Recursion

Very much like mathematical induction.

Inductive Proofs: trying to prove some statement for all natural #'s.

① Base case: explicitly demonstrate truth for small value of n. Say $n = 1$.

② Assume truth for all values $< n$. Then, show this would $\Rightarrow$ truth for n.

③ ① & ② together $\Rightarrow$ truth for all $n \in \mathbb{N}$.

Proof for $n = 5$ look like?

① $\Rightarrow t(1)$, $t(1) \xRightarrow{②} t(2) \xRightarrow{②} t(3)$

$\cdots \xRightarrow{②} t(5)$

Recursive functions: just like induction.

① Prove your function works on small input. Say with an if statement: `if (n == 1) return answer.`

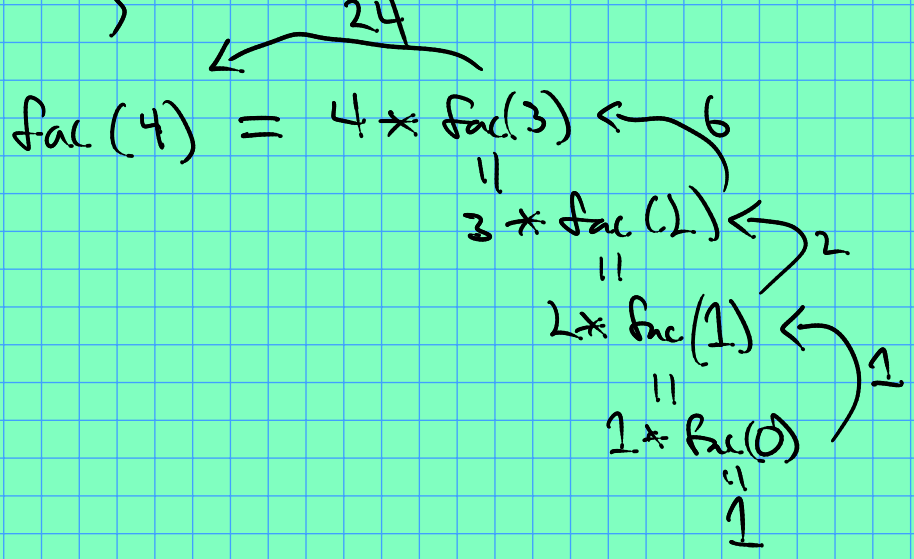
② Assume your function works for any input

of size $< n$, & build the solution for inputs of size n by calling the same function on smaller inputs ($< n$).

③ Conclude your function works for all inputs.

Examples: write a recursive function to compute $n!$.

```
int fac(int n) {
    if(n==0) return 1;
    return n * fac(n-1);
}
```



Naively, this looks worse than a loop:

