# CSC 236 Tutorial 7

July 5, 2023

# Today

# A Mystery Algorithm

Towers of Hanoi

# Mystery

What does the following function do? Converts a number to binary representation!

```
def mystery(n: int) -> str:
    if n < 2:
        return str(n)
    else:
        return mystery(n // 2) + mystery(n % 2)
```

Try finding the value of `mystery(7)`, `mystery(12)`...

A Mystery Algorithm

# Towers of Hanoi

# Towers of Hanoi

Go to this website and try the game.

If you can do it with 3 disks, try it with 4!

# Recursive Insight

Let `tower_of_hanoi`$(n, a, b, c)$ be a function that moves the top $n$ disks from peg $a$ to peg $c$ using peg $b$ as an auxiliary disk. Note that this function should move the disks in a way that respects the game's rules. I.e., larger disks cannot be placed on smaller disks.

Here's the recursive insight.

- To move $n$ disks from $a$ to $c$, I move $n-1$ disks from $a$ to $b$, then move the largest disk from $a$ to $c$ and then move the $n-1$ disks from $b$ to $c$.

# Here is that insight summarized in an algorithm

```
def towers_of_hanoi(n, a, b, c):
    if n > 0:
        towers_of_hanoi(n-1, a, c, b)
        c.append(a.pop())
        towers_of_hanoi(n-1, b, a, c)
```

# Let's check that it works!

link to code

# Runtime

What is a recurrence for the runtime of this algorithm?

The recurrence is $T(n) = 2T(n-1) + 1$

# Runtime

Prove $T(n) = \Theta(2^n)$ using the substitution method.

# Runtime

Prove $T(n) = \Theta(2^n)$ using the substitution method.

If you get stuck, try utilizing lower order term!

# Solution

Claim $T(n) \leq c2^n + d$. Use the substitution method.

$$\begin{aligned} T(n) &= 2T(n-1) + 1 \\ &\leq 2(c2^{n-1} + d) + 1 \\ &\leq c2^n + 2d + 1 \end{aligned}$$

This is at most $c2^n + d$ when $d \leq -1$.

Claim $T(n) \geq c2^n$. Use the substitution method.

$$\begin{aligned} T(n) &= 2T(n-1) + 1 \\ &\geq 2(c2^{n-1}) + 1 \\ &\geq c2^n \end{aligned}$$

Thus, $T(n) = \Omega(2^n)$.

Isn't this a little strange?
$T(n) \leq c2^{n-1} - 1$ but $T(n) \geq c2^{n-1}$!

- Subtracting a lower order term helps the inductive step go through in the Big-O proof. **This is counter-intuitive!**
- Remember that the constants are different, so there's no contradiction.

# Prove correctness

$P(n)$. If $a$, $b$, and $c$ contain a valid ordering of pegs and the number of disks on $a$ is at most $n$, `tower_of_hanoi`$(n, a, b, c)$ moves the top $n$ disks from $a$ to $c$ while respecting the rules of the game.

# Proof

**Base case.** For $n = 0$, there are no disks to move, so we are done.

**Inductive step.** Let $k \in \mathbb{N}$ and assume $P(k)$. We'll show $P(k+1)$. Let $a, b, c$ be any valid configuration of pegs. Consider the execution of `tower_of_hanoi`$(k+1, a, b, c)$.

- By the inductive hypothesis, the first call to `tower_of_hanoi` moves the top $k$ disks from $a$ to $b$.
- The next line the moves the $k+1$th disk from $a$ to $c$.
- By the inductive hypothesis, the last line moves the $k$ disks moved to $b$ in the first recursive call to $c$.

What does 'valid' mean? Some (not super interesting) technical details are swept under the rug here, but you should try to work them out if you want!

# If there's more time

Use the time as OH!