Date
Page

Assignment 2.

1) void fun(int n)

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

1st time : $i=1$

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

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

:

nth time : $i=$

2) → $fib(n) = fib(n-1)+fib(n-2)$

Let $T(0)=1$

fib(n) : If n<=1
return 1
return fib(n-1) +fib (n-2)

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

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

[let $T(n-1) \simeq T(n-2)$]

$$T(n-2) = 2^{*}\,(2T(n-2-2)+c)+c$$
$$= 2^{*}\,(2T(n-4)+c)+c$$
$$= 4T(n-4)+3c$$

$$T(n-4) = 2^{*}(4T(n-4)+3c)+c$$
$$= 2^{k}\,T(n-k) + (2^{k}-1)c$$

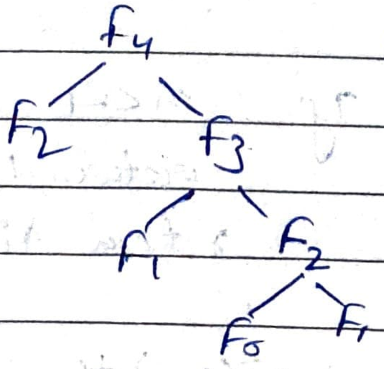
$n-k=0$, $n=k$, $k=n$

$$T(n) = 2^{n}\,T(0)+(2^{n}-1)c$$
$$= 2^{n} + 2^{n}c - c$$
$$= 2^{n}(1+c) - c$$

$O(2^{n})$

Space Complexity:-

The space is proportional to the maximum depth of recursion tree.



S.Comp. $O(N)$

Date
Page

3] i) Merge Sort (n log n)

ii) For Time Comp. (n^3)

```
For (int i=0 ; i<n ; i++)
    for (int j=0 ; j<n ; j++)
        for (int k=0 ; k<n ; k++)
            O(1) expression.
```

(iii) log (log n)

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

iv) n log n

```
int fou(int n
{
    for (int i=1 ; i<=n ; i++)
    {
    for (j=1 ; j<=n ; j+=i)
    {
        O(1)
    }
    }
```

Date
Page

4) $T(n) = 2T\left(\frac{n}{2}\right) + cn^2$ $\qquad T\left(\frac{n}{2}\right) \geq T\left(\frac{n}{4}\right)$

Master's method.

$T(n) = aT\left(\frac{n}{b}\right) + f(n)$

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

Comp. n^c & $f(n)$

we get $c = \log_2 2 = 1$

$f(n) > n^c$

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

$\Theta(n^2)$

5)
```
int fun(int n) {
  for(int i=1; i<=n; i++)
    for(int j=1; j<n; j+=i)
      O(1)
```

when $i=1 \to j = 1, 2, 3, \ldots n$
$i=2 \to j = 1, 3, 5, \ldots n/2$
$i=3 \to j = 1, 4, 7, \ldots n/3$

$T(n) = n + n/2 + n/3 + \ldots n$
$= n\left[1 + \frac{1}{2} + \frac{1}{3} + \ldots\right]$

$$= n \int_1^n \left(\frac{1}{x}\right) = n \int_1^n \frac{dx}{x}$$

$$= n[\log n]_1^n$$

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

(6) for first iteration $i = 2$

2^{nd} iteration $i = 2^{\wedge K}$

3^{rd} iteration $i = (2^K)^K = 2^{K^2}$

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

$2^{K^i} = n$

apply log $\log n = \log 2^{K^i}$

$$\Rightarrow K^i = \log n$$

$$\log K^i = \log n$$

$$i = \log_c(\log n)$$

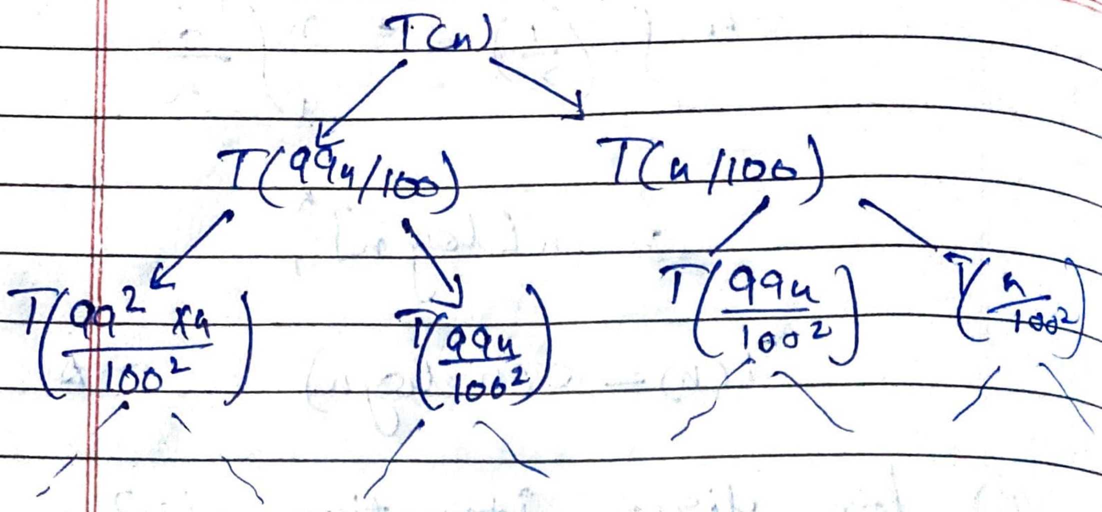
(7) 99 to 1 in Quick Sort
possibly when pivot is last either
from front or end always.

So

$$T(n) = T(99n/100) + T(1/100) + O(n)$$

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

Date
Page



$$n \times \left(\frac{99}{100}\right)^K = 1$$

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

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

$$n = \left(\frac{100}{99}\right)^k$$

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

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

(8) 100 < log log(n) < $\log^2 n$ < log n < log n!
< n < n log n < n^2 < 2^n < ~~4~~ 4^n <
2^(2^n)
< n!