1

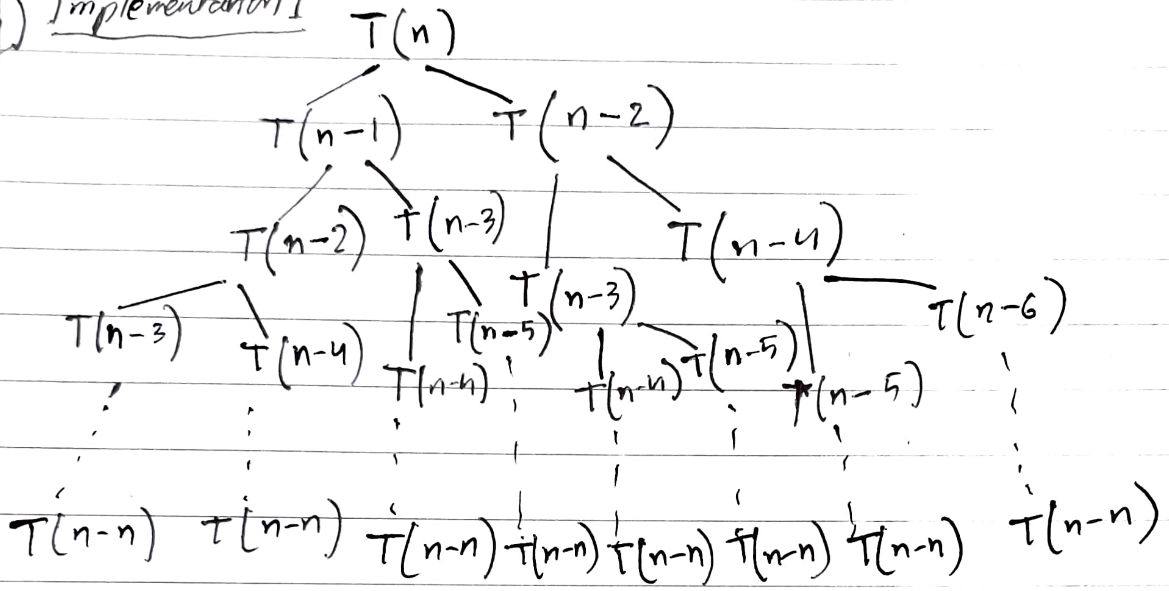
ANIQA IBNAT JISA

ID : 20201136

Section : 4

TASK 2

2) Implementation 1



node = n	height = n
height = log n	node = 2^n

node	height
n	log n
2^n	$2^{\log_2 n}$
2^n	n

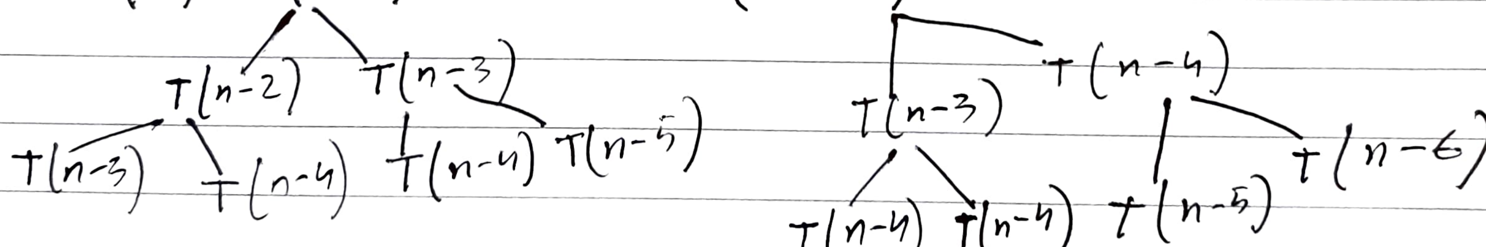
Time complexity : $O(2^n)$

Implementation-2

```
def fibonacci_2(n)
    fibonacci_array = [0,1]
    if n<0:
        print("Invalid input!")      } O(1)
    elif n<=2:
        return fibonacci_array[n-1]  } O(1)
    else:
O(n) {  for i in range(2,n):
            fibonacci_array.append(fibonacci_array[i-1]+
                                   fibonacci_array[i-2])
        return fibonacci_array[-1]

n = int(input("Enter a number: "))
nth_fib = fibonacci_2(n)
print("The %d-th fibonacci number is %d" % (n, nth_fib))
```

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



function runs two times here

$T(n) = 2n$

$= O(n)$

TASK 4

4

```
for i=0 to n-1 } O(n)
    for j=0 to n-1 } O(n)
        for k=0 to n-1
            C[i,j] += A[i,k] * B[k,j]   } O(n)
        end for
    end for
end for
```

∴ Time complexity $= O(n^3)$

5 TASK 5

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

$a=1, b=2, c=1, k=1$

$b^k = 2^1 = 2$

$b^k > a$

Time complexity $= O(n^k)$

$= O(n)$

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

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

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

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

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

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

$T(n) = \cancel{T}\ T(n-k) + (n-k) + (n-(k-1))) + n-(k-2)$
$+ \cdots\cdots + (n-3) + (n-2) + (n-1)$

let $n-k = 1$

$n = k+1$

$T(n) = T(1) + (k+1) - k + (k+1-k+1) +$
$(k-1-k+2) + \cdots (n-2) + (n-1)$

$= 0 + 1 + 2 + 3 + \cdots\cdots (n-2) + (n-1)$

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

$= O(n^2)$

3 $T(n) = T(n/3) + 2T(n/3) + n$

$T(n) = 3T(n/3) + n$

$a = 3, \; b = 3, \; c = 1, \; k = 1$

$b^k = 3^1 = 3 = a$

$\therefore b^k = a$

Time complexity $= O(n^k \log n)$

$= O(n \log n)$

4 $T(n) = 2T(n/2) + n^2$

$a = 2, \; b = 2, \; c = 1, \; k = 2$

$b^k = 2^2 = 4$

$b^k > 0$

Time complexity $= O(n^k)$

$= O(n^2)$ (proved)