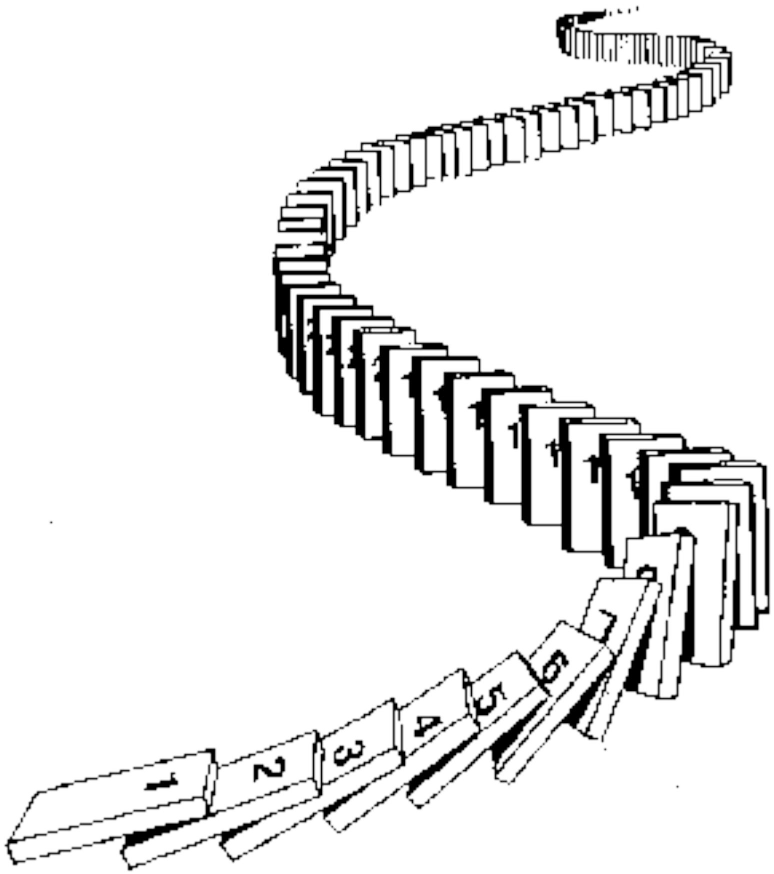
# CSE 311

# Foundations of Computing I

* All slides are a combined effort between previous instructors of the course

# CSE 311: Foundations of Computing

## Lecture 16: Recursively Defined Sets



# Strong Induction

$$P(0)$$
$$\forall k\ \Big(\big(P(0) \wedge P(1) \wedge P(2) \wedge \cdots \wedge P(k)\big) \rightarrow P(k+1)\Big)$$

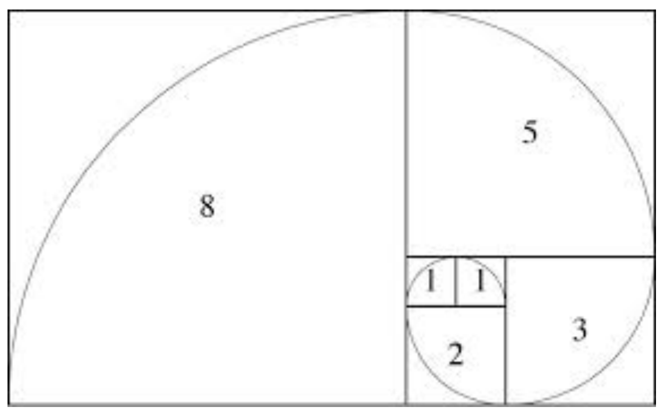
---

$$\therefore\ \forall n\ P(n)$$

1. By induction we will show that $P(n)$ is true for every $n \geq 0$
2. <u>Base Case</u>: Prove $P(0)$
3. <u>Inductive Hypothesis:</u>
   Assume that for some arbitrary integer $k \geq 0$, $P(j)$ is true for every $j$ from $0$ to $k$
4. <u>Inductive Step:</u>
   Prove that $P(k+1)$ is true using the Inductive Hypothesis (that $P(j)$ is true for all values □ $k$)
5. <u>Conclusion:</u> Result follows by induction

# Fibonacci Numbers

$$f_0 = 0$$
$$f_1 = 1$$
$$f_n = f_{n-1} + f_{n-2} \quad \text{for all } n \geq 2$$




