1. Consider the set of strings $L \subseteq \{0, 1\}^*$ defined recursively as follows:

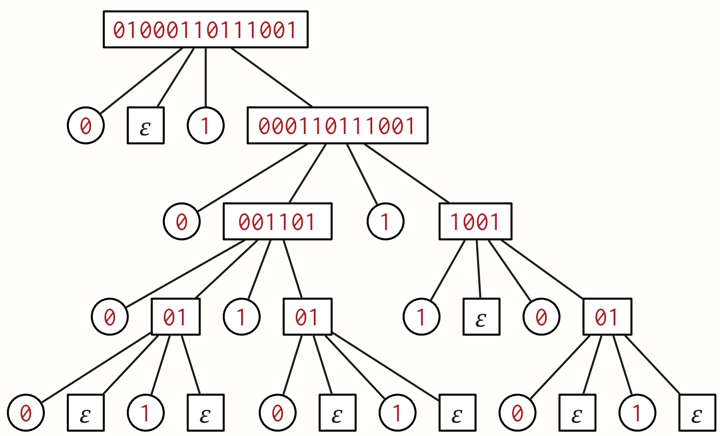
- The empty string ε is in L.
- For any two strings x and y in L, the string $0x1y$ is also in L.
- For any two strings x and y in L, the string $1x0y$ is also in L.
- These are the only strings in L.

(a) Prove that the string 01000110111001 is in L.

Solution:

- $\varepsilon \in L$ by definition.
- $01 = 0 \bullet \varepsilon \bullet 1 \bullet \varepsilon \in L$ because $\varepsilon \in L$ and $\varepsilon \in L$.
- $1001 = 1 \bullet \varepsilon \bullet 0 \bullet 01$ because $\varepsilon \in L$ and $01 \in L$.
- $001101 = 0 \bullet 01 \bullet 1 \bullet 01 \in L$ because $01 \in L$ and $01 \in L$.
- $10001101 = 1 \bullet \varepsilon \bullet 0 \bullet 001101 \in L$ because $\varepsilon \in L$ and $001101 \in L$.
- $01000110111001 = 0 \bullet 10001101 \bullet 1 \bullet 1001 \in L$ because $1001 \in L$ and $10001101 \in L$. ■

Solution: The following parse tree illustrates the proof. Every string in a rectangle is in L, either because that string is empty, or because it can be broken into four smaller strings (the four children) as described in the definition of L. Circles represent individual symbols.



■

Solution (clever): By part (c), it suffices to observe that $\#(0, 01000110111001) = \#(1, 01000110111001) = 7$, as stated in the homework handout. ■

Rubric: 2 points. These are neither the only correct derivations nor the correct proof structures for these derivations. The first proof is more detailed than necessary for full credit, but any proof must separately justify each component substring (for example, 10001101, 001101, 1001, and 01 in the first proof). The clever solution is worth full credit even without a solution to part (c).

(b) Prove that $\#(0, w) = \#(1, w)$ for every string $w \in L$.

Solution: Let w be an arbitrary string in L.
Assume, for any string $x \in L$ where $|x| < |w|$, that $\#(0, x) + \#(1, x)$.
There are three cases to consider.

- Suppose $w = \varepsilon$. Then $\#(0, w) = 0 = \#(1, \varepsilon)$ by definition of $\#$.

$$\begin{aligned}
\#(0, w) &= \#(0, \varepsilon) \\
&= 0 && \text{by definition of } \# \\
&= \#(1, \varepsilon) && \text{by definition of } \# \\
&= \#(1, w)
\end{aligned}$$

- Suppose $w = 0x1y$ for some strings $x, y \in L$.

