What's that Doctor McCluckles? Making them ovoid would increase structural integrity and enable a more comfortable delivery? He's right again Professor!
Midterm

• Midterm in class next Wednesday

• Covers material up to ordinary induction (HW5)

• Closed book, closed notes
  – will provide reference sheets

• No calculators
  – arithmetic is intended to be straightforward
  – (only a small point deduction anyway)
Midterm

• 5 problems covering:
  – Propositional Logic
    Including circuits / Boolean algebra / normal forms
  – Predicate Logic/English Translation
  – Modular arithmetic
  – Set theory
  – Induction

• 10 minutes per problem
  – write quickly, don’t get stuck on one problem
  – focus on the overall structure of the solution
How to prove $\forall x \in S, P(x)$ is true:

**Base Case:** Show that $P(u)$ is true for all specific elements $u$ of $S$ mentioned in the *Basis step*.

**Inductive Hypothesis:** Assume that $P$ is true for some arbitrary values of each of the *existing named elements* mentioned in the *Recursive step*.

**Inductive Step:** Prove that $P(w)$ holds for each of the *new elements* $w$ constructed in the *Recursive step* using the named elements mentioned in the Inductive Hypothesis.

**Conclude** that $\forall x \in S, P(x)$.
Last time: Using Structural Induction

• Let $S$ be given by...
  – **Basis:** $6 \in S; \; 15 \in S$
  – **Recursive:** if $x, y \in S$ then $x + y \in S$.

**Claim:** Every element of $S$ is divisible by 3.
**Claim:** Every element of $S$ is divisible by 3.

1. Let $P(x)$ be “$3 \mid x$”. We prove that $P(x)$ is true for all $x \in S$ by structural induction.

**Basis:** $6 \in S$; $15 \in S$

**Recursive:** if $x, y \in S$ then $x + y \in S$
**Claim:** Every element of $S$ is divisible by 3.

1. Let $P(x)$ be “$3 \mid x$”. We prove that $P(x)$ is true for all $x \in S$ by structural induction.

2. **Base Case:** $3 \mid 6$ and $3 \mid 15$ so $P(6)$ and $P(15)$ are true.

3. **Inductive Hypothesis:** Suppose that $P(x)$ and $P(y)$ are true for some arbitrary $x, y \in S$.

4. **Inductive Step:** Goal: Show $P(x+y)$.

   Since $P(x)$ is true, $3 \mid x$ and so $x = 3m$ for some integer $m$.

   Since $P(y)$ is true, $3 \mid y$ and so $y = 3n$ for some integer $n$.

   Therefore, $x + y = 3m + 3n = 3(m + n)$ and thus $3 \mid (x + y)$.

   Hence, $P(x + y)$ is true.

5. Therefore, by induction, $3 \mid x$ for all $x \in S$.

---

**Basis:** $6 \in S; 15 \in S$

**Recursive:** if $x, y \in S$ then $x + y \in S$
Claim: Every element of $S$ is divisible by 3.

1. Let $P(x)$ be “$3|x$”. We prove that $P(x)$ is true for all $x \in S$ by structural induction.

2. Base Case: $3|6$ and $3|15$ so $P(6)$ and $P(15)$ are true.

3. Inductive Hypothesis: Suppose that $P(x)$ and $P(y)$ are true for some arbitrary $x, y \in S$.

4. Inductive Step: **Goal:** Show $P(x+y)$

   Since $P(x)$ is true, $3|x$ and so $x=3m$ for some integer $m$ and since $P(y)$ is true, $3|y$ and so $y=3n$ for some integer $n$.

   Therefore $x+y=3m+3n=3(m+n)$ and thus $3|(x+y)$.

   Hence $P(x+y)$ is true.

5. Therefore by induction $3|x$ for all $x \in S$.

Basis: $6 \in S$; $15 \in S$

Recursive: if $x, y \in S$ then $x + y \in S$
Claim: Every element of $S$ is divisible by 3.

