Semantics of Programming Languages

The following grammar specifies the syntax of a simple imperative programming language. It is a fragment of L3.

Values: \( v ::= \text{skip} \mid n \mid x \mid \ell \)  
(n ranges over integers, \( x \) over variables, and \( \ell \) over locations)

Expressions: \( e ::= v \mid \text{let } x = e \text{ in } e' \mid v + v' \mid v := v' \mid !v \mid \text{ref}(v) \)

Types: \( T ::= \text{unit} \mid \text{int} \mid T \text{ref} \)

Stores: \( s \) finite partial functions from locations to values

Environments: \( \Gamma \) finite partial functions from locations and variables to types

Note that the grammar is very restrictive. For instance, the expression \((3 + 4) + 7\) is not allowed.

The language is typed according to the following standard rules.

\[
\begin{align*}
\Gamma \vdash \text{skip} : \text{unit} & \quad \Gamma \vdash n : \text{int} \quad \text{for } \text{an integer } n \\
\Gamma \vdash x : T & \quad \text{if } \Gamma(x) = T \\
\Gamma \vdash \ell : T \text{ref} & \quad \text{if } \Gamma(\ell) = T \text{ref} \\
\Gamma \vdash e : T & \quad \Gamma, x : T \vdash e' : T' \\
\Gamma \vdash \text{let } x = e \text{ in } e' : T' & \quad \Gamma \vdash v : T \text{int} \quad \Gamma \vdash v' : T \text{int} \\
\Gamma \vdash v + v' : T \text{int} \\
\Gamma \vdash v := v' : \text{unit} & \quad \Gamma \vdash v : T \text{ref} \quad \Gamma \vdash \text{ref}(v) : T \text{ref} \\
\Gamma \vdash !v : T & \quad \Gamma \vdash \text{ref}(v) : T \text{ref}
\end{align*}
\]

(a) Give a reasonable operational semantics for this language by defining a relation over configurations. [7 marks]

(b) Write down all the reduction steps of the following expression. You do not need to give their derivations.
\[
\text{let } x = \text{ref}(0) \text{ in let } y = !x \text{ in let } z = y + 3 \text{ in } x := z
\]
[3 marks]

(c) State and prove a Type Preservation Theorem for this language.

You may assume the following definition:

- a store \( s \) is well-typed for \( \Gamma \), written \( \Gamma \vdash s \),
  if for all locations \( \ell \in \text{dom}(s) \), there is a type \( T \) such that \( \Gamma(\ell) = T \text{ref} \) and \( \Gamma \vdash s(\ell) : T \)

You may also assume the following substitution lemma:

\[
\text{If } \Gamma \vdash v : T \text{ and } \Gamma, x : T \vdash e : T' \text{ with } x \not\in \text{dom}(\Gamma) \text{ then } \Gamma \vdash \{v/x\}e : T'
\]
[10 marks]