# CS 173, Spring 18 Examlet 9, Part A 

**Name:**________________________________________

**NetID:**____________________ **Lecture:** **A** **B**

**Discussion:** **Thursday** **Friday** **9** **10** **11** **12** **1** **2** **3** **4** **5** **6**

(18 points) Recall that $\epsilon$ is shorthand for the empty (zero-length) string. Here is a grammar $G$, with start symbol $S$ and terminal symbols $a$ and $b$.

$$S \rightarrow a\ S\ b\ S \mid b\ S\ a\ S \mid \epsilon$$

Use (strong) induction to prove that any string with equal numbers of a's and b's can be generated by grammar $G$. That is, show how to build parse trees for these strings. You can use (without proof) this fact (and the similar fact with a and b swapped).

> Fact: Suppose the number of a's in a string $w$ is one more than the number of b's in $w$. Then we can divide $w$ into $w = xay$, where string $x$ (and therefore also string $y$) has equal numbers of a's and b's.

**Solution:** The induction variable is named ___**h**___ and it is the ___**number of a's**___ of/in the string.

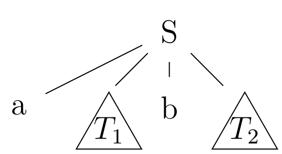
**Base Case(s)**: At $h = 0$, there is only one string with 0 a's and the same number of b's, i.e. $\epsilon$. It can be generated by $G$ as follows:

S<br>|<br>$\epsilon$

**Inductive Hypothesis** [Be specific, don't just refer to "the claim"]: Suppose that any string with equal numbers of a's and b's can be generated by grammar $G$, for strings with $h$ a's (length $2h$) where $h = 0, 2, \ldots, k-1$ ($k \geq 1$).

**Inductive Step:** Let $v$ be a string of length $2k$ with equal numbers of a's and b's. Since $k \geq 1$, there are two cases

Case 1: $v$ starts with an a. Then $v = aw$. The number of b's in $w$ must be one more than the number of a's. Using the Fact, we can divide $w$ as $w = xby$ where $x$ and $y$ each have equal numbers of a's and b's. So $v = axby$. Since $x$ has equal numbers of a's and b's, we can build a parse tree $T_1$ for $x$. Similarly, we can build a parse tree $T_2$for $y$. We can join these into a parse tree for $v$ using the rule $S \rightarrow a\ S\ b\ S$ like this




Case 2: $v$ starts with a b. Then $v = bw$. The number of a's in $w$ must be one more than the number of b's. Using the Fact, we can divide $w$ as $w = xay$. So $v = bxay$. By the inductive hypothesis, we can build parse trees for $x$ and $y$. We can join these into a parse tree for $v$ using the rule $S \rightarrow b\ S\ a\ S$.

In both cases, we can build a parse tree for the string $v$, which is what we needed to show.

**Name:**________________________________________

**NetID:**________________________ **Lecture:** **A** **B**

**Discussion:** **Thursday** **Friday** **9** **10** **11** **12** **1** **2** **3** **4** **5** **6**

(18 points) A Vintage tree is a binary tree in which each node X contains an integer label $v(X)$ such that

- If $X$ is a leaf, $v(X)$ is 7, 23, or 31.
- If $X$ has one child $Y$, then $v(X) = v(Y) + 7$.
- If $X$ has two children $Y$ and $Z$, then $v(X) = v(Y)v(Z)$.

Use strong induction to prove that the value in the root of a Vintage tree is always positive.

**Solution:** The induction variable is named **h** and it is the **height** of/in the tree.

**Base Case(s):** $h = 0$. The tree consists of a single leaf. $v(X)$ is 7, 23, or 31, all of which are positive.

**Inductive Hypothesis** [Be specific, don't just refer to "the claim"]: The value in the root of a Vintage tree is always positive, for tree height $h = 0, 1, \ldots, k-1$ ($k \geq 1$).

**Inductive Step:** Let $T$ be a Vintage tree of height $k$, with root $R$. There are two cases:

Case 1: $R$ has one child $Y$. By the inductive hypothesis, $v(Y)$ is positive. So $v(R) = v(Y) + 7$ must be positive.

Case 2: $R$ has two children $Y$ and $Z$. By the inductive hypothesis, $v(Y)$ and $v(Z)$ are positive. So $v(R) = v(Y)v(Z)$ must also be positive.

In both cases $v(R)$ is positive, which is what we needed to prove.

