1. Structural Induction

(a) Consider the following recursive definition of strings $\Sigma^*$ over an alphabet $\Sigma$.

**Basis Step:** $\epsilon$ is a string

**Recursive Step:** If $w$ is a string and $a \in \Sigma$ is a character then $wa$ is a string.

Recall the following recursive definition of the function $\text{len}$:

$$
\begin{align*}
\text{len}(\epsilon) &= 0 \\
\text{len}(wa) &= 1 + \text{len}(w)
\end{align*}
$$

Now, consider the following recursive definition:

$$
\text{double}(\epsilon) = \epsilon \\
\text{double}(wa) = \text{double}(w)aa
$$

Prove that for any string $x$, $\text{len}(\text{double}(x)) = 2\text{len}(x)$.

**Solution:**

For a string $x$, let $P(x)$ be "$\text{len}(\text{double}(x)) = 2\text{len}(x)$". We prove $P(x)$ for all strings $x \in \Sigma^*$ by structural induction.

**Base Case.** We show $P(\epsilon)$ holds. By definition $\text{len}(\text{double}(\epsilon)) = \text{len}(\epsilon) = 0$. On the other hand, $2\text{len}(\epsilon) = 0$ as desired.

**Induction Hypothesis.** Suppose $P(w)$ holds for some arbitrary string $w \in \Sigma^*$.

**Induction Step.** We show that $P(wa)$ holds for any character $a \in \Sigma$.

$$
\begin{align*}
\text{len}(\text{double}(wa)) &= \text{len}(\text{double}(w)aa) & \text{[By Definition of double]} \\
&= 1 + \text{len}(\text{double}(w)a) & \text{[By Definition of len]} \\
&= 1 + 1 + \text{len}(\text{double}(w)) & \text{[By Definition of len]} \\
&= 2 + 2\text{len}(w) & \text{[By IH]} \\
&= 2(1 + \text{len}(w)) & \text{[Algebra]} \\
&= 2(\text{len}(wa)) & \text{[By Definition of len]}
\end{align*}
$$

This proves $P(wa)$.

Thus, $P(x)$ holds for all strings $x \in \Sigma^*$ by structural induction.

(b) Consider the following definition of a (rooted binary) Tree:

**Basis Step:** $\bullet$ is a Tree.

**Recursive Step:** If $L$ is a Tree and $R$ is a Tree then Tree($\bullet$, $L$, $R$) is a Tree.

The function $\text{leaves}$ returns the number of leaves of a Tree. It is defined as follows:

$$
\begin{align*}
\text{leaves}(\bullet) &= 1 \\
\text{leaves}(& \text{Tree}(\bullet, L, R)) = \text{leaves}(L) + \text{leaves}(R)
\end{align*}
$$

Also, recall the definition of size on trees:

$$
\begin{align*}
|\bullet| &= 1 \\
|\text{Tree}(\bullet, L, R)| &= 1 + |L| + |R|
\end{align*}
$$

Prove that $\text{leaves}(T) \geq |T|/2 + 1/2$ for all Trees $T$. 


2. Regular Expressions

(a) Write a regular expression that matches base 10 non-negative numbers (e.g., there should be no leading zeroes).

**Solution:**

\[
0 \cup ((1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)(0 \cup 1 \cup 2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7 \cup 8 \cup 9)^*)
\]

(b) Write a regular expression that matches all non-negative base-3 numbers that are divisible by 3.

**Solution:**

\[
0 \cup ((1 \cup 2)(0 \cup 1 \cup 2)^*0)
\]

(c) Write a regular expression that matches all binary strings that contain the substring “111”, but not the substring “000”.

**Solution:**

\[
(01 \cup 001 \cup 1^*)^* (0 \cup 00 \cup \varepsilon)111 (01 \cup 001 \cup 1^*)^* (0 \cup 00 \cup \varepsilon)
\]

(If you don’t want the substring 000, the only way you can produce 0s is if there are only one or two 0s in a row, and they are immediately followed by a 1 or the end of the string.)