# Bounding the Fibonacci Numbers

**Define $f_n$ as:** **$f_0 = 0$**
**$f_1 = 1$**
**$f_n = f_{n-1} + f_{n-2}$ for all $n \geq 2$**

**Theorem:**
**$2^{n/2-1} \leq f_n$ and $f_n < 2^n$ for all $n \geq 2$**

**Proof:**
Let P(n) be "$2^{n/2-1} \leq f_n$ and $f_n < 2^n$" for all $n \geq 2$.
We go by strong induction on n.

**Base Case:** $2^{2/2-1} = 2^0 = 1 \leq 0 + 1 = f_2$, and
$f_2 = 0 + 1 = 1 < 4 = 2^2$. So, P(2) is true.

**Induction Hypothesis:**
Suppose P(j) for all integers j s.t. $2 \leq j \leq k$ for some $k \geq 2$.

**Induction Step:** We want to show **$2^{(k+1)/2-1} \leq f_{k+1}$ and $f_{k+1} < 2^n$**

# Bounding the Fibonacci Numbers

**Define $f_n$ as:** $f_0 = 0$
$f_1 = 1$
$f_n = f_{n-1} + f_{n-2}$ **for all $n \geq 2$**

**Theorem:**
$2^{n/2-1} \leq f_n$ **and** $f_n < 2^n$ **for all $n \geq 2$**

**Induction Step:** We want to show $2^{(k+1)/2-1} \leq f_{k+1}$ **and** $f_{k+1} < 2^n$

**If k+1=3**, $2^{3/2-1} = 2^{1/2} \leq 2 = 1 + 1 = f_3$, and
$f_3 = 1 + 1 = 2 < 8 = 2^3$. So, P(3) is true.

**Otherwise,** note that $f_{k+1} = f_k + f_{k-1}$ by definition.

Taking each inequality separately:

$$
\begin{aligned}
f_{k+1} = f_k + f_{k-1} &< 2^k + 2^{k-1} && \text{(by IH)} \\
&< 2^k + 2^k && (2^{k-1} < 2^k) \\
&= 2^{k+1}
\end{aligned}
$$

$$
\begin{aligned}
f_{k+1} = f_k + f_{k-1} &\geq 2^{k/2-1} + 2^{(k-1)/2-1} && \text{(by IH)} \\
&\geq 2^{(k-1)/2-1} + 2^{(k-1)/2-1} && \text{(Because } 2^{k/2-1} > 2^{(k-1)/2-1}) \\
&= 2(2^{(k-1)/2-1}) && \text{(Combining terms)} \\
&= 2^{2/2+(k-1)/2-1} && \text{(Multiplying)} \\
&= 2^{(k+1)/2-1}
\end{aligned}
$$

**So, the claim is true by strong induction.**

# Running time of Euclid's algorithm

**Theorem:** **Suppose that Euclid's Algorithm takes n steps for gcd(a,b) with a > b. Then, $a \geq f_{n+1}$.**

**We go by strong induction on n.**
**Let P(n) be "gcd(a,b) with a > b takes n steps → $a \geq f_{n+1}$" for all $n \geq 1$.**

**Base Case:**
If Euclid's Algorithm on a, b, with a > b, takes 1 step, then it must be the case that b | a.
Note that $f_2 = 1$.
Note that if a were 0, then gcd(0, b), which takes zero steps. So, the smallest possible value for a is 1, which is $f_2$.
**Induction Hypothesis:** Suppose P(j) for all integers j s.t. $1 \leq j \leq k$ for some $k \geq 1$.

**Induction Step:** We want to show if gcd(a,b) takes k+1 steps, then $a \geq f_{k+2}$.
**If k = 2**, note that a > 1, because gcd(1, b) takes one step. Also, $f_3 = 2$.

# Running time of Euclid's algorithm

**Theorem: Suppose that Euclid's Algorithm takes n steps for gcd(a,b) with a > b. Then, $a \geq f_{n+1}$.**

Since the algorithm took k+1 steps, let's give them names:

Say $r_{k+1} = a$ and $r_k = b$, and $r_i = r_{i-1} \bmod r_{i-2.}$

So, $\gcd(a, b) = \gcd(r_{k+1,}\ r_k)$

$= \gcd(r_k, r_k \bmod r_{k+1}) = \gcd(r_k, r_{k-1})$

$= \gcd(r_{k-1}, r_{k-1} \bmod r_k) = \gcd(r_{k-1}, r_{k-2})$

$= \ldots$

Writing these as equations, we have:

$$
\begin{aligned}
r_{k+1} &= q_k r_k + r_{k-1} \\
r_k &= q_{k-1} r_{k-1} + r_{k-2} \\
&\ \ \ldots \\
r_3 &= q_2 r_2 + r_1 \\
r_2 &= q_1 r_1
\end{aligned}
$$

Note that $q_i \geq 1$, $r_i \geq 1$.

Note that after one iteration of the algorithm, we're left with $\gcd(r_k, r_{k-1})$ which takes k steps. By the IH, $r_k \geq f_{k+1}$. So,

$$
\begin{aligned}
r_{k+1} &= q_k r_k + r_{k-1} && \text{(by gcd algorithm)} \\
&\geq q_k f_{k+1} + f_k && \text{(by IH)} \\
&\geq f_{k+1} + f_k && (q_k \geq 1) \\
&\geq f_{k+2} && \text{(definition of f)}
\end{aligned}
$$

# Recursive Definition of Sets

## Recursive Definition

- Basis Step: $0 \in S$
- Recursive Step: If $x \in S$, then $x + 2 \in S$
- Exclusion Rule: Every element in S follows from basis steps and a finite number of recursive steps.

# Recursive Definitions of Sets

Basis: $6 \in S$, $15 \in S$
Recursive: If $x, y \in S$, then $x+y \in S$

Basis: $[1, 1, 0] \in S$, $[0, 1, 1] \in S$
Recursive: If $[x, y, z] \in S$, then $[\alpha x, \alpha y, \alpha z] \in S$
If $[x_1, y_1, z_1] \in S$ and $[x_2, y_2, z_2] \in S$, then
$[x_1 + x_2, y_1 + y_2, z_1 + z_2] \in S$.

Powers of 3:
Basis: $1 \in S$
Recursive: If $x \in S$, then $3x \in S$.