$$\begin{aligned}
\#(0, w) &= \#(0, 0x1y) \\
&= 1 + \#(0, x1y) && \text{by definition of } \# \\
&= 1 + \#(0, x) + \#(0, 1y) && \#(a, xy) = \#(a, x) + \#(a, y) \\
&= 1 + \#(0, x) + \#(0, y) && \text{by definition of } \# \\
&= 1 + \#(1, x) + \#(1, y) && \text{by the inductive hypothesis} \\
&= \#(1, x) + \#(1, 1y) && \text{by definition of } \# \\
&= \#(1, x1y) && \#(a, xy) = \#(a, x) + \#(a, y) \\
&= \#(1, 0x1y) && \text{by definition of } \# \\
&= \#(1, w)
\end{aligned}$$

- Suppose $w = 1x0y$ for some strings $x, y \in L$.

$$\begin{aligned}
\#(0, w) &= \#(0, 1x0y) \\
&= \#(0, x0y) && \text{by definition of } \# \\
&= \#(0, x) + \#(0, 0y) && \#(a, xy) = \#(a, x) + \#(a, y) \\
&= \#(0, x) + 1 + \#(0, y) && \text{by definition of } \# \\
&= \#(1, x) + 1 + \#(1, y) && \text{by the inductive hypothesis} \\
&= \#(1, x) + 1 + \#(1, 0y) && \text{by definition of } \# \\
&= 1 + \#(1, x0y) && \#(a, xy) = \#(a, x) + \#(a, y) \\
&= \#(1, 1x0y) && \text{by definition of } \# \\
&= \#(1, w)
\end{aligned}$$

In all three cases, we conclude that $\#(0, w) = \#(1, w)$. ■

Rubric: 4 points: standard induction rubric (scaled) *(c) Prove that L contains every string $w \in \{0,1\}^*$ such that $\#(0,w) = \#(1,w)$.

Solution: To simplify notation, let $\Delta(w) = \#(1,w) - \#(0,w)$ for any string w. Because $\#(a, x \bullet y) = \#(a,x) + \#(a,y)$, we have $\Delta(x \bullet y) = \Delta(x) + \Delta(y)$ for all strings x and y. In particular, we have $\Delta(x \bullet 0) = \Delta(x) - 1$ and $\Delta(x \bullet 1) = \Delta(x) + 1$.

We prove by induction that L contains every string w such that $\Delta(w) = 0$.

Proof: Let w be an arbitrary string such that $\Delta(w) = 0$.
Assume L contains every string x such that $|x| < |w|$ and $\Delta(x) = 0$.
There are three cases to consider.

- If $w = \varepsilon$, then $w \in L$ by definition.
- Suppose $w = 0y$ for some string y.

  Write $w = p \bullet z$, where p is the *shortest* non-empty prefix of w such that $\Delta(p) \geq 0$. We know that such a prefix exists, because w is a non-empty prefix of w with $\Delta(w) \geq 0$. (The suffix z might be empty.)

  Because p is non-empty, we can write $p = qa$ for some string q and some symbol $a \in \{0,1\}$. The definition of p implies that $\Delta(q) \leq -1$.

  If $a = 0$, then $\Delta(p) = \Delta(q0) = \Delta(q) - 1 < 0$, which is impossible because $\Delta(p) \geq 0$. So we must have $a = 1$ and therefore $\Delta(p) = \Delta(q) + 1$. It follows that $\Delta(p) = 0$ and $\Delta(q) = -1$.

  Because p starts with 0 and ends with 1, we must have $p = 0x1$ for some string x (which might be empty). It follows that $\Delta(x) = \Delta(p) = 0$, and therefore $x \in L$ by the inductive hypothesis.

  We also have $\Delta(w) = \Delta(p) + \Delta(z) = \Delta(z)$, and therefore $\Delta(z) = 0$. So the inductive hypothesis implies $z \in L$.

  We conclude that $w = 0x1z$, where $x \in L$ and $z \in L$, and thus $w \in L$.
- A symmetric argument implies that if $w = 1y$ for some string y, then $w \in L$.

