- Harsh Saini
- 2002103
- D
- 05

Date / /

Tutorial-2 → DAA

1)

```
void fun (int n)
{
    int j=1 ; i=0;
    while (i<n)
        i = i+j ;
        j++;
}
```

time complexity → O(sqrt n)

1st time ⇒ i = 1
2nd time ⇒ i = 3 (i = 1+2)
3rd time ⇒ i = 6 (i = 1+2+3)
⋮
nth time ⇒ $i = \frac{i(i+1)}{2} = x^2 < n$

$x = sqrt(n)$

2)

let T(0) = 1
* fib(n) = fib(n-1) + fib(n-1)

```
Fib(n):
    if n<=1
        return 1
    return fib(n-1) + fib(n-2)
```

time complexity :-

Date / /

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

$$T(n-2) = 2^* [T2 (n-2-2) + c] + c$$
$$= 2^* (2T(n-2) + c) + c$$
$$= 4T(n-2) + 3c$$

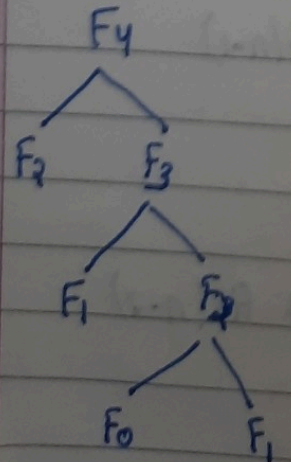
$$T(n-4) = 2^* (4T(n-2) + 3c) + c$$
$$= 8T(n-3) + 4c$$
$$= 2^k + T(n-k) + (2^k - 1)c$$

$n-k = 0 \Rightarrow n = k \Rightarrow k = n$

$$T(n) = 2^n * T(0) + (2^n - 1)c$$
$$= 2^n * 1 + 2^n c - c$$
$$= 2^n (1+c) - c$$
$$= 2^n$$
$$= O(2^n)$$

Space Complexity:- Space is proportional

is the maximum depth of the recursion ~~tree~~. tree.



Hence the space complexity of fibonacci recursive is O(N)

Date / /

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
      }
```

Date / /

Q4 =

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

using master's method

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

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

$c = \log_2^2 = 1$

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

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

$\Rightarrow \theta(n^2)$

Q5 =

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

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

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

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

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

$n \int_1^n 1/x \Rightarrow n \int_1^n dx/x \Rightarrow \log x]_1^n$

T.C $= n \log n$

Q6 =

for first iteration $i = 2$

second iteration $i = 2^{*k}$

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

$\vdots$

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

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

Q7 =

Date / /

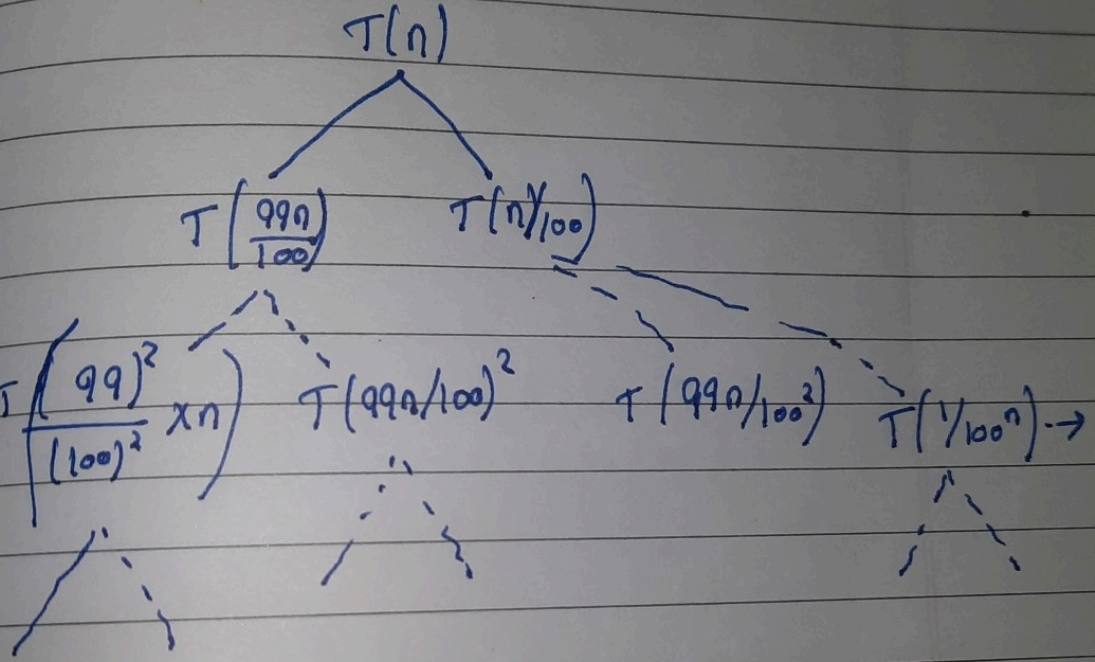
99 to 1 in quick sort

when pivot is ~~where~~ either from front or end always

so,

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

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



$$\frac{n}{(99/100)^k} = 1$$

$$n = (99/100)^k$$

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

$$k = \frac{\log n}{\frac{100}{99}}$$

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

Date / /

Q8 a. $1 < \log\log(n) < \log^2 n < \log n < \log n! < n < n\log n < n^2 < 2^n < 4^n < 3^n(2^n n) < n!$

b.