

Diagram illustrating the recursive process of calculating Fibonacci(5):

- $\text{fib}(5)$  calls  $\text{fib}(4)$  and  $\text{fib}(3)$
- $\text{fib}(4)$  calls  $\text{fib}(3)$  and  $\text{fib}(2)$
- $\text{fib}(3)$  calls  $\text{fib}(2)$  and  $\text{fib}(1)$
- $\text{fib}(2)$  calls  $\text{fib}(1)$  and  $\text{fib}(0)$
- Base cases:  $\text{fib}(1) = 1$ ,  $\text{fib}(0) = 0$
- Return values are propagated back up the call stack:
  - $\text{fib}(2)$  returns 1
  - $\text{fib}(3)$  returns 2
  - $\text{fib}(4)$  returns 3
  - $\text{fib}(5)$  returns 5

```
function destroyer(...args) {
  let [repeatedArr, ...args] = args;
  console.log(repeatedArr);
  console.log(args);
  args.forEach(ele => {
    repeatedArr.filter((num) => num !== ele);
  });
  return repeatedArr;
}

console.log(destroyer([1, 2, 3, 1, 2, 3], 2, 3));
```

$\text{arr} = [1, 1, 2, 2, 2, 3]$   
 $\text{sa} = [1, 2, 2, 2, 3]$   
 $\text{ans} = [2, 3]$   
 $\text{ele} = 2 \neq$

```
whatIsInAName({ "apple": 1, "bat": 2 }, { "bat": 2 }, { "apple": 1, "bat": 2, "cookie": 2 })
```

[illegible]

50m  
 ans  $\leftarrow [\{a:1, b:2\}, \{a:1, b:1, c:3\}] \rightarrow \text{and only}$   
 $\{a:1, b:2\}$   $\nwarrow$  ~~filtering~~  $\nwarrow$  ~~for~~  $\nwarrow$  ~~in~~  $\nwarrow$  has O(nlogn)  
 $\{a:1, b:2\}$   $\nwarrow$  101 == 1  $\nwarrow$  ~~true~~  $\nwarrow$   $\{a:1, b:2\}$   
 $\nwarrow$  if (true) {  
color false  
 }  
 $\nwarrow$  return false  
 }  
 $\nwarrow$  win  
 }

aerob

Algorithm + way  $\rightarrow$  algorithmway

①  $a_k e \rightarrow a_k e + c \rightarrow a_k e + a_j \rightarrow \underline{a_k e a_j}$

str[0] = a c i o v try at  
else - try

die - ~~the~~  
 01234  
 glove  
 110011  
 line of 2  
 012352  
 speaker  
 1100  
 electronics  
 shoe (0, 2)  
 shoe 2  
 cons gl  
 10 wot - one right + gl  
 under cons way