Q

$$\text{Sum}(L-R) = \underset{x}{PS[R]} - \underset{y}{PS[L-1]} \quad (L>0)$$

$$(x-y) \% K = 0$$

$$x = nK + r_1$$
$$y = mK + r_2$$
$$x-y = K(n-m) + (r_1 - r_2)$$

if

$$(x-y) \% K = 0$$
$$\Rightarrow r_1 - r_2 = 0$$
$$\Rightarrow r_1 = r_2$$

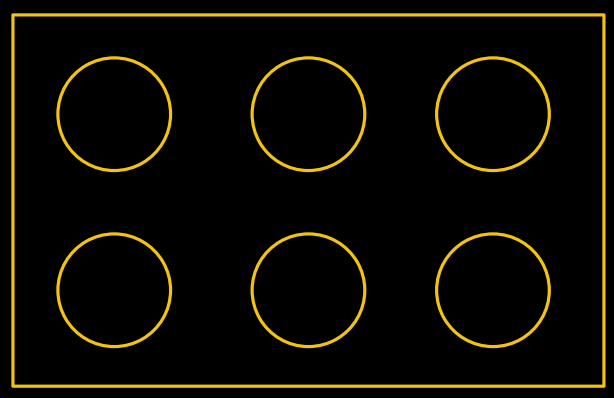
A :

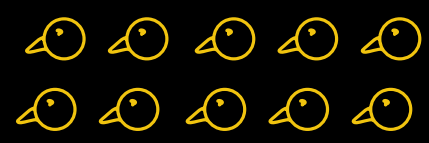
PS :

$\downarrow$ $PS[i] \% K \rightarrow r$

$O(N)$

$a \% N = [0, N-1]$





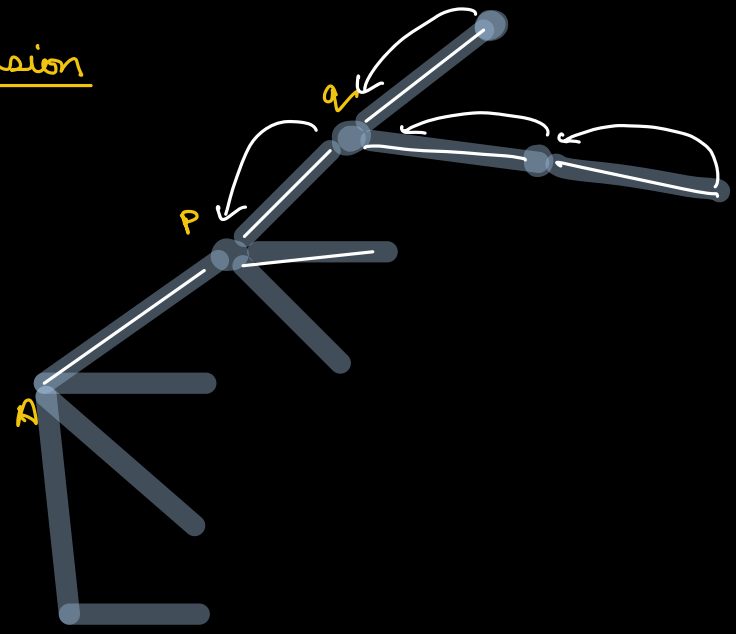
0	1	2	3	4	5	6	7	8
2	3	1		7	8	5	6	4

[0, 8]

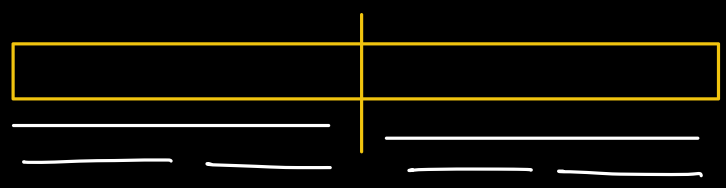
$O(1)$

```
boolean subArraysum (int[] A){

return true;

}
```

