The language \( L \) and functions \( R, A, \) and \( T \) defined below are the same as in Section 7.6.

\[
L \subseteq \{a, b, \cdot\}^+
\]

1. \( \cdot \in L \)
2a. \( u \in L \Rightarrow au \in L \)
2b. \( u \in L \Rightarrow bu \in L \)
3. n. e.

It is proved in the book that

- **Proposition 7.6.** \( A \) is associative; that is, for all \( u, v, w \in L \), \( A(u, A(v, w)) = A(A(u, v), w) \).

- **Proposition 7.9.** Assuming Proposition 7.8, below, \( R \) is self-cancelling; that is, for all \( u \in L \), \( R(R(u)) = u \).

Prove the following:

(a) **Proposition 7.7.** For all \( u \in L \), \( A(u, \cdot) = u \).

(b) **Proposition 7.8.** For all \( u, v \in L \), \( R(A(u, v)) = A(R(v), R(u)) \).

(c) **Proposition 7.11.** For all \( u, v \in L \), \( T(u, v) = A(R(u), v) \).

(d) **Proposition 7.10.** For all \( u \in L \), \( T(u, \cdot) = R(u) \).

(e) **Proposition 7.12.** For all \( u \in L \), \( T(T(u, \cdot), \cdot) = u \).
2. The program below is called Wensley’s algorithm for computing the quotient of real numbers $x$ and $y$ to within tolerance $t$. Use the Theorem on Loop Invariants to prove this program satisfies the post-condition $\{z \leq x/y < z + t\}$.

\[
\begin{align*}
\{0 \leq x < y \leq 1\} \\
\text{begin} \\
z := 0; \ d := 1; \ u := 0; \ v := \frac{1}{2}y; \\
\text{while } d > t \text{ do} \\
\{ \text{ INV } \equiv z \leq x/y < z + d \land u = zy \land v = \frac{1}{2}dy \} \\
\text{begin} \\
d := \frac{1}{2}d; \\
\text{if } u + v > x \text{ then skip} \\
\text{else begin } z := z + d; \ u := u + v; \ \text{end} \\
v := \frac{1}{2}v \\
\text{end} \\
\text{end} \\
\{z \leq x/y < z + t\}
\end{align*}
\]
3. Define $F : \mathbb{N} \to \mathbb{N}$ and $G : \mathbb{N}^2 \to \mathbb{N}$ as follows:

$$
F(0) = 1 \\
F(k + 1) = (k + 1) \times F(k)
$$

$$
G(0, m) = m \\
G(k + 1, m) = G(k, m \times (k + 1))
$$

(a) Prove by induction on $n \in \mathbb{N}$: For all $n, m \in \mathbb{N}$, $G(n, m) = m \times G(n, 1)$.

(b) Prove: For all $n \in \mathbb{N}$, $F(n) = G(n, 1)$. 

4. Performance estimation for recursive programs often involves recurrence relations like the one below. Let $a \in \mathbb{N}$. The function $T : \mathbb{N} \to \mathbb{N}$ is defined recursively by

\begin{align*}
T(0) &= a \\
T(k + 1) &= T(k) + k + 1
\end{align*}

We would like to find a closed form for $T$, that is, an algebraic expression that does not involve recursion.

Prove that for all $n \in \mathbb{N}$, $T(n) = a + \frac{n^2 + n}{2}$. 

Supplemental Problem. *H241 students should attempt this programming problem, but don’t spend more than two or three hours on it.*

In a programming language of your choice, write a program that takes no input and outputs its own source code.