Adding Runtimes

The big idea:

If your algorithm is in the form “do this, then, when you’re all done, do that” then you add the runtimes.

— *Cracking the Coding Interview*

```
logTwoArrays(["A", "B", "C", "D"], [1, 2, 3, 4])

function logTwoArrays(arrA, arrB){
  for(let i=0; i<arrA.length; i++){
    console.log(arrA[i])
  }
  for(let j=0; j<arrB.length; j++){
    console.log(arrB[j])
  }
}
```

In this example, the runtime is influenced by the size of arrA and the size of arrB, so the Big O notation is O(arrA + arrB). This may be abstracted to O(N + M).

Multiplying Runtimes

The big idea:

If your algorithm is in the form “do this for each time you do that” then you multiply the runtimes.

— *Cracking the Coding Interview*

```
1   logTwoArrays(["A", "B", "C", "D"], [1, 2, 3, 4])
2
3   function logTwoArrays(arrA, arrB){
4     for(let i=0; i<arrA.length; i++){
5       console.log(arrA[i])
6       for(let j=0; j<arrB.length; j++){
7         console.log(arrB[j])
8       }
9     }
10  }
11
```

In this example, we iterate through arrB for every item in arrA, so the runtimes are multiplied together making the Big O Notation O(arrA * arrB), which may be abstracted as O(N * M)