In all three cases, we conclude that $w \in L$. ■

Rubric: 4 points = 1 for induction boilerplate (including strong induction hypothesis and exhaustive case analysis) + ½ for base case + 2 for first inductive case + ½ for second inductive case. Yes, the one-line argument is enough for the last ½ point.

2. Consider the following pair of mutually recursive functions on strings:

$$odds(w) := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ a \cdot evens(x) & \text{if } w = ax \end{cases} \qquad evens(w) := \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ odds(x) & \text{if } w = ax \end{cases}$$

(a) Give a self-contained recursive definition for the function *evens* that does not involve the function *odds*.

Solution:

$$evens(w) = \begin{cases} \varepsilon & \text{if } w = \varepsilon \\ \varepsilon & \text{if } w = a \text{ for some symbol } a \in \Sigma \\ b \cdot evens(x) & \text{if } w = abx \text{ for some } a, b \in \Sigma \text{ and some } x \in \Sigma^* \end{cases}$$

■

Rubric: 2 points = ½ for case breakdown + ½ for base cases + 1 for final recursive case.

(b) Prove the following identity for all strings w and x:

$$evens(w \bullet x) = \begin{cases} evens(w) \bullet evens(x) & \text{if } |w| \text{ is even,} \\ evens(w) \bullet odds(x) & \text{if } |w| \text{ is odd.} \end{cases}$$

Solution: Let w and x be arbitrary strings.
Assume for all strings y shorter than w that

$$evens(y \bullet x) = \begin{cases} evens(y) \bullet evens(x) & \text{if } |y| \text{ is even,} \\ evens(y) \bullet odds(x) & \text{if } |y| \text{ is odd.} \end{cases}$$

There are **three** cases to consider, mirroring our recursive definition in part (a).

- Suppose $w = \varepsilon$. Then $|w| = 0$ is even, by definition of length.

$$\begin{aligned} evens(w \bullet x) &= evens(\varepsilon \bullet x) && \text{because } w = \varepsilon \\ &= evens(x) && \text{by definition of } \bullet \\ &= \varepsilon \bullet evens(x) && \text{by definition of } \bullet \\ &= evens(\varepsilon) \bullet evens(x) && \text{by definition of } evens \\ &= evens(w) \bullet evens(x) && \text{because } w = \varepsilon \end{aligned}$$

- Suppose $w = a$ for some symbol a. Then $|w| = 1$ is odd, by definition of length.

$$\begin{aligned} evens(w \bullet x) &= evens(a \bullet x) && \text{because } w = a \\ &= odds(x) && \text{by definition of } evens \\ &= \varepsilon \bullet odds(x) && \text{by definition of } \bullet \\ &= evens(a) \bullet odds(x) && \text{by definition of } evens \\ &= evens(w) \bullet odds(x) && \text{because } w = \varepsilon \end{aligned}$$

- Finally suppose $w = aby$ for some symbols a and b and some string y.[a] The definition of length implies $|w| = |aby| = |y| + 2$. We immediately have

$$\begin{aligned} evens(w \bullet x) &= evens(aby \bullet x) && \text{because } w = aby \\ &= evens(ab \bullet (y \bullet x)) && \text{because } \bullet \text{ is associative} \\ &= odds(b \bullet (y \bullet x)) && \text{by definition of } evens \\ &= b \cdot evens(y \bullet x) && \text{by definition of } evens \text{ or } odds \end{aligned}$$

  To complete this case of the proof, we consider two subcases, mirroring the cases in the statement we are proving.