1. Let $P(x)$ be “$3 | x$”. We prove that $P(x)$ is true for all $x \in S$ by structural induction.

2. Base Case: $3 | 6$ and $3 | 15$ so $P(6)$ and $P(15)$ are true.

3. Inductive Hypothesis: Suppose that $P(x)$ and $P(y)$ are true for some arbitrary $x, y \in S$.

4. Inductive Step: Goal: Show $P(x+y)$.

   Since $P(x)$ is true, $3 | x$ and so $x = 3m$ for some integer $m$ and
   since $P(y)$ is true, $3 | y$ and so $y = 3n$ for some integer $n$.
   Therefore $x+y = 3m+3n = 3(m+n)$ and thus $3 | (x+y)$.
   Hence $P(x+y)$ is true.

5. Therefore by induction $3 | x$ for all $x \in S$.

Basis: $6 \in S; \ 15 \in S$

Recursive: if $x, y \in S$ then $x + y \in S$
More Structural Induction

• Let $R$ be given by...
  – **Basis:** $12 \in R$; $15 \in R$
  – **Recursive:** if $x \in R$, then $x + 6 \in R$ and $x + 15 \in R$

• Two base cases and two recursive cases, one existing element.

**Claim:** $R \subseteq S$; i.e. every element of $R$ is also in $S$.

Proof needs structural induction using definition of $R$ since statement is of the form $\forall x \in R. P(x)$
Claim: Every element of \( R \) is in \( S \). \( (R \subseteq S) \)

1. Let \( P(x) \) be “\( x \in S \)”. We prove that \( P(x) \) is true for all \( x \in R \) by structural induction.

2. Base Case: (12): \( 6 \in S \) so \( 6+6=12 \in S \) by definition of \( S \), so \( P(12) \)
   (15): \( 15 \in S \), so \( P(15) \) is also true

3. Ind. Hyp: Suppose that \( P(x) \) is true for some arbitrary \( x \in R \)

4. Inductive Step: \( \text{Goal: Show } P(x+6) \text{ and } P(x+15) \)
   Since \( P(x) \) holds, we have \( x \in S \). Since \( 6 \in S \) from the recursive step of \( S \), we get \( x + 6 \in S \), so \( P(x+6) \) is true, and since \( 15 \in S \) we get \( x + 15 \in S \), so \( P(x+15) \) is true.

5. Therefore \( P(x) \) (i.e., \( x \in S \)) for all \( x \in R \) by induction.

Basis: \( 6 \in S; \ 15 \in S \)

Recursive: if \( x, y \in S \), then \( x + y \in S \)

Basis: \( 12 \in R; \ 15 \in R \)

Recursive: if \( x \in R \), then \( x + 6 \in R \) and \( x + 15 \in R \)
Recursive Definitions

• Recursively defined functions and sets are our mathematical models of code and the data it uses
  – recursively defined sets can be translated into Java classes
  – recursively defined functions can be translated into Java functions
    some (but not all) can be written more cleanly as loops

• Can now do proofs about CS-specific objects
Lists of Integers

• **Basis:** nil ∈ List

• **Recursive step:**

  if $L \in \text{List}$ and $a \in \mathbb{Z}$,

  then $a :: L \in \text{List}$

Examples:

- nil $\rightarrow$ []
- 1 :: nil $\rightarrow$ [1]
- 2 :: 1 :: nil $\rightarrow$ [2, 1]
- 3 :: 2 :: 1 :: nil $\rightarrow$ [3, 2, 1]
Assume that the recursive definition of $S$ gives a unique way to construct every element of $S$.

We can define the values of a function $f$ on $S$ recursively as follows:

**Basis**: Define $f(u)$ for all specific elements $u$ of $S$ mentioned in the *Basis step*.

**Recursive Step**: Define $f(w)$ for each of the new elements $w$ constructed in terms of $f$ applied to each of the existing named elements mentioned in the *Recursive step*.
Functions on Lists

Length:

\[
\begin{align*}
\text{len}(\text{nil}) & := 0 \\
\text{len}(a :: L) & := \text{len}(L) + 1 \\
\end{align*}
\]

for any \( L \in \text{List} \) and \( a \in \mathbb{Z} \)

