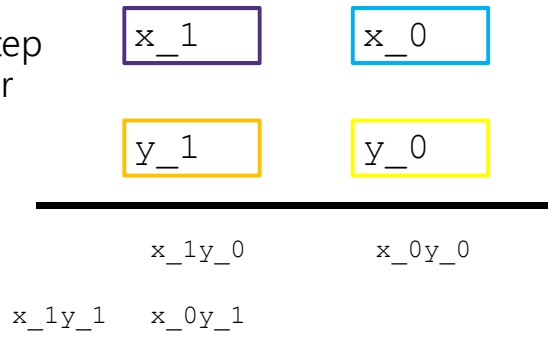
Classic Application

If n bits is too many to multiply in one step (e.g. it's more than one byte, or whatever your processor does in one cycle)

Recurse! Running time?

$$T(n) = \begin{cases} 4T\left(\frac{n}{2}\right) + O(n) & \text{if } n \text{ is large} \\ O(1) & \text{if } n \text{ fits in one byte} \end{cases}$$

Why $O(n)$? It takes $O(n)$ time to add up $O(n)$ bit numbers – they have $O(n)$ bytes!
(Why have you never seen this before? We assumed our numbers were ints, where the number of bytes is a constant)



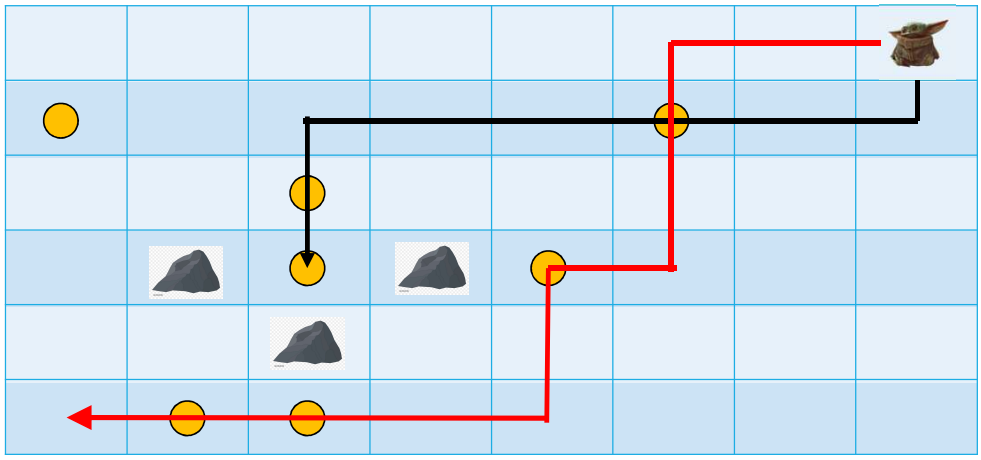
Overall running time is $\Theta(n^2)$

Baby Yoda Searching

Baby Yoda has to get from the upper-right corner to the lower left.
His cradle will only let him go left and down.
He can't get past the rocks (too high) – he has to go around them (but still only going left and down)

As usual…he's hungry.
He wants to eat as many frog eggs as possible on the way.

Baby Yoda Searching



Black path: get stuck. Invalid.

Red path: valid! And optimal (no path collects more than 4 eggs.)

Let `OPT(i,j)` be the maximum number of eggs we can get on a legal path from `(i,j)` to `(0,0)` (including the egg in `(i,j)` if there is one)

What recursive calls do we need?

Don't try to divide & conquer, think closer to home…

We have to decide whether to go down or left…