1. For any string $w \in \{0,1\}^*$, let $sink(w)$ be the function defined recursively as follows:

$$sink(w) := \begin{cases} w & \text{if } |w| \le 1 \\ a \cdot sink(1x) & \text{if } w = 1ax \text{ for some } a \in \{0,1\} \text{ and } x \in \{0,1\}^* \\ 0 \cdot sink(ax) & \text{if } w = 0ax \text{ for some } a \in \{0,1\} \text{ and } x \in \{0,1\}^* \end{cases}$$

(a) Prove that $|sink(w)| = |w|$.

Solution: Let w be an arbitrary string.
Assume $|sink(y)| = |y|$ for every string y such that $|y| < |w|$.
There are three cases to consider (mirroring the definition of *sink*):

- If $|w| \le 1$, then

$$|sink(w)| = |w| \qquad \text{by definition of } sink$$

- If $w = 1ax$ for some $a \in \{0,1\}$ and $x \in \{0,1\}^*$, then

$$\begin{aligned} |sink(w)| &= |sink(1ax)| && \text{because } w = 1ax \\ &= |a \cdot sink(1x)| && \text{by definition of } sink \\ &= 1 + |sink(1x)| && \text{by definition of } |\cdot| \\ &= 1 + |1x| && \text{by the inductive hypothesis} \\ &= 2 + |x| && \text{by definition of } |\cdot|\text{; arithmetic} \\ &= |1a| + |x| && \text{by definition of } |\cdot| \\ &= |1ax| && \text{because } |y \bullet z| = |y| + |z| \\ &= |w| && \text{because } w = 1ax \end{aligned}$$

- Finally, if $w = 0ax$ for some $a \in \{0,1\}$ and $x \in \{0,1\}^*$, then

$$\begin{aligned} |sink(w)| &= |sink(0ax)| && \text{because } w = 0ax \\ &= |0 \cdot sink(ax)| && \text{by definition of } sink \\ &= 1 + |sink(ax)| && \text{by definition of } |\cdot| \\ &= 1 + |ax| && \text{by the inductive hypothesis} \\ &= 2 + |x| && \text{by definition of } |\cdot|\text{; arithmetic} \\ &= |0a| + |x| && \text{by definition of } |\cdot| \\ &= |0ax| && \text{because } |y \bullet z| = |y| + |z| \\ &= |w| && \text{because } w = 0ax \end{aligned}$$

In all cases, we conclude that $|sink(w)| = |w|$. ■

Rubric: 4 points: Standard induction rubric (scaled)

(b) Prove that if $\#(1, w) \geq 1$, then $sink(w) = x \bullet 1$ for some string x.

Solution: Let w be an arbitrary string such that $\#(1, w) \geq 1$.

Assume, for all strings y such that $|y| < |w|$ and $\#(1, y) \geq 1$, we have $sink(y) = x \bullet 1$ for some string x.

There are three cases to consider:

- Suppose $|w| \leq 1$. Then, $w = 1$, because $\#(1, w) \geq 1$. Further,

$$\begin{aligned} sink(w) &= w && \text{by definition of } sink \\ &= 1 && \text{because } w = 1 \\ &= \varepsilon \bullet 1 && \text{by definition of } \bullet \end{aligned}$$

  Therefore, $sink(w) = x \bullet 1$ with $x = \varepsilon$.

- If $w = 1ay$ for some $a \in \{0, 1\}$ and $y \in \{0, 1\}^*$, then

$$\begin{aligned} sink(w) &= sink(1ay) && \text{because } w = 1ay \\ &= a \cdot sink(1y) \end{aligned}$$

  We have $\#(1, 1y) \geq 1$ by the definition of $\#$. By the inductive hypothesis, $sink(1y) = z \bullet 1$ for some string z. Therefore, $sink(w) = x \bullet 1$ with $x = az$.