  - Suppose $|y|$ is even. Then $|w| = |y| + 2$ is also even.

$$evens(w \bullet x) = b \cdot evens(y \bullet x) \qquad \text{proved above}$$

$$\begin{aligned}
&= b \cdot (\mathit{evens}(y) \bullet \mathit{evens}(x)) && \text{by the induction hypothesis}\\
&= (b \cdot \mathit{evens}(y)) \bullet \mathit{evens}(x)) && \text{because } \bullet \text{ is associative}\\
&= \mathit{odds}(by) \bullet \mathit{evens}(x) && \text{by definition of } \mathit{evens}\\
&= \mathit{evens}(aby) \bullet \mathit{evens}(x) && \text{by definition of } \mathit{evens} \text{ or } \mathit{odds}\\
&= \mathit{evens}(w) \bullet \mathit{evens}(x) && \text{because } w = aby
\end{aligned}$$

– Suppose $|y|$ is odd. Then $|w| = |y| + 2$ is also odd.

$$\begin{aligned}
\mathit{evens}(w \bullet x) &= b \cdot \mathit{evens}(y \cdot x) && \text{proved above}\\
&= b \cdot (\mathit{evens}(y) \bullet \mathit{odds}(x)) && \text{by the induction hypothesis}\\
&= (b \cdot \mathit{evens}(y)) \bullet \mathit{odds}(x)) && \text{because } \bullet \text{ is associative}\\
&= \mathit{odds}(by) \bullet \mathit{odds}(x) && \text{by definition of } \mathit{evens}\\
&= \mathit{evens}(aby) \bullet \mathit{odds}(x) && \text{by definition of } \mathit{evens} \text{ or } \mathit{odds}\\
&= \mathit{evens}(w) \bullet \mathit{odds}(x) && \text{because } w = aby
\end{aligned}$$

In all cases, we conclude that

$$\mathit{evens}(w \bullet x) = \begin{cases} \mathit{evens}(w) \bullet \mathit{evens}(x) & \text{if } |w| \text{ is even,}\\ \mathit{evens}(w) \bullet \mathit{odds}(x) & \text{if } |w| \text{ is odd.} \end{cases}$$

■

[a] We can't use the variable x here because it's already in use.

Solution (mutual induction): We actually prove *two* identities for all strings w and x, by mutual induction on w:

$$\mathit{evens}(w \bullet x) = \begin{cases} \mathit{evens}(w) \bullet \mathit{evens}(x) & \text{if } |w| \text{ is even,}\\ \mathit{evens}(w) \bullet \mathit{odds}(x) & \text{if } |w| \text{ is odd.} \end{cases}$$

$$\mathit{odds}(w \bullet x) = \begin{cases} \mathit{odds}(w) \bullet \mathit{odds}(x) & \text{if } |w| \text{ is even,}\\ \mathit{odds}(w) \bullet \mathit{evens}(x) & \text{if } |w| \text{ is odd.} \end{cases}$$

Let w and x be arbitrary strings.
Assume that the following identities hold for all strings y shorter than w:

$$\mathit{evens}(y \bullet x) = \begin{cases} \mathit{evens}(y) \bullet \mathit{evens}(x) & \text{if } |y| \text{ is even,}\\ \mathit{evens}(y) \bullet \mathit{odds}(x) & \text{if } |y| \text{ is odd.} \end{cases}$$

$$\mathit{odds}(y \bullet x) = \begin{cases} \mathit{odds}(y) \bullet \mathit{odds}(x) & \text{if } |y| \text{ is even,}\\ \mathit{odds}(y) \bullet \mathit{evens}(x) & \text{if } |y| \text{ is odd.} \end{cases}$$

There are **three** cases to consider.

- Suppose $w = \varepsilon$. Then $|w| = 0$ is even, by definition of length.

$$\begin{aligned}
\mathit{evens}(w \bullet x) &= \mathit{evens}(\varepsilon \bullet x) && \text{because } w = \varepsilon\\
&= \mathit{evens}(x) && \text{by definition of } \bullet\\
&= \varepsilon \bullet \mathit{evens}(x) && \text{by definition of } \bullet\\
&= \mathit{evens}(\varepsilon) \bullet \mathit{evens}(x) && \text{by definition of } \mathit{evens}\\
&= \mathit{evens}(w) \bullet \mathit{evens}(x) && \text{because } w = \varepsilon
\end{aligned}$$

$$\begin{aligned}
\mathit{odds}(w \bullet x) &= \mathit{odds}(\varepsilon \bullet x) && \text{because } w = \varepsilon\\
&= \mathit{odds}(x) && \text{by definition of } \bullet\\
&= \varepsilon \bullet \mathit{odds}(x) && \text{by definition of } \bullet\\
&= \mathit{odds}(\varepsilon) \bullet \mathit{odds}(x) && \text{by definition of } \mathit{odds}\\
&= \mathit{odds}(w) \bullet \mathit{odds}(x) && \text{because } w = \varepsilon
\end{aligned}$$