Concatenation:

\[
\begin{align*}
\text{concat}(\text{nil}, R) & := R \\
\text{concat}(a :: L, R) & := a :: \text{concat}(L, R) \\
\end{align*}
\]

for any \( R \in \text{List} \) and

for any \( L, R \in \text{List} \) and \( a \in \mathbb{Z} \)

Basis: \( \text{nil} \in \text{List} \)

Recursive step:

if \( L \in \text{List} \) and \( a \in \mathbb{Z} \),

then \( a :: L \in \text{List} \)
Structural Induction

How to prove $\forall x \in S, P(x)$ is true:

Base Case: Show that $P(u)$ is true for all specific elements $u$ of $S$ mentioned in the Basis step

Inductive Hypothesis: Assume that $P$ is true for some arbitrary values of each of the existing named elements mentioned in the Recursive step

Inductive Step: Prove that $P(w)$ holds for each of the new elements $w$ constructed in the Recursive step using the named elements mentioned in the Inductive Hypothesis

Conclude that $\forall x \in S, P(x)$

Basis: $\text{nil} \in \text{List}$
Recursive step: if $L \in \text{List}$ and $a \in \mathbb{Z}$, then $a :: L \in \text{List}$
**Claim:** \( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( L, R \in \text{List} \)

**Length:**
- \( \text{len}(\text{nil}) := 0 \)
- \( \text{len}(a :: L) := \text{len}(L) + 1 \)

**Concatenation:**
- \( \text{concat}(\text{nil}, R) := R \)
- \( \text{concat}(a :: L, R) := a :: \text{concat}(L, R) \)
**Claim:** \( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( L, R \in \text{List} \)

Let \( P(L) \) be “\( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( R \in \text{List} \)”.

We prove \( P(L) \) for all \( L \in \text{List} \) by structural induction.

---

**Length:**

\[
\begin{align*}
\text{len}(\text{nil}) & := 0 \\
\text{len}(a :: L) & := \text{len}(L) + 1
\end{align*}
\]

**Concatenation:**

\[
\begin{align*}
\text{concat}(\text{nil}, R) & := R \\
\text{concat}(a :: L, R) & := a :: \text{concat}(L, R)
\end{align*}
\]
**Claim:** \( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( L, R \in \text{List} \)

Let \( P(L) \) be “\( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( R \in \text{List} \)”.

We prove \( P(L) \) for all \( L \in \text{List} \) by structural induction.

**Base Case** (\( \text{nil} \)): Let \( R \in \text{List} \) be arbitrary. Then,

\[
\text{Length:}
\begin{align*}
\text{len}(\text{nil}) & := 0 \\
\text{len}(a :: L) & := \text{len}(L) + 1
\end{align*}
\]

\[
\text{Concatenation:}
\begin{align*}
\text{concat}(\text{nil}, R) & := R \\
\text{concat}(a :: L, R) & := a :: \text{concat}(L, R)
\end{align*}
\]
**Claim:** \( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \quad \text{for all } L, R \in \text{List} \)

Let \( P(L) \) be “\( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( R \in \text{List} \)”. We prove \( P(L) \) for all \( L \in \text{List} \) by structural induction.

**Base Case** \( (\text{nil}) \): Let \( R \in \text{List} \) be arbitrary. Then,

\[
\begin{align*}
\text{len}(\text{concat}(\text{nil}, R)) &= \text{len}(R) & \text{def of concat} \\
&= 0 + \text{len}(R) \\
&= \text{len}(\text{nil}) + \text{len}(R) & \text{def of len}
\end{align*}
\]

Since \( R \) was arbitrary, \( P(\text{nil}) \) holds.
Claim: \( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( L, R \in \text{List} \)

Let \( P(L) \) be “\( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( R \in \text{List} \)”.

We prove \( P(L) \) for all \( L \in \text{List} \) by structural induction.

Base Case (\( \text{nil} \)): Let \( R \in \text{List} \) be arbitrary. Then, \( \text{len}(\text{concat}(\text{nil}, R)) = \text{len}(R) = 0 + \text{len}(R) = \text{len}(\text{nil}) + \text{len}(R) \), showing \( P(\text{nil}) \).

Inductive Hypothesis: Assume that \( P(L) \) is true for some arbitrary \( L \in \text{List} \), i.e., \( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( R \in \text{List} \).
**Claim:** \( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( L, R \in \text{List} \)

Let \( P(L) \) be “\( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( R \in \text{List} \)”.