Recursion



B



100

BS
IS
SS

$\} \; O(N^2)$

$N = 100 \rightarrow 10000$

$N = \underline{50} \rightarrow 50^2 + 50^2 = 5000$

$\quad 100$

$\overline{5100}$

$N = 25 \rightarrow 25^2 \times 4 = 2500$

$\quad 200$

$\overline{2700}$

- → Binary Trees
- → Graphs
- → Heaps, tries, Seg Tree

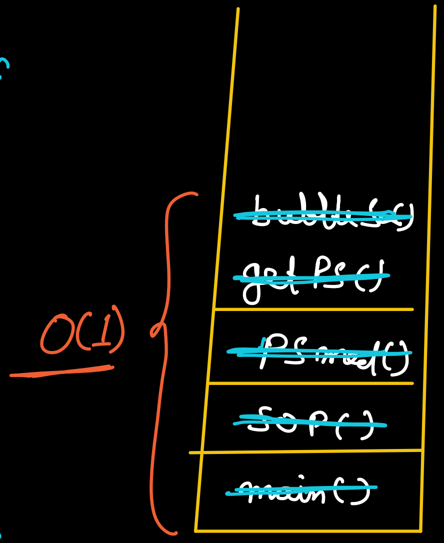
- → Backtrack
- → Dynamic Program

$$\text{fact}(N) = N \times \text{fact}(N-1)$$

$$\Rightarrow \text{Sum}(N) = N + \text{Sum}(N-1)$$

```
int sum (int N) {

        ret N + sum(N-1);
}
```

```
int     PS med ( int[] A) {
               int[] PS = getPS ( A);
               bubbleSort (PS);
               ret   PS[ PS.len /2 ] ;
}


main ( ) {
        int[] A = {  _ ___  };
        SOP ( PS med (A) );
}   →
```



```
int sum (int N) {
    if (N==1) ret 1;
    ret N + sum(N-1);
}
```

TC : O(N)

SC : O(N)

sum(4) → 10

ret 4 + sum(3) → 6

sum(3)

ret 3 + sum(2) → 3

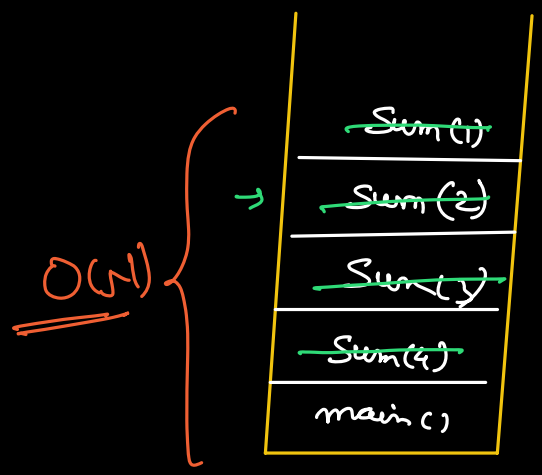
sum(2)

ret 2 + sum(1) → 1

sum(1)

ret 1

O(N)



Q N^{th} Fibo no.

