

```

tabnine: test | explain | document | ask
function add(a,b) : number {
  return a + b;
}

tabnine: test | explain | document | ask
function helper(a,b){
  console.log(a,b);
  let sum = add(a,b);
  console.log(a,b,sum);
}
helper(10,20)

```

10 20
10 20 30
Where to return?

Call Stack

```

function fibonacci(n: number): number {
  if (n == 1) return 0;
  if (n == 2) return 1;
  let fnm1 = fibonacci(n - 1);
  let fnm2 = fibonacci(n - 2);
  return fnm1 + fnm2;
}

let ans = fibonacci(5);
console.log(ans);

```

3

```

function printer(n: number) : void {
  if (n == 0) {
    return;
  }
  console.log(n);
  printer(n-1);
  console.log(n);
}

printer(5);

```

1) Call → stack pe add
2) kahun a kamse jao stack
3) kaha se call krni
4) All line execute
5) Top = execute

Call Stack

Stack top → execution happening
5
4
3
2
1
0

Factorial
 $n!$
 $5! = 1 \times 2 \times 3 \times 4 \times 5$
 $10 = 1$
 $10! = 1 \times 2 \times 3 \times 4 \times 5 \times 6 \times 7 \times 8 \times 9 \times 10$

- Call Stack - Understood
- Components - Code
- False explanation logic?

Fibonacci Numbers
 $f(n) = f(n-1) + f(n-2)$
 $f(0) = 0, f(1) = 1, f(2) = 1, f(3) = 2, f(4) = 3, f(5) = 5, f(6) = 8, f(7) = 13, f(8) = 21, f(9) = 34, f(10) = 55$

nth fib. no.?

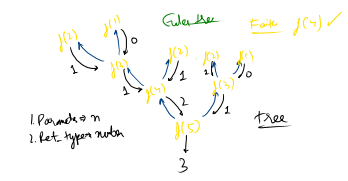
```

function printer(n: number) : void {
  if (n == 0) {
    return;
  }
  console.log(n);
  printer(n-1);
  console.log(n);
}

printer(5);

```

• Faith
• expect



```

function findFirstIdx(idx: number, target: number, arr: number[]) : number {
  if (arr[idx] == target) return idx;
  let ans = findFirstIdx(idx+1, target, arr);
  return ans;
}

```

target = 2
if (idx == arr.length) return -1

last idx
Start from start idx, last occurrence of target