Wednesday, February 16, 2022 9:53 AM

Factorials

- ↳ solve w/ loops OR recursion ^ has to reach base case
- ↳ sometimes can't use loops

$$\text{factorial} = \begin{cases} 1 & n=0 \\ 1 & n=1 \\ n_n(n-1))! & n>0 \end{cases}$$

?

base case:

$n=1$ $f(n)=1$

Other n's $f(n)=f(n-1)\cdot n$

↳ indefinite

```
def factorial 2(n): // input can be any positive #
    if n=1:
        return 1 // tiny doll, definite & solid → base case
    elif n == 0:
        return 0
    else
        return(n • factorial2(n-1)) // calling function inside self
```

O(3 or 4)

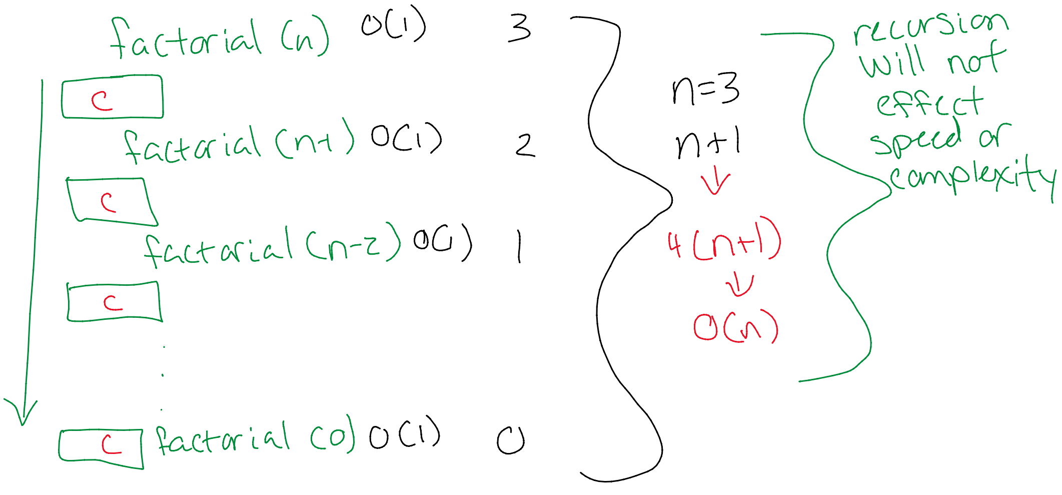
O(c)

↓

constant

↳ Runs until factorial ↳ base case

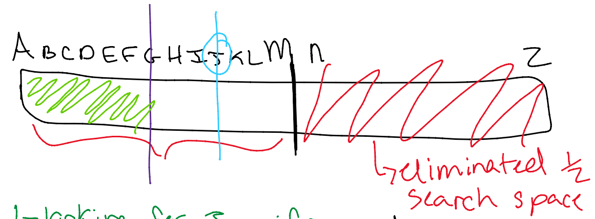
↳ opens up name spaces for unknowns



recursion will not effect speed or complexity

Binary search => O(logn)

↳sort -> know where you to find it



↳eliminated ½ search space

↳looking for Jennifer
is J>m or J<m } ↳reduces search space ½ each time
$n \to 26$

↳is J>G or J<G } $\frac{n}{2} \to 13$

↳is J>J or J<J }
↳neither, equal
↳found J
J==J

$$\underbrace{\frac{n}{4} \to 6 \Rightarrow \frac{6}{2} \Rightarrow 3 \Rightarrow \frac{3}{2} \Rightarrow \boxed{1}}_{O(\log n)}$$

Jake
Jennifer } move on to next letter
Josh

↳Solve w/ recursion

↳reset min & max => ÷2

↳base case -> return what you were looking for

↳base case can be min or max

↳can have multiple conditions