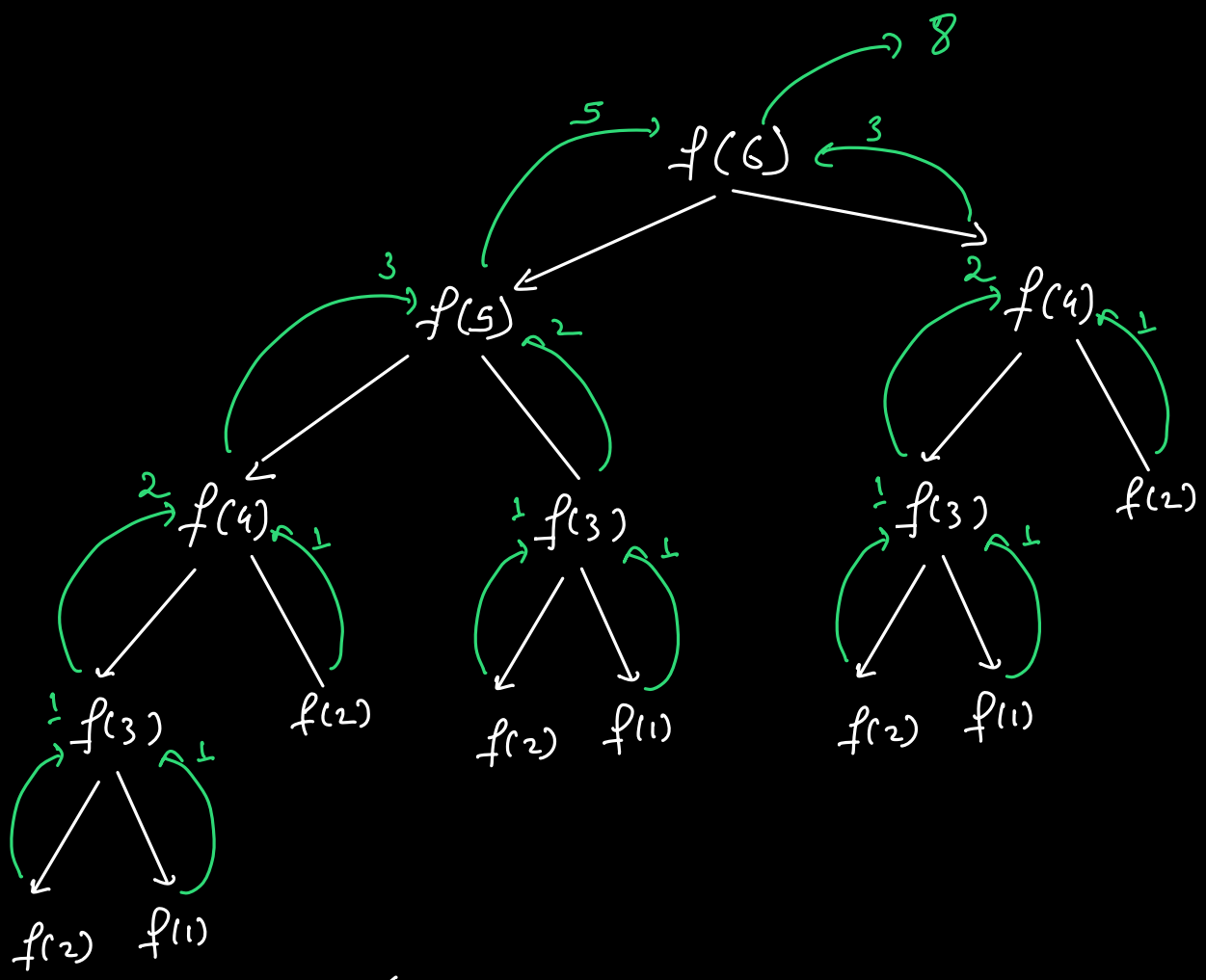
1	2	3	4	5	6	7	
1	1	2	3	5	8	13	----

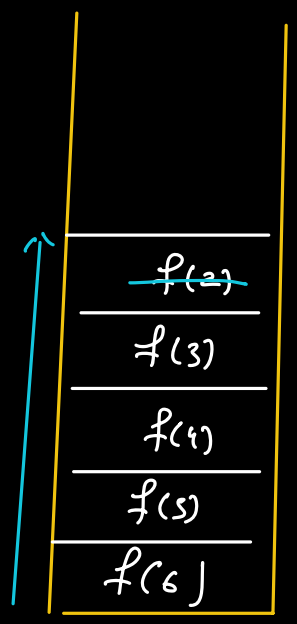
fib(x) → x^{th} fib no.

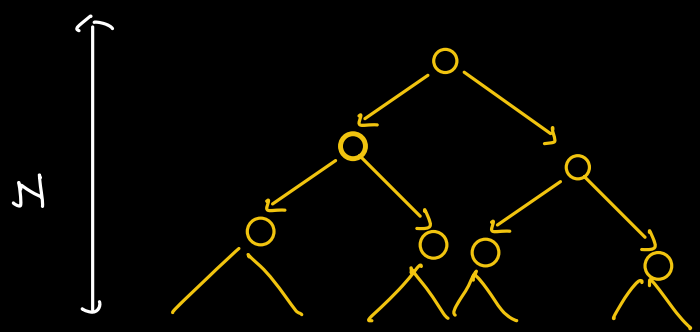
```
int fib (int N) {
    if ( N<=2) ret 1;
    ret fib(N-1) + fib(N-2);
}
```

TC : $O(2^N)$

SC : O(N)







→ 1
→ 2
→ 4
→ 8
→ 16
→ 32
→ 64
⋮

$1 + 2 + 4 + 8 \cdots 2^{N-1}$ } $O(2^N)$

{ } → $O(N)$

```
int sum (int N) {
        if (N==1) ret 1;
        for (i=0; i<N; i++) SOP(i);
    ret N + sum(N-1);
}
```

$O(N^2)$

TC : No. of fn calls
×
TC of each fn call

SC : Max stack height
×
SC of each fn call

Q Given a string check if it is a palindrome

a b c b a

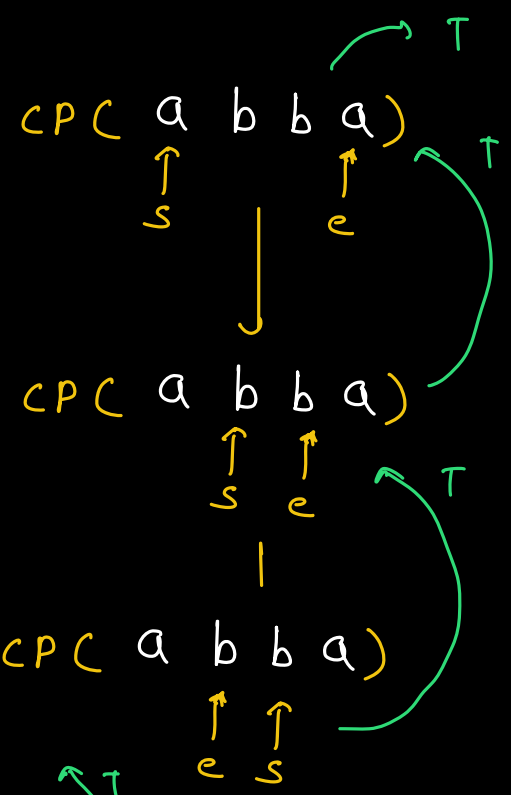
s e

```
while (s<=e){
}
```

checkPalindrome (s , s, e)

```
boolean checkPalindrome (Str A, int s, int e){

    if (s>=e) ret true;
    if (A.charAt[s] != A.charAt[e])
        ret false;

    ret checkPalindrome (A, s+1, e-1);
}
```

a b b a



a b c b a