- Suppose $|w|$ is even and $w = ay$ for some symbol a and some string y. Then $|y|$ is odd, which implies the following:

$$\begin{aligned}
\mathit{evens}(w \bullet x) &= \mathit{evens}(ay \bullet x) && \text{because } w = ay\\
&= \mathit{evens}(a \cdot (y \bullet x)) && \text{because } \bullet \text{ is associative}\\
&= \mathit{odds}(y \cdot x) && \text{by definition of } \mathit{evens}\\
&= \mathit{odds}(y) \bullet \mathit{evens}(x) && \text{by the induction hypothesis}\\
&= \mathit{evens}(ay) \bullet \mathit{evens}(x)) && \text{by definition of } \mathit{evens}\\
&= \mathit{evens}(w) \bullet \mathit{evens}(x) && \text{because } w = ay
\end{aligned}$$

$$\begin{aligned}
\mathit{odds}(w \bullet x) &= \mathit{odds}(ay \bullet x) && \text{because } w = ay\\
&= \mathit{odds}(a \cdot (y \bullet x)) && \text{because } \bullet \text{ is associative}\\
&= a \cdot \mathit{evens}(y \bullet x) && \text{by definition of } \mathit{evens}\\
&= a \cdot (\mathit{evens}(y) \bullet \mathit{odds}(x)) && \text{by the induction hypothesis}\\
&= (a \cdot \mathit{evens}(y)) \bullet \mathit{odds}(x) && \text{because } \bullet \text{ is associative}\\
&= \mathit{odds}(ay) \bullet \mathit{odds}(x)) && \text{by definition of } \mathit{evens}\\
&= \mathit{odds}(w) \bullet \mathit{odds}(x) && \text{because } w = ay
\end{aligned}$$

- Suppose $|w|$ is odd. We must have $w = ay$ for some symbol a and some string y, such that $|y|$ is even.

$$\begin{aligned}
\mathit{evens}(w \bullet x) &= \mathit{evens}(ay \bullet x) && \text{because } w = ay\\
&= \mathit{evens}(a \cdot (y \bullet x)) && \text{because } \bullet \text{ is associative}\\
&= \mathit{odds}(y \cdot x) && \text{by definition of } \mathit{evens}\\
&= \mathit{odds}(y) \bullet \mathit{odds}(x) && \text{by the induction hypothesis}\\
&= \mathit{evens}(ay) \bullet \mathit{odds}(x)) && \text{by definition of } \mathit{evens}\\
&= \mathit{evens}(w) \bullet \mathit{odds}(x) && \text{because } w = ay
\end{aligned}$$

$$\begin{aligned}
odds(w \bullet x) &= odds(ay \bullet x) && \text{because } w = ay \\
&= odds(a \cdot (y \bullet x)) && \text{because } \bullet \text{ is associative} \\
&= a \cdot evens(y \bullet x) && \text{by definition of } evens \\
&= a \cdot (evens(y) \bullet evens(x)) && \text{by the induction hypothesis} \\
&= (a \cdot evens(y)) \bullet evens(x) && \text{because } \bullet \text{ is associative} \\
&= odds(ay) \bullet evens(x)) && \text{by definition of } evens \\
&= odds(w) \bullet evens(x) && \text{because } w = ay
\end{aligned}$$

In all three cases, we have proved both claimed identities. ■

Rubric: 8 points, standard induction rubric (scaled). **These are not the only correct proofs.**

Every variant of the first proof must consider four distinct cases, to capture the case distinctions in the recursive definition of strings, in the recursive definition of *evens*, and in the statement that we are proving:

- w is empty
- w has length 1
- w has positive even length
- w has odd length greater than 1

These cases can be clustered into coarser cases with subcases (as above), or presented as four flat cases. The even/odd cases can be based either on the length of w (implying similar conditions on $|y|$), or on the length of y (implying similar conditions on $|w|$.) But no matter how the proof is organized, its case structure should be clear *without reading the rest of the proof*.

