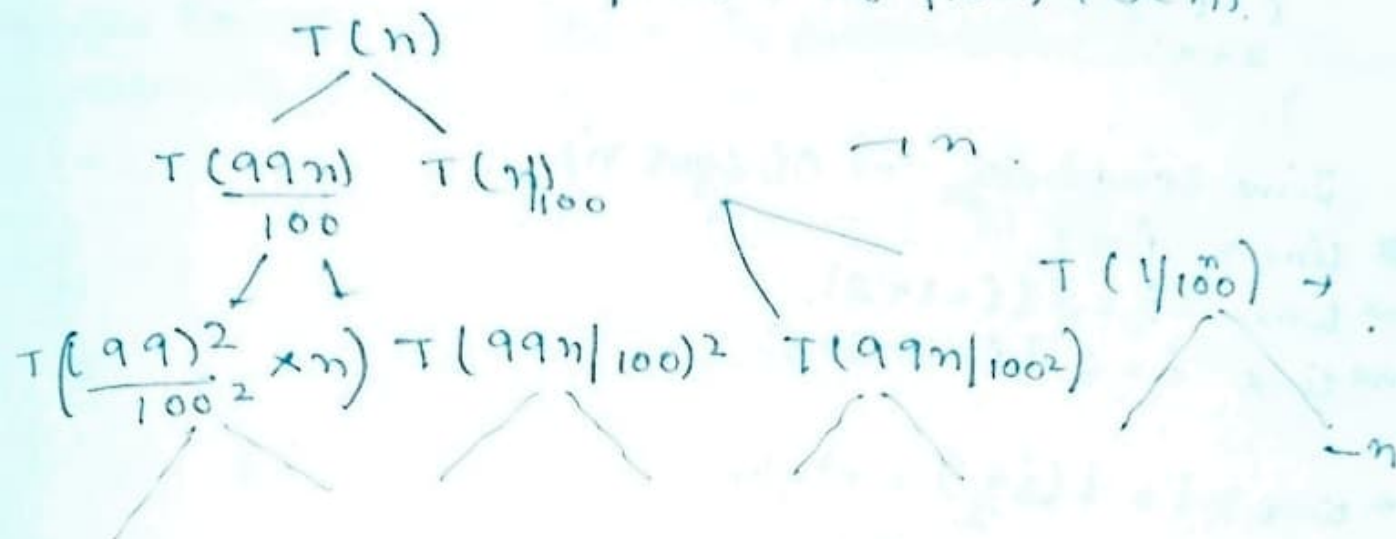
$$\text{apply } \log n = \log 2^{k^2} = k^2 = \log n \Rightarrow i = \log_c(\log n)$$

Q:-7 . 99 to 1 in Quick sort

when pivot is either from front or end always

$$\text{So } T(n) = T(99n/100) + T(n/100) + O(n)$$

$$T(n) = T(99n/100) + T(n/100) + O(n)$$



$$\frac{n}{(99|100)^K} = 1.$$

$$n = (99|100)^K$$

$$\log n = k \log 99/100$$

$$R = \log n$$
$$\frac{100}{99}.$$

$$\therefore TC = n * \log_{100/99}(n).$$

Ques:- a

a.) $100 < \log \log(n) < \log^2 n < \log n < \log n!$
 $< n < n \log n < n^2 < 2^n < n^n < 2^n (2^n n) < n!$

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

c.) $96 < \log_3 n < \log_2 n < 5n < n(\log_3 n) < n(\log_2 n) < \log(n!) < 8n^2 < 7n^2 < n! < 8^{27}$