# CSE 311: Foundations of Computing I

## Section 7: Structural Induction and Regular Expressions Solutions

## 1. Strong Induction repeat question

Xavier Cantelli owns some rabbits. The number of rabbits he has in any given year is described by the function $f$:

$$f(0) = 0$$
$$f(1) = 1$$
$$f(n) = 2f(n-1) - f(n-2) \text{ for } n \geq 2$$

Determine, with proof, the number, $f(n)$, of rabbits that Cantelli owns in year $n$.

**Solution:**

Let $P(n)$ be "$f(n) = n$". We prove that $P(n)$ is true for all $n \in \mathbb{N}$ by strong induction on $n$.

**Base Cases** $(n = 0, n = 1)$**:** $f(0) = 0$ and $f(1) = 1$ by definition.

**Induction Hypothesis:** Assume that $P(0) \wedge P(1) \wedge \ldots P(n-1)$ are true for some fixed but arbitrary $n-1 \geq 1$.

**Induction Step:** We show $P(n)$:

$$\begin{aligned} f(n) &= 2f(n-1) - f(n-2) && \text{[Definition of } f\text{]} \\ &= 2(n-1) - (n-2) && \text{[Induction Hypothesis]} \\ &= n && \text{[Algebra]} \end{aligned}$$

Therefore, $P(n)$ is true for all $n \in \mathbb{N}$.

## 2. Structural Induction

(a) Consider the following recursive definition of strings.

**Basis Step:** "" is a string

**Recursive Step:** If $X$ is a string and $c$ is a character then append$(c, X)$ is a string.

Recall the following recursive definition of the function len:

$$\begin{aligned} \text{len}(\text{""}) &= 0 \\ \text{len}(\text{append}(c, X)) &= 1 + \text{len}(X) \end{aligned}$$

Now, consider the following recursive definition:

$$\begin{aligned} \text{double}(\text{""}) &= \text{""} \\ \text{double}(\text{append}(c, X)) &= \text{append}(c, \text{append}(c, \text{double}(X))). \end{aligned}$$

Prove that for any string $X$, len(double$(X)$) $= 2$len$(X)$.

**Solution:**

For a string $X$, let $\mathsf{P}(X)$ be "$\mathsf{len}(\mathsf{double}(X)) = 2\mathsf{len}(X)$. We prove $\mathsf{P}(X)$ for all strings $X$ by structural induction.

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

**Induction Hypothesis.** Suppose $\mathsf{P}(X)$ holds for some arbitrary string $X$.

**Induction Step.** We show that $\mathsf{P}(\mathsf{append}(c, X))$ holds for any character $c$.

$$\begin{aligned}
\mathsf{len}(\mathsf{double}(\mathsf{append}(c, X))) &= \mathsf{len}(\mathsf{append}(c, \mathsf{append}(c, \mathsf{double}(X)))) && \text{[By Definition of double]} \\
&= 1 + \mathsf{len}(\mathsf{append}(c, \mathsf{double}(X))) && \text{[By Definition of len]} \\
&= 1 + 1 + \mathsf{len}(\mathsf{double}(X)) && \text{[By Definition of len]} \\
&= 2 + 2\mathsf{len}(X) && \text{[By IH]} \\
&= 2(1 + \mathsf{len}(X)) && \text{[Algebra]} \\
&= 2(\mathsf{len}(\mathsf{append}(c, X))) && \text{[By Definition of len]}
\end{aligned}$$

This proves $\mathsf{P}(\mathsf{append}(c, X))$.

Thus, $\mathsf{P}(X)$ holds for all strings $X$ by structural induction.

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

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

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

The function leaves returns the number of leaves of a **Tree**. It is defined as follows:

$$\begin{aligned}
\mathsf{leaves}(\bullet) &= 1 \\
\mathsf{leaves}(\texttt{Tree}(\bullet, L, R)) &= \mathsf{leaves}(L) + \mathsf{leaves}(R)
\end{aligned}$$

Also, recall the definition of size on trees:

$$\begin{aligned}
\mathsf{size}(\bullet) &= 1 \\
\mathsf{size}(\texttt{Tree}(\bullet, L, R)) &= 1 + \mathsf{size}(L) + \mathsf{size}(R)
\end{aligned}$$

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

**Solution:**

For a tree $T$, let $\mathsf{P}(T)$ be $\mathsf{leaves}(T) \geq \mathsf{size}(T)/2 + 1/2$. We prove $\mathsf{P}(T)$ for all trees $T$ by structural induction.

**Base Case.** We show that $\mathsf{P}(\cdot)$ holds. By definition of leaves(.), $\mathsf{leaves}(\bullet) = 1$ and $\mathsf{size}(\bullet) = 1$. So, $\mathsf{leaves}(\bullet) = 1 \geq 1/2 + 1/2 = \mathsf{size}(\bullet)/2 + 1/2$.

**Induction Hypothesis:** Suppose $\mathsf{P}(L)$ and $\mathsf{P}(R)$ hold for some arbitrary trees $L$ and $R$.

**Induction Step:** We prove that $\mathsf{P}(\texttt{Tree}(\bullet, L, R))$ holds.

$$\begin{aligned}
\mathsf{leaves}(\texttt{Tree}(\bullet, L, R)) &= \mathsf{leaves}(L) + \mathsf{leaves}(R) && \text{[By Definition of leaves]} \\
&\geq (\mathsf{size}(L)/2 + 1/2) + (\mathsf{size}(R)/2 + 1/2) && \text{[By IH]} \\
&= (\mathsf{size}(L) + \mathsf{size}(R) + 1)/2 + 1/2 \\
&= \mathsf{size}(\texttt{Tree}(\bullet, L, R))/2 + 1/2 && \text{[By Definition of size]}
\end{aligned}$$

This proves $\mathsf{P}(\texttt{Tree}(\bullet, L, R))$.

Thus, the P($T$) holds for all trees $T$.

## 3. 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.)