We prove \( P(L) \) for all \( L \in \text{List} \) by structural induction.

**Base Case** (\( \text{nil} \)): Let \( R \in \text{List} \) be arbitrary. Then, \( \text{len}(\text{concat}(\text{nil}, R)) = \text{len}(R) = 0 + \text{len}(R) = \text{len}(\text{nil}) + \text{len}(R) \), showing \( P(\text{nil}) \).

**Inductive Hypothesis:** Assume that \( P(L) \) is true for some arbitrary \( L \in \text{List} \), i.e., \( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( R \in \text{List} \).

**Inductive Step:** \( \textbf{Goal: Show that } P(a :: L) \text{ is true } \)
**Claim:** \( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \quad \text{for all } L, R \in \text{List} \)

Let \( P(L) \) be “\( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( R \in \text{List} \)”.

We prove \( P(L) \) for all \( L \in \text{List} \) by structural induction.

**Base Case** (\( \text{nil} \)): Let \( R \in \text{List} \) be arbitrary. Then, \( \text{len}(\text{concat}(\text{nil}, R)) = \text{len}(R) = 0 + \text{len}(R) = \text{len}(\text{nil}) + \text{len}(R) \), showing \( P(\text{nil}) \).

**Inductive Hypothesis:** Assume that \( P(L) \) is true for some arbitrary \( L \in \text{List} \), i.e., \( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( R \in \text{List} \).

**Inductive Step:** Goal: Show that \( P(a :: L) \) is true

Let \( R \in \text{List} \) be arbitrary. Then,

**Length:**
\[
\begin{align*}
\text{len}(\text{nil}) & := 0 \\
\text{len}(a :: L) & := \text{len}(L) + 1
\end{align*}
\]

**Concatenation:**
\[
\begin{align*}
\text{concat}(\text{nil}, R) & := R \\
\text{concat}(a :: L, R) & := a :: \text{concat}(L, R)
\end{align*}
\]
**Claim:** \( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( L, R \in \text{List} \)

Let \( P(L) \) be “\( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( R \in \text{List} \)”.

We prove \( P(L) \) for all \( L \in \text{List} \) by structural induction.

**Base Case** (\( \text{nil} \)): Let \( R \in \text{List} \) be arbitrary. Then, \( \text{len}(\text{concat}(\text{nil}, R)) = \text{len}(R) = 0 + \text{len}(R) = \text{len}(\text{nil}) + \text{len}(R) \), showing \( P(\text{nil}) \).

**Inductive Hypothesis:** Assume that \( P(L) \) is true for some arbitrary \( L \in \text{List} \), i.e., \( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( R \in \text{List} \).

**Inductive Step:** Goal: Show that \( P(\text{a :: L}) \) is true

Let \( R \in \text{List} \) be arbitrary. Then, we can calculate

\[
\begin{align*}
\text{len}(\text{concat}(\text{a :: L}, R)) &= \text{len}(\text{a :: concat}(L, R)) & \text{def of concat} \\
&= 1 + \text{len}(\text{concat}(L, R)) & \text{def of len} \\
&= 1 + \text{len}(L) + \text{len}(R) & \text{IH} \\
&= \text{len}(\text{a :: L}) + \text{len}(R) & \text{def of len}
\end{align*}
\]
**Claim:** \( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( L, R \in \text{List} \)

Let \( P(L) \) be “\( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( R \in \text{List} \)”. We prove \( P(L) \) for all \( L \in \text{List} \) by structural induction.

**Base Case** (\( \text{nil} \)): Let \( R \in \text{List} \) be arbitrary. Then, \( \text{len}(\text{concat}(\text{nil}, R)) = \text{len}(R) = 0 + \text{len}(R) = \text{len}(\text{nil}) + \text{len}(R) \), showing \( P(\text{nil}) \).

**Inductive Hypothesis:** Assume that \( P(L) \) is true for some arbitrary \( L \in \text{List} \), i.e., \( \text{len}(\text{concat}(L, R)) = \text{len}(L) + \text{len}(R) \) for all \( R \in \text{List} \).