# CS 173, Spring 18 Examlet 9, Part A

**Name:**________________________________________

**NetID:**____________________ **Lecture:** **A** **B**

**Discussion:** **Thursday** **Friday** **9** **10** **11** **12** **1** **2** **3** **4** **5** **6**

(18 points) Orange trees are binary trees whose nodes are labelled with strings, such that

- Each leaf node has label `tip`, `top`, or `tack`
- If a node has one child, it has label $\alpha\alpha$ where where $\alpha$ is the child's label. E.g. if the child has label `top` then the parent has `toptop`.
- If a node has two children, it contains $\alpha\beta$ where $\alpha$ and $\beta$ are the child labels. E.g. if the children have labels `tip` and `top`, then the parent has label `tiptop`.

Let S(n) be the length of the label on node n. Let L(n) be the number of leaves in the subtree rooted at n. Use (strong) induction to prove that $S(n) \geq 3L(n)$ if $n$ is the root node of any Orange tree.

**Solution:** The induction variable is named **h** and it is the **height** of/in the tree.

**Base case(s):** $h = 0$. The tree consists of a single leaf node, so $L(n) = 1$. The node has label `tip`, `top`, or `tack`, so $S(n) \geq 3$. So $S(n) \geq 3L(n)$.

**Inductive hypothesis** [Be specific, don't just refer to "the claim"]:

Suppose that $S(n) \geq 3L(n)$ if $n$ is the root node of any Orange tree of height $< k$ (where $k \geq 1$).

**Rest of the inductive step:**

Suppose that $T$ is a Orange tree of height $k$. There are two cases:

Case 1: The root $n$ of $T$ has a single child node $p$. By the inductive hypothesis $S(p) \geq 3L(p)$. $L(n) = L(p)$. And $S(n) = 2S(p) \geq 6L(p) = 6L(n) \geq 3L(n)$. So $S(n) \geq 3L(n)$.

Case 2: The root $n$ of $T$ has two children $p$ and $q$. By the inductive hypothesis $S(p) \geq 3L(p)$ and $S(q) \geq 3L(q)$.

Notice that $L(n) = L(p) + L(q)$. And $S(n) = S(p) + S(q)$.

So $S(n) = S(p) + S(q) \geq 3L(p) + 3L(q) = 3L(n)$.

# CS 173, Spring 18 Examlet 9, Part A

**Name:**________________________________________

**NetID:**________________________ **Lecture: A B**

**Discussion: Thursday Friday 9 10 11 12 1 2 3 4 5 6**

(18 points) Recall that a node in a full binary tree is either a leaf or has exactly two children. A Illini tree is a full binary tree whose nodes contain a color (blue or orange) plus a positive integer, such that

- All leaf nodes have label $(\text{blue}, 1)$.
- If an internal node has label $(\text{orange}, p)$, then both its children must have label $(\text{blue}, p)$.
- If an internal node has label $(\text{blue}, p)$, then its children must have label $(\text{blue}, p-1)$ or $(\text{orange}, p-1)$.

Use (strong) induction to prove that a Illini tree with root label $(\text{blue}, p)$ or $(\text{orange}, p)$ has at least $2^p - 1$ nodes.

**Solution:** The induction variable is named **h** and it is the **height** of/in the tree.

**Base Case(s):** At $h = 0$, the tree contains exactly one node, which is a leaf and therefore labelled $(\text{blue}, 1)$. The number of nodes is 1 which equals $2^1 - 1$.

**Inductive Hypothesis** [Be specific, don't just refer to "the claim"]: Suppose that every Illini tree with root label $(\text{blue}, p)$ or $(\text{orange}, p)$ has at least $2^p - 1$ nodes, for heights $h = 0, \ldots, k-1$.

**Inductive Step:** Consider a Illini tree $T$ with height $k$. We can assume that $k > 1$, so the root node is not a leaf. There are two cases:

Case 1: the root node of $T$ has label $(\text{orange}, p)$ for some integer $p$. Then its left child must have label $(\text{blue}, p)$. By the induction hypothesis, the tree rooted at the left child must have at least $2^p - 1$ nodes. Therefore, $T$ must have at least $2^p - 1$ nodes.

