Adding Runtimes

3 min

Sometimes, a program has so much going on that it’s hard to find the

Preview: Docs Loading link description

runtime

of it. Take a look at the

Preview: Docs Loading link description

pseudocode

program that first prints all the positive values up to N and then returns the number of times it takes to divide N by 2 until N is 1.

```
Function that takes a positive integer N:
  Set a variable i equal to 1
  Loop until i is equal to N:
    Print i
    Increment i

  Set a count variable to 0
  Loop while N is not equal to 1:
    Increment count
    N = N/2
  Return count
```

to Clipboard

Rather than look at this program all at once, let’s divide into two chunks: the first loop and the second loop.

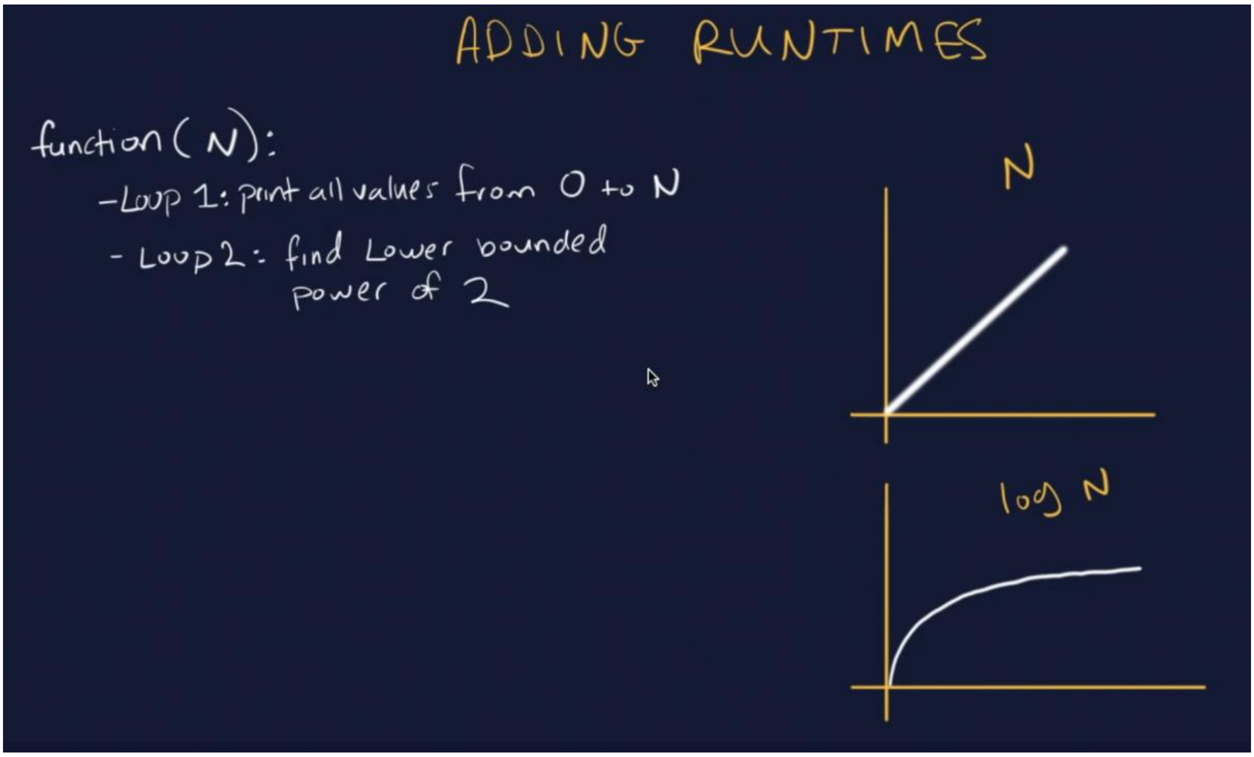
- In the first loop, we iterate until we reach N. Thus the runtime of the first loop is Θ(N).
- However, the second loop, as demonstrated in a previous exercise, runs in Θ(log N).

Now, we can add the runtimes together, so the runtime is Θ(N) + Θ(log N).

However, when analyzing the runtime of a program, we only care about the slowest part of the program, and because Θ(N) is slower than Θ(log N), we would actually just say the runtime of this program is Θ(N). **It is also appropriate to say the runtime is O(N) because if it runs in Θ(N) for every case, then it also runs in Θ(N) for the worst case. Most of the time people will just use big O notation.**

Instructions

Play the video to go through an example of adding runtimes.



ADDING RUNTIMES

function(N):

- Loop 1: print all values from 0 to N
- Loop 2: find Lower bounded power of 2