**Inductive Step:** Goal: Show that \( P(a :: L) \) is true

Let \( R \in \text{List} \) be arbitrary. Then, we can calculate

\[
\begin{align*}
\text{len}(\text{concat}(a :: L, R)) &= \text{len}(a :: \text{concat}(L, R)) & \text{def of concat} \\
&= 1 + \text{len}(\text{concat}(L, R)) & \text{def of len} \\
&= 1 + \text{len}(L) + \text{len}(R) & \text{IH} \\
&= \text{len}(a :: L) + \text{len}(R) & \text{def of len}
\end{align*}
\]

Since \( R \) was arbitrary, we have shown \( P(a :: L) \).

By induction, we have shown the claim holds for all \( L \in \text{List} \).
Rooted Binary Trees

• **Basis:** is a rooted binary tree

• **Recursive step:**

If $T_1$ and $T_2$ are rooted binary trees,

then also is a rooted binary tree.
Defining Functions on Rooted Binary Trees

- $\text{size} (\bullet) := 1$

- $\text{size} \left( \begin{array}{c} \text{T}_1 \\ \text{T}_2 \end{array} \right) := 1 + \text{size}(\text{T}_1) + \text{size}(\text{T}_2)$

- $\text{height} (\bullet) := 0$

- $\text{height} \left( \begin{array}{c} \text{T}_1 \\ \text{T}_2 \end{array} \right) := 1 + \max\{\text{height}(\text{T}_1), \text{height}(\text{T}_2)\}$
Claim: For every rooted binary tree $T$, $\text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1$
Claim: For every rooted binary tree $T$, $\text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1$

1. Let $P(T)$ be “$\text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1$”. We prove $P(T)$ for all rooted binary trees $T$ by structural induction.

\[
\begin{align*}
\text{size}(\bullet) & := 1 \\
\text{size}\left( \begin{array}{c} 
T_1 \vline T_2 
\end{array} \right) & := 1 + \text{size}(T_1) + \text{size}(T_2) \\
\text{height}(\bullet) & := 0 \\
\text{height}\left( \begin{array}{c} 
T_1 \vline T_2 
\end{array} \right) & := 1 + \max\{\text{height}(T_1), \text{height}(T_2)\}
\end{align*}
\]
**Claim:** For every rooted binary tree $T$, $\text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1$

1. Let $P(T)$ be “$\text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1$”. We prove $P(T)$ for all rooted binary trees $T$ by structural induction.
2. Base Case: $\text{size}(\bullet) = 1$, $\text{height}(\bullet) = 0$, and $2^{0+1-1} = 2^{1-1} = 1$ so $P(\bullet)$ is true.
Claim: For every rooted binary tree $T$, $\text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1$

1. Let $P(T)$ be “$\text{size}(T) \leq 2^{\text{height}(T)+1} - 1$”. We prove $P(T)$ for all rooted binary trees $T$ by structural induction.

2. Base Case: $\text{size}(\bullet) = 1$, $\text{height}(\bullet) = 0$, and $2^{0+1} - 1 = 2^1 - 1 = 1$ so $P(\bullet)$ is true.

3. Inductive Hypothesis: Suppose that $P(T_1)$ and $P(T_2)$ are true for some rooted binary trees $T_1$ and $T_2$, i.e., $\text{size}(T_k) \leq 2^{\text{height}(T_k)} + 1 - 1$ for $k=1,2$


5. So, the $P(T)$ is true for all rooted binary trees by structural induction.
**Claim:** For every rooted binary tree $T$, $\text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1$

1. Let $P(T)$ be “$\text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1$”. We prove $P(T)$ for all rooted binary trees $T$ by structural induction.

2. Base Case: $\text{size}(\bullet) = 1$, $\text{height}(\bullet) = 0$, and $2^{0+1} - 1 = 2^1 - 1 = 1$ so $P(\bullet)$ is true.