Similarly. any variant of the second mutual-induction proof only requires three cases (empty, even and nonempty, and odd), but each case requires two separate arguments (for *evens* and *odds*). Again, these various cases and arguments can be correctly organized in several different ways, but the case structure must be clear without reading the rest of the proof.

No penalty for silently applying associativity without justification.

Finally, it is not necessary to use the standalone definition of *evens* from part (a). The lines in gray in the first proof show how to apply the original definitions of *evens* and *odds*.

*3. ***Practice only. Do not submit solutions.***

For each non-negative integer n, we recursively define two binary trees P_n and V_n, called the nth *Piṅgala tree* and the nth *Virahāṇka tree*, respectively.

- P_0 and V_0 are empty trees, with no nodes.
- P_1 and V_1 each consist of a single node.
- For any integer $n \geq 2$, the tree P_n consists of a root with two subtrees; the left subtree is a copy of P_{n-1}, and the right subtree is a copy of P_{n-2}.
- For any integer $n \geq 2$, the tree V_n is obtained from V_{n-1} by attaching a new right child to every leaf and attaching a new left child to every node that has only a right child.

(a) Prove that the tree P_n has exactly F_n leaves.

Solution: To make the presentation simpler, let me define some notation. Let ε denote the empty binary tree, and let (L, R) denote the non-empty binary tree with left subtree L and right subtree R. Let $\bullet = (\varepsilon, \varepsilon)$ denote the binary tree consisting of a single node. Then we can define Piṅgala trees more succinctly as follows:

$$P_n = \begin{cases} \varepsilon & \text{if } n = 0 \\ \bullet & \text{if } n = 1 \\ (P_{n-2}, P_{n-1}) & \text{otherwise} \end{cases}$$

Now we're ready for the proof:

Proof: Let n be an arbitrary non-negative integer.
Assume $\#\text{leaves}(P_m) = F_m$ for every non-negative integer $m < n$.
There are three cases to consider.

- Suppose $n = 0$. $P_0 = \varepsilon$ has no nodes, so $\#\text{leaves}(P_0) = 0 = F_0$.
- Suppose $n = 1$. The single node in $P_1 = \bullet$ is a leaf, so $\#\text{leaves}(P_1) = 1 = F_1$.
- Finally, suppose $n \geq 2$. The root of P_n is not a leaf, so

$$\begin{aligned} \#\text{leaves}(P_n) &= \#\text{leaves}(\,(P_{n-2}, P_{n-1})\,) && \text{by definition of } P_n \\ &= \#\text{leaves}(P_{n-2}) + \#\text{leaves}(P_{n-1}) && \text{root of } P_n \text{ is not a leaf} \\ &= F_{n-2} + F_{n-1} && \text{by ind. hyp.} \\ &= F_n && \text{by definition of } F_n \end{aligned}$$

In all cases, we conclude that $\#\text{leaves}(P_n) = F_n$. ■

Rubric: Standard induction rubric. Weak induction *cannot* work here.

(b) Prove that the tree V_n has exactly F_n leaves.
[Hint: You need to prove a stronger result.]

Solution: To simplify the presentation, let $\#(d, T)$ denote the number of nodes in the tree T with exactly d children. In particular, $\#(0, T)$ is the number of leaves in T. We need to prove that $\#(0, V_n) = F_n$ for all $n \geq 0$.