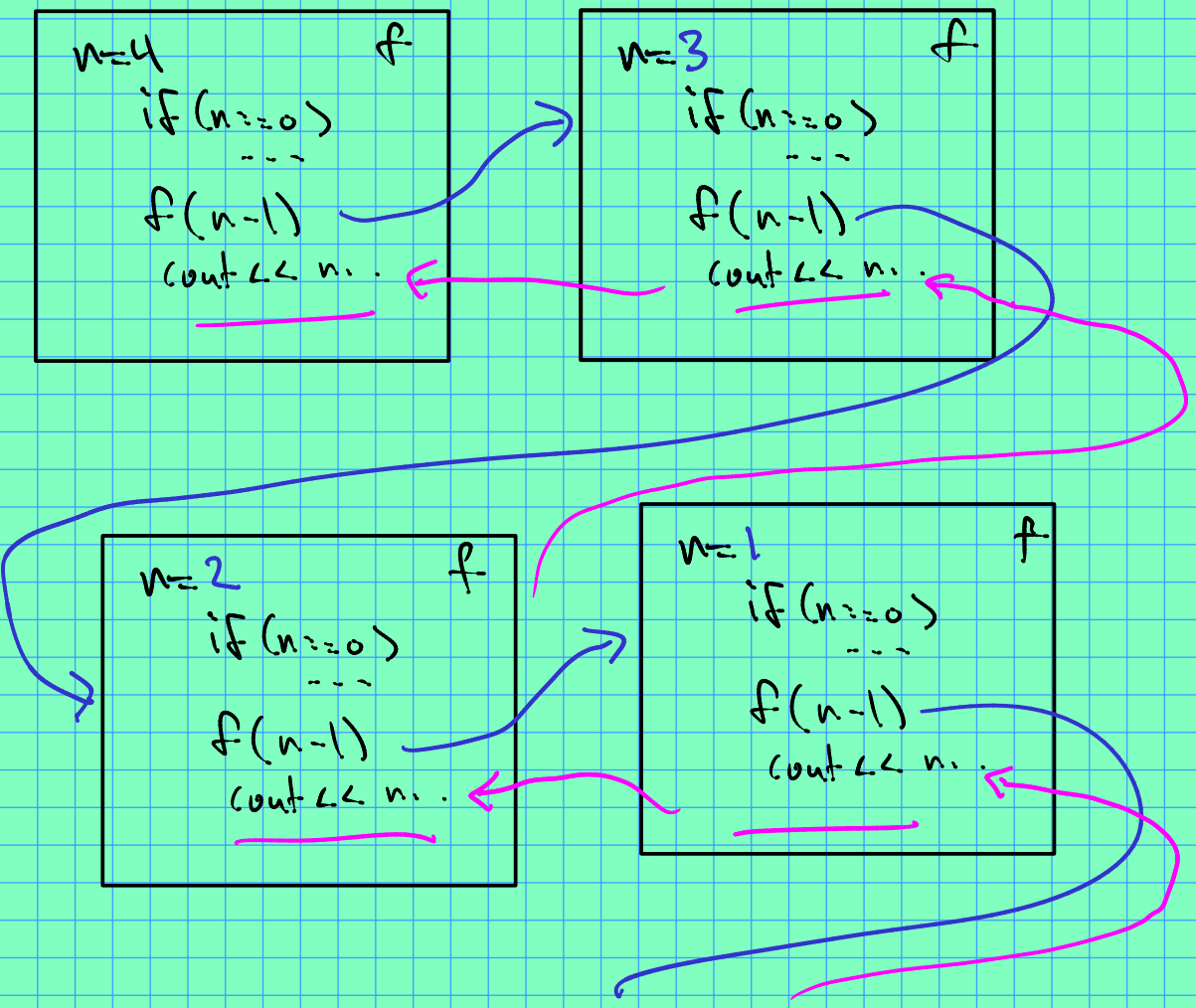
takes a linear amount of stack space

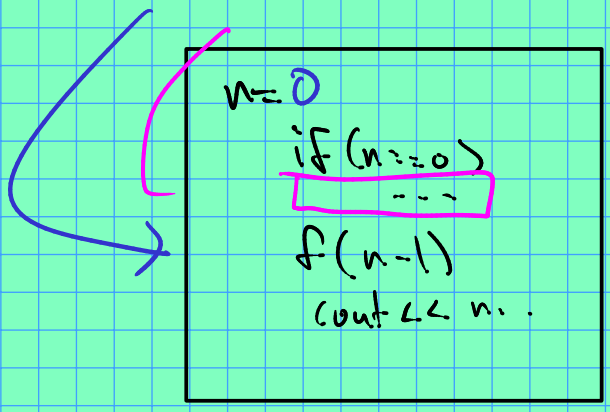
(we have to save all of the other copies of n, as well as the points where you "left off" (return address).)

(Note: for simple recursive functions like this, an optimizing compiler (e.g. g++ -O2) will turn this into a loop for you. Cf. "tail recursion".)

Example: What is the output of f(4), where

```
void f(int n) {
    if(n==0) {
        cout << "0 ";
    }
    f(n-1);
    cout << n << " ";
}
```




n=0
if(n==0)
---
f(n-1)
cout << n;