Semantics of Programming Languages

(a) State one potential advantage of programming languages that do not have a static type system. [1 mark]

(b) Consider the following language syntax:

\[
e ::= \text{skip} \mid b \mid n \mid \text{if } e_1 \text{ then } e_2 \text{ else } e_3 \mid \text{while } e_1 \text{ do } e_2 \mid \\
\quad \text{fn } x \Rightarrow e \mid e_1 \mid e_2 \mid x \mid \\
\quad \text{ref } e \mid e_1 ::= e_2 \mid !e \mid \ell
\]

where \( b \) ranges over the booleans \( \{\text{true}, \text{false}\} \), \( n \) ranges over the natural numbers, and \( \ell \) ranges over an infinite set of locations.

Design an operational semantics for this language that is well-defined and reasonable for arbitrary expressions (not just those that would be admitted by some static type system). Your semantics should:

1. involve clearly-specified notions of value \( v \) and store \( s \);
2. define a small-step reduction relation \( \langle e, s \rangle \longrightarrow \langle e', s' \rangle \);
3. be call-by-value; and
4. not be stuck for any configuration \( \langle e, s \rangle \) where \( e \) is not a value.

Explain any parts of your definition that differ from those in the definition of a conventional typed language, such as the typed languages in the course notes. [15 marks]

(c) State property 4 precisely. [1 mark]

(d) Give an outline proof of property 4, including the form of induction used and one non-trivial case. [3 marks]