3. Inductive Hypothesis: Suppose that $P(T_1)$ and $P(T_2)$ are true for some rooted binary trees $T_1$ and $T_2$, i.e., $\text{size}(T_k) \leq 2^{\text{height}(T_k)} + 1 - 1$ for $k=1,2$

4. Inductive Step: Goal: Prove $P(T_{1,2})$.

$$\text{size}(T_{1,2}) = \text{size}(T_1) + \text{size}(T_2)$$

- $\text{size}(\bullet) ::= 1$
- $\text{size}(T_{1,2}) ::= 1 + \text{size}(T_1) + \text{size}(T_2)$
- $\text{height}(\bullet) ::= 0$
- $\text{height}(T_{1,2}) ::= 1 + \max\{\text{height}(T_1), \text{height}(T_2)\}$

$$\leq 2^{\text{height}(T_{1,2})} + 1 - 1$$
**Claim:** For every rooted binary tree \( T \), \( \text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1 \)

1. Let \( P(T) \) be “\( \text{size}(T) \leq 2^{\text{height}(T)} + 1 - 1 \)”. We prove \( P(T) \) for all rooted binary trees \( T \) by structural induction.

2. **Base Case:** \( \text{size}(\bullet) = 1 \), \( \text{height}(\bullet) = 0 \), and \( 2^{0+1}-1=2^1-1=1 \) so \( P(\bullet) \) is true.

3. **Inductive Hypothesis:** Suppose that \( P(T_1) \) and \( P(T_2) \) are true for some rooted binary trees \( T_1 \) and \( T_2 \), i.e., \( \text{size}(T_k) \leq 2^{\text{height}(T_k)} + 1 - 1 \) for \( k=1,2 \)

4. **Inductive Step:**

   **Goal:** Prove \( P(\text{tree}) \).

   By def, \( \text{size}(\text{tree}) = 1 + \text{size}(T_1) + \text{size}(T_2) \)

   \[
   \leq 1 + 2^{\text{height}(T_1)} + 1 - 1 + 2^{\text{height}(T_2)} + 1 - 1 \leq 2\cdot\max(2^{\text{height}(T_1)}, 2^{\text{height}(T_2)}) - 1
   \]

   by IH for \( T_1 \) and \( T_2 \)

   \[
   = 2^{\text{height}(T_1)} + 2^{\text{height}(T_2)} + 1 - 1
   \]

   \[
   \leq 2\cdot\max(2^{\text{height}(T_1)} + 1, 2^{\text{height}(T_2)} + 1) - 1
   \]

   \[
   \leq 2(2^{\max(\text{height}(T_1), \text{height}(T_2))} + 1) - 1
   \]

   \[
   \leq 2(2^{\text{height}(\text{tree})} - 1) \leq 2^{\text{height}(\text{tree})} + 1 - 1
   \]

   which is what we wanted to show.

5. So, the \( P(T) \) is true for all rooted binary trees by structural induction.
Strings

• An alphabet $\Sigma$ is any finite set of characters

• The set $\Sigma^*$ of strings over the alphabet $\Sigma$
  – example: $\{0,1\}^*$ is the set of binary strings
    $0, 1, 00, 01, 10, 11, 000, 001, ...$ and “”

• $\Sigma^*$ is defined recursively by
  – Basis: $\varepsilon \in \Sigma^*$ ($\varepsilon$ is the empty string, i.e., “”)
  – Recursive: if $w \in \Sigma^*$, $a \in \Sigma$, then $wa \in \Sigma^*$
Palindromes

Palindromes are strings that are the same when read backwards and forwards

**Basis:**

\[ \varepsilon \text{ is a palindrome} \]

any \( a \in \Sigma \) is a palindrome

**Recursive step:**

If \( p \) is a palindrome,
then \( apa \) is a palindrome for every \( a \in \Sigma \)
Functions on Recursively Defined Sets (on $\Sigma^*$)

Length:
\[
\text{len}(\varepsilon) := 0
\]
\[
\text{len}(wa) := \text{len}(w) + 1 \text{ for } w \in \Sigma^*, a \in \Sigma
\]

Concatenation:
\[
x \cdot \varepsilon := x \text{ for } x \in \Sigma^*
\]
\[
x \cdot wa := (x \cdot w)a \text{ for } x \in \Sigma^*, a \in \Sigma
\]