Case 2: the root node of $T$ has label $(\text{blue}, p)$ for some integer $p$. Then its children have label $(\text{blue}, p-1)$ or $(\text{orange}, p-1)$. By the induction hypothesis the tree rooted at the left child must have at least $2^{p-1} - 1$ nodes, and similarly for the tree rooted at the right child. The number of nodes in $T$ is the sum of these two node counts, plus one for the root node of $T$. So $T$ must have at least $2(2^{p-1} - 1) + 1 = 2^p - 1$ nodes.

In both cases, $T$ must have at least $2^p - 1$ nodes, which is what we needed to prove.

**Name:**________________________________________

**NetID:**________________________ **Lecture: A B**

**Discussion: Thursday Friday 9 10 11 12 1 2 3 4 5 6**

(18 points) Lemon trees have nodes labelled with integer values such that:

- Every leaf node has label 5 or 7.
- An internal node with label 0 has exactly three children.
- An internal node with label 1 has exactly two children.

The "total value" of a Lemon tree is the sum of the labels on all its nodes. Use (strong) induction that the total value of any Lemon tree is odd. You may assume basic divisibility facts e.g. the sum of two odd numbers is even.

**Solution:** The induction variable is named **h** and it is the **height** of/in the tree.

**Base Case(s):** Lemon trees of height $h = 0$ consist of a single node containing either 5 or 7. So the total value of the tree is 5 or 7, both of which are odd.

**Inductive Hypothesis** [Be specific, don't just refer to "the claim"]: Suppose that that the total value of any Lemon tree is odd, for trees of height $h = 0, 1, \ldots, k-1$. ($k \geq 1$).

**Inductive Step:** Let $T$ be a Lemon tree of height $k$. There are two cases:

Case 1: The root of $T$ contains 0, and it has three child subtrees. By the inductive hypothesis, the total value of each child subtree is odd. Since the sum of three odd numbers is odd, the total value of $T$ is also odd.

Case 2: The root of $T$ contains 1 and it has two child subtrees. By the inductive hypothesis, the total value of each child subtree is odd. The total value of $T$ is the sum of the total values of these child subtrees, plus 1. Again, we have the sum of three odd numbers, so the total value of $T$ must be odd.

In both cases, the total value of $T$ is odd, which is what we needed to show.

## CS 173, Spring 18 Examlet 9, Part A



**Name:**________________________________________

**NetID:**____________________ **Lecture: A B**

**Discussion: Thursday Friday 9 10 11 12 1 2 3 4 5 6**

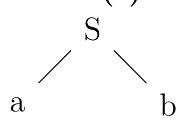
(18 points) Here is a grammar $G$, with start symbol $S$ and terminal symbols $a$ and $b$.

$$S \rightarrow a\,S\,b \;|\; b\,S\,a \;|\; S\,S \;|\; a\,b$$

Use (strong) induction to prove that any tree matching (aka generated by) grammar $G$ has equal numbers of a's and b's. Use $A(T)$ and $B(T)$ as shorthand for the number of a's and b's in a tree $T$.

**Solution:** The induction variable is named __**h**__ and it is the __**height**__ of/in the tree.

**Base Case(s)**: At $h = 1$, there is only one tree matching grammar $G$, which looks like




This contains the same number of a's and b's.

**Inductive Hypothesis** [Be specific, don't just refer to "the claim"]: Suppose that any tree matching (aka generated by) grammar $G$ has equal numbers of a's and b's, for trees with heights $h = 1, 2, \ldots, k-1$ ($k \geq 1$).

**Inductive Step:** Let $T$ be a tree of height $k$ matching grammar $G$. Let $r$ be the root of $T$. Since $k \geq 1$, there are three cases

Case 1: The children of $r$ have labels (left to right): $a$, $S$, $b$. The middle child is the root of a subtree $T_1$. By the inductive hypothesis, $T_1$ contains equal numbers of a's and b's. $A(T) = A(T_1) + 1$ and $B(T) = B(T_1) + 1$. So $A(T) = B(T)$.

Case 2: The children of $r$ have labels (left to right): $b$, $S$, $a$. The middle child is the root of a subtree $T_1$. This is exactly like Case 1.

Case 3: $r$ has two child subtrees $T_1$ and $T_2$, both with a root labelled $S$. By the inductive hypothesis, $A(T_1) = B(T_1)$ and $A(T_2) = B(T_2)$. But all the a's and b's in $T$ must live in the two subtrees, so $A(T) = A(T_1) + A(T_2)$ and $B(T) = B(T_1) + B(T_2)$. Combining these equations, we get that $A(T) = B(T)$.

In both cases $A(T) = B(T)$, which is what we needed to show.