- Finally, if $w = 0ay$ for some $a \in \{0, 1\}$ and $y \in \{0, 1\}^*$, then

$$\begin{aligned} sink(w) &= sink(0ay) && \text{because } w = 0ay \\ &= 0 \cdot sink(ay) \end{aligned}$$

  We have $\#(1, ay) \geq 1$ by the definition of $\#$ and basic arithmetic. By the inductive hypothesis, $sink(ay) = z \bullet 1$ for some string z. Therefore, $sink(w) = x \bullet 1$ with $x = 0z$.

In all cases, we conclude that $sink(w) = x \bullet 1$ for some string x. ■

Rubric: 3 points: standard induction rubric (scaled)

(c) Prove that if $sink(w) = x \bullet 1$ for some string x, then $\#(1, w) \geq 1$.

Solution: Let w be an arbitrary string such that $sink(w) = x \bullet 1$ for some string x.

Assume, for all strings y such that $|y| < |w|$ and $sink(w) = x \bullet 1$ for some string x, we have $\#(1, w) \geq 1$. There are three cases to consider:

- If $|w| \leq 1$, then

$$sink(w) = w \qquad \text{by definition of } sink$$

  Therefore, $w = x \bullet 1$ for some string x. Because $|w| \leq 1$, we have $w = 1$ and $\#(1, w) = 1$.
- If $w = 1ay$ for some $a \in \{0, 1\}$ and $y \in \{0, 1\}^*$, then $\#(1, w) \geq 1$ by definition of $\#$. We're already done with this case!
- Finally, if $w = 0ay$ for some $a \in \{0, 1\}$ and $y \in \{0, 1\}^*$, then

$$\begin{aligned} sink(w) &= sink(0ay) && \text{because } w = 0ay \\ &= 0 \cdot sink(ay) \end{aligned}$$

  By assumption, $sink(w) = x \bullet 1$ for some string x beginning with 0. Define z so that $x = 0z$. We have $sink(ay) = z \bullet 1$. Therefore,

$$\begin{aligned} \#(1, w) &= \#(1, 0ay) && \text{because } w = 0ay \\ &= \#(1, ay) && \text{by definition of } \# \\ &\geq 1 && \text{by the inductive hypothesis} \end{aligned}$$

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

Rubric: 3 points: standard induction rubric (scaled)

2. Consider the following 3 sets of strings $L_0, L_1, L_2 \subseteq \{0,1\}^*$ defined (mutually) recursively as follows:

   - The empty string ε is in L_0.
   - For any $i \in \{0,1,2\}$ and any string $x \in L_i$, the string $0x$ is in $L_{i-1 \pmod 3}$ and the string $1x$ is in $L_{i+1 \pmod 3}$.
   - These are the only strings in L_0, L_1, and L_2.

   (a) Prove that for each $i \in \{0,1,2\}$ and string $w \in L_i$, we have $\#(1,w) \equiv \#(0,w) + i \pmod 3$.

Solution: Let i be an arbitrary member of $\{0,1,2\}$, and let w be an arbitrary string in L_i.

Assume, for any $j \in \{0,1,2\}$ and any string $x \in L_j$ where $|x| < |w|$, that $\#(1,x) \equiv \#(0,x) + j \pmod 3$.

There are three cases to consider.

- Suppose $w = \varepsilon$. We must have $i = 0$, because L_1 and L_2 contain only non-empty strings. Then,

$$\begin{aligned} \#(1,w) &\equiv \#(1,\varepsilon) && \text{because } w = \varepsilon \\ &\equiv 0 && \text{by definition of } \# \\ &\equiv \#(0,\varepsilon) + 0 && \text{by definition of } \#\text{; arithmetic} \\ &\equiv \#(0,w) + i \pmod 3 && \text{because } i = 0 \text{ and } w = \varepsilon \end{aligned}$$

- Suppose $w = 0x$ for some $j \in \{0,1,2\}$ and string $x \in L_j$. By the definitions above, $i = j - 1 \pmod 3$.