Reversal:
\[
\varepsilon^R := \varepsilon
\]
\[
(wa)^R := a \cdot w^R \text{ for } w \in \Sigma^*, a \in \Sigma
\]

Number of $c$’s in a string:
\[
\#_c(\varepsilon) := 0
\]
\[
\#_c(wc) := \#_c(w) + 1 \text{ for } w \in \Sigma^*
\]
\[
\#_c(wa) := \#_c(w) \text{ for } w \in \Sigma^*, a \in \Sigma, a \neq c
\]
Last time: Structural Induction

How to prove $\forall x \in S, P(x)$ is true:

**Base Case:** Show that $P(u)$ is true for all specific elements $u$ of $S$ mentioned in the Basis step.

**Inductive Hypothesis:** Assume that $P$ is true for some arbitrary values of each of the existing named elements mentioned in the Recursive step.

**Inductive Step:** Prove that $P(w)$ holds for each of the new elements $w$ constructed in the Recursive step using the named elements mentioned in the Inductive Hypothesis.

**Conclude** that $\forall x \in S, P(x)$.

**Basis:** $\varepsilon \in \Sigma^*$

**Recursive Steps:**
if $w \in \Sigma^*$ and $a \in \Sigma$,
then $wa \in \Sigma^*$
Claim: \( \text{len}(x \cdot y) = \text{len}(x) + \text{len}(y) \) for all \( x, y \in \Sigma^* \)

Let \( P(y) \) be “\( \text{len}(x \cdot y) = \text{len}(x) + \text{len}(y) \) for all \( x \in \Sigma^* \)”.

We prove \( P(y) \) for all \( y \in \Sigma^* \) by structural induction.

Base Case (\( y = \epsilon \)): Let \( x \in \Sigma^* \) be arbitrary. Then, \( \text{len}(x \cdot \epsilon) = \text{len}(x) = \text{len}(x) + \text{len}(\epsilon) \) since \( \text{len}(\epsilon) = 0 \). Since \( x \) was arbitrary, \( P(\epsilon) \) holds.

Inductive Hypothesis: Assume that \( P(w) \) is true for some arbitrary \( w \in \Sigma^* \), i.e., \( \text{len}(x \cdot w) = \text{len}(x) + \text{len}(w) \) for all \( x \)
Claim: \( \text{len}(x\cdot y) = \text{len}(x) + \text{len}(y) \) for all \( x,y \in \Sigma^* \)

Let \( P(y) \) be “\( \text{len}(x\cdot y) = \text{len}(x) + \text{len}(y) \) for all \( x \in \Sigma^* \)”.

We prove \( P(y) \) for all \( y \in \Sigma^* \) by structural induction.

Base Case (\( y = \varepsilon \)): Let \( x \in \Sigma^* \) be arbitrary. Then, \( \text{len}(x \cdot \varepsilon) = \text{len}(x) = \text{len}(x) + \text{len}(\varepsilon) \) since \( \text{len}(\varepsilon) = 0 \). Since \( x \) was arbitrary, \( P(\varepsilon) \) holds.

Inductive Hypothesis: Assume that \( P(w) \) is true for some arbitrary \( w \in \Sigma^* \), i.e., \( \text{len}(x \cdot w) = \text{len}(x) + \text{len}(w) \) for all \( x \in \Sigma^* \).

Inductive Step: Goal: Show that \( P(wa) \) is true for every \( a \in \Sigma \).

Let \( a \in \Sigma \) and \( x \in \Sigma^* \). Then \( \text{len}(x \cdot wa) = \text{len}((x \cdot w)a) \) by def of \( \cdot \)

\[
= \text{len}(x \cdot w) + 1 \quad \text{by def of len}
= \text{len}(x) + \text{len}(w) + 1 \quad \text{by I.H.}
= \text{len}(x) + \text{len}(wa) \quad \text{by def of len}
\]

Therefore, \( \text{len}(x \cdot wa) = \text{len}(x) + \text{len}(wa) \) for all \( x \in \Sigma^* \), so \( P(wa) \) is true.

So, by induction \( \text{len}(x \cdot y) = \text{len}(x) + \text{len}(y) \) for all \( x,y \in \Sigma^* \)