

## 5. Multiple Recursion Calls

Multiple Recursion Calls :

$f(1)$

{

$f(1)$   
 $f(1)$  } 2 times

}

Fibonacci

0 1 1 2 3 5 8 13 21 34 ...

↙ ↙ ↙ ↙

0<sup>th</sup> 1<sup>st</sup> 2<sup>nd</sup> 3<sup>rd</sup> . . . . .

↗  $f(5)$

N  $\rightarrow$   $f(N) \rightarrow$   $n^{\text{th}}$  fibonacci nos.

$f(3) \rightarrow 2$

$f(4) \rightarrow 3$

$f(5) \Rightarrow f(4) + f(3)$

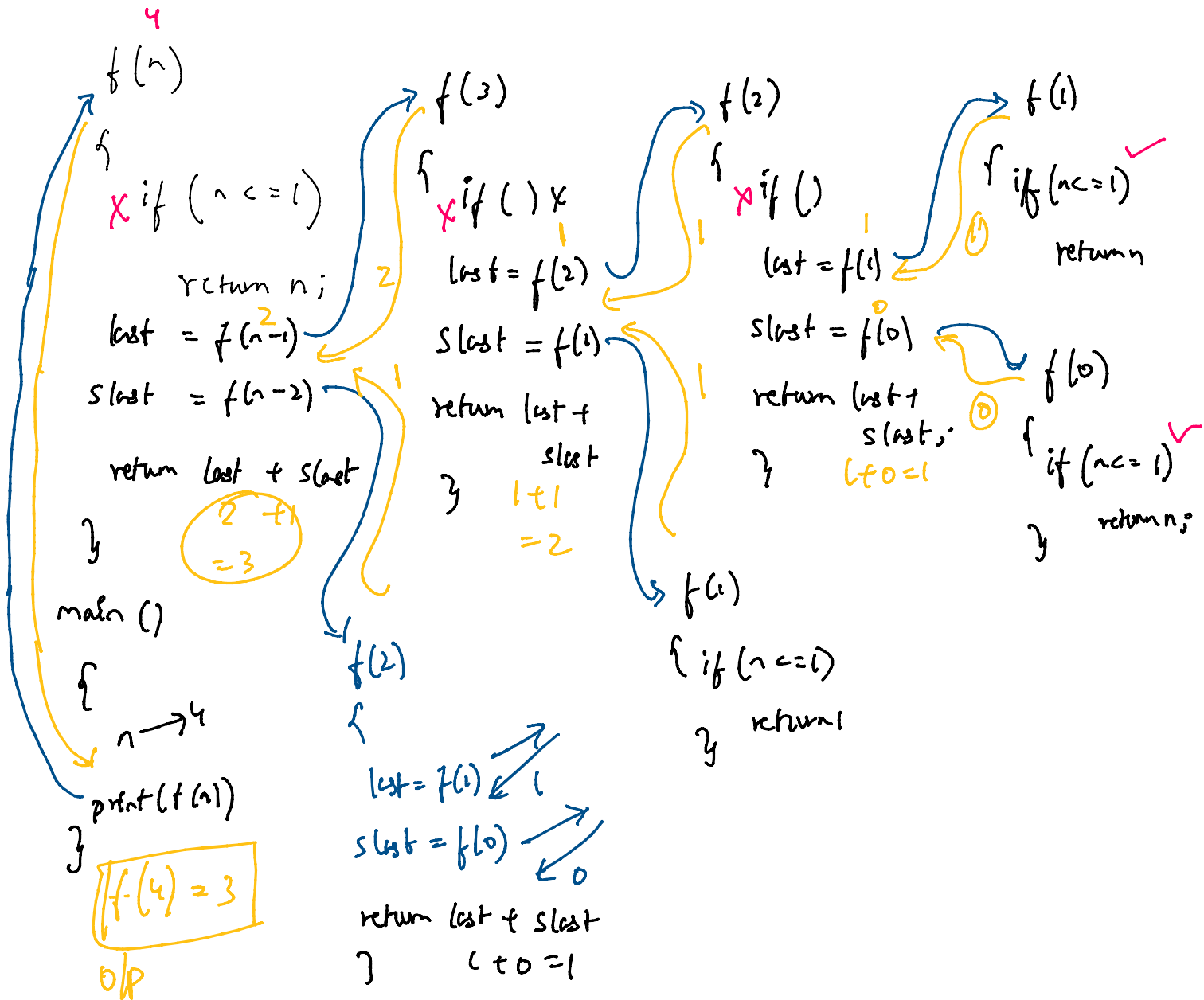
Iterative Approach

```

f[0] = 0    f[1] = 1
for (int i = 2 → n)
    f[i] = f[i-1] + f[i-2]

```

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



$f(1)$

{

f(1)

f(1)

f(1)

}

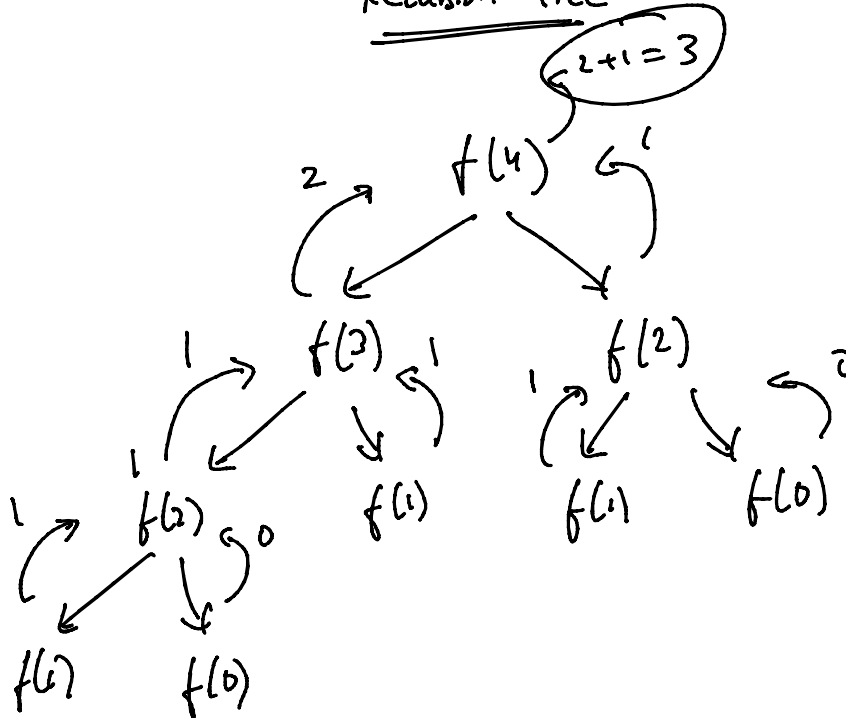
f(1) →

f(1) →

f(1) →

One is ended, next comes.

## Recursion Tree



```

#include <bits/stdc++.h>
using namespace std;
int fibo(int n){
    if ( n <= 1)
        return n;
    int last = fibo(n - 1);
    int secondLast = fibo(n - 2);
    return last + secondLast;
}

int main(){
    cout << fibo(4) << endl;
    return 0;
}
  
```

1

output.in

1 3  
2

ished in 3.9s]

$T.C \Rightarrow O(2^n)$  Exponential in nature

② ② ②  
n n-1 n-2

$$\frac{2}{1} \quad \frac{2}{n-1} \quad \frac{2}{n-2}$$