$$\begin{aligned} \#(1,w) &\equiv \#(1,0x) && \text{because } w = 0x \\ &\equiv \#(1,x) && \text{by definition of } \# \\ &\equiv \#(0,x) + j && \text{by the inductive hypothesis} \\ &\equiv \#(0,0x) + j - 1 && \text{by definition of } \#\text{; arithmetic} \\ &\equiv \#(0,w) + i \pmod 3 && \text{because } i = j - 1 \pmod 3 \text{ and } w = 0x \end{aligned}$$

- Suppose $w = 1x$ for some $j \in \{0,1,2\}$ and string $x \in L_j$. By the definitions above, $i = j + 1 \pmod 3$.

$$\begin{aligned} \#(1,w) &\equiv \#(1,1x) && \text{because } w = 1x \\ &\equiv \#(1,x) + 1 && \text{by definition of } \# \\ &\equiv \#(0,x) + j + 1 && \text{by the inductive hypothesis} \\ &\equiv \#(0,1x) + j + 1 && \text{by definition of } \#\text{; arithmetic} \\ &\equiv \#(0,w) + i \pmod 3 && \text{because } i = j + 1 \pmod 3 \text{ and } w = 1x \end{aligned}$$

In all three cases, we conclude that $\#(1, w) \equiv \#(0, w) + i \pmod 3$. ■

Rubric: 5 points: standard induction rubric (scaled)

(b) Prove that for each $i \in \{0,1,2\}$, the set L_i contains every string $w \in \{0,1\}^*$ such that $\#(1,w) \equiv \#(0,w) + i \pmod 3$.

Solution: Let i be an arbitrary member of $\{0,1,2\}$, and let w be an arbitrary string in $\{0,1\}^*$ such that $\#(1,w) \equiv \#(0,w) + i \pmod 3$.

Assume, for any $j \in \{0,1,2\}$, set L_j contains every string x such that $|x| < |w|$ and $\#(1,x) \equiv \#(0,x) + j \pmod 3$.

There are three cases to consider.

- Suppose $w = \varepsilon$.

$$\begin{aligned}
\#(1,w) &\equiv \#(1,\varepsilon) && \text{because } w = \varepsilon \\
&\equiv 0 && \text{by definition of } \# \\
&\equiv \#(0,\varepsilon) + 0 && \text{by definition of } \#\text{; arithmetic} \\
&\equiv \#(0,w) + 0 \pmod 3 && \text{because } w = \varepsilon
\end{aligned}$$

  Indeed, $w \in L_0$ by definition.

- Suppose $w = 0x$ for some string x. Then,

$$\begin{aligned}
\#(1,x) &\equiv \#(1,0x) && \text{by definition of } \# \\
&\equiv \#(1,w) && \text{because } w = 0x \\
&\equiv \#(0,w) + i && \text{by definition of } i \text{ and } w \\
&\equiv \#(0,0x) + i && \text{because } w = 0x \\
&\equiv \#(0,x) + i + 1 \pmod 3 && \text{by definition of } \#\text{; arithmetic}
\end{aligned}$$

  By the inductive hypothesis, $x \in L_{i+1 \pmod 3}$. Therefore, $w = 0x$ is in L_i.

- Suppose $w = 1x$ for some string x. Then,

$$\begin{aligned}
\#(1,x) &\equiv \#(1,1x) - 1 && \text{by definition of } \# \\
&\equiv \#(1,w) - 1 && \text{because } w = 1x \\
&\equiv \#(0,w) + i - 1 && \text{by definition of } i \text{ and } w \\
&\equiv \#(0,1x) + i - 1 && \text{because } w = 1x \\
&\equiv \#(0,x) + i - 1 \pmod 3 && \text{by definition of } \#\text{; arithmetic}
\end{aligned}$$

  By the inductive hypothesis, $x \in L_{i-1 \pmod 3}$. Therefore, $w = 1x$ is in L_i.

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

Rubric: 5 points: standard induction rubric (scaled). The modulo arithmetic is more detailed than necessary for full credit.

*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!