This claim is trivial when $n = 0$. For all $n > 0$, I'll actually prove the stronger claim that $\#(0, V_n) = F_n$ and $\#(1, V_n) = F_{n-1}$.

Let n be an arbitrary positive integer. Assume for all positive integers $m < n$ that $\#(0, V_m) = F_m$ and $\#(1, V_m) = F_{m-1}$. There are two cases to consider, mirroring the definition of V_n.

- Suppose $n = 1$. Then by definition, V_1 has a single node, which is a leaf, so

$$\begin{aligned} \#(0, V_n) &= \#(0, V_1) = 1 = F_1 = F_n \\ \#(1, V_n) &= \#(1, V_1) = 0 = F_0 = F_{n-1} \end{aligned}$$

- Suppose $n \geq 2$. Then each leaf of V_n is a child of either a leaf of V_{n-1} or a node with one child in V_{n-1}. Conversely, each node with zero or one children in V_{n-1} is the parent of exactly one leaf in V_n. So the inductive hypothesis implies

$$\#(0, V_n) = \#(0, V_{n-1}) + \#(1, V_{n-1}) = F_{n-1} + F_{n-2} = F_n.$$

  Similarly, each node with one child in V_n is a leaf in V_{n-1}, so the inductive hypothesis implies $\#(1, V_n) = \#(0, V_{n-1}) = F_{n-1}$.

In both cases, we conclude that $\#(0, V_n) = F_n$ and $\#(1, V_n) = F_{n-1}$. ■

Rubric: Standard induction rubric. We do need to consider the case $n = 0$ outside the main induction argument; our stronger claim is false when $n = 0$, because $F_{-1} = 1$!

(c) Prove that the trees P_n and V_n are identical, for all $n \geq 0$.

Solution: Let *T.left* and *T.right* denote the left and right subtrees of any non-empty tree T. For any tree T, we recursively define

$$sprout(T) = \begin{cases} \bullet & \text{if } T = \varepsilon \\ (\varepsilon, \bullet) & \text{if } T = \bullet \\ (sprout(T.left), sprout(T.right)) & \text{otherwise} \end{cases}$$

Then we can rewrite the definition of V_n as $V_n = sprout(V_{n-1})$ for all $n \geq 1$.

Let n be an arbitrary non-negative integer. Assume for all non-negative integers $m < n$ that $P_n = V_n$. There are four cases to consider.

- $P_0 = \varepsilon$ and $V_0 = \varepsilon$ by definition.
- $P_1 = \bullet$ and $V_1 = \bullet$ by definition.
- $P_2 = (P_0, P_1) = (\varepsilon, \bullet)$ and $V_2 = sprout(V_1) = sprout(\bullet) = (\varepsilon, \bullet)$ by definition.
- Finally, if $n \geq 3$, we have

$$\begin{aligned} V_n &= sprout(V_{n-1}) && \text{by definition} \\ &= sprout(P_{n-1}) && \text{by the inductive hypothesis} \\ &= sprout(\,(P_{n-3}, P_{n-2})\,) && \text{by definition of } P_n \\ &= sprout(\,(V_{n-3}, V_{n-2})\,) && \text{by the inductive hypothesis (twice)} \\ &= (\,sprout(V_{n-3}), sprout(V_{n-2})\,) && \text{by definition of } sprout\text{, since } n \geq 4 \\ &= (V_{n-2}, V_{n-1}) && \text{by definition of } V_n \\ &= (P_{n-2}, P_{n-1}) && \text{by the inductive hypothesis (twice)} \\ &= P_n && \text{by definition of } P_n \end{aligned}$$

In all cases, we conclude that $V_n = P_n$. ■

Rubric: Standard induction rubric. Weak induction cannot work here. We need to consider the case $n = 2$ separately, because the main inductive proof refers to V_{n-3} and P_{n-3}.

Watch for switching between P_m and V_m without explicitly invoking the induction hypothesis. Yes, we really do have to invoke the induction hypothesis five times!