Semantics

The abstract syntax of IMP commands is given by the following grammar:

\[
Com ::= \text{skip} | \text{Iexp} | Com ; Com |
\]
\[
\text{if } Bexp \text{ then } Com \text{ else } Com | \text{while } Bexp \text{ do } Com
\]

where \( \text{Iexp} \) and \( Bexp \) are syntactic categories of integer and boolean expressions and \( Pvar \) is a set of program variables. Let \( \text{States} = [Pvar \rightarrow \mathbb{Z}] \) and \( \text{Cont} \), the cpo of continuations, be \([\text{States} \rightarrow A_\bot] \), where \( A \) is an unspecified set of program answers. A continuation represents what is to be done with the state resulting from the execution of a command in order to return the result of the whole program.

The continuation semantics of IMP is defined by giving the meaning \( [C] \) of each \( C \in \text{Com} \) as a function which takes a continuation, representing what is to be done when the command has finished, together with a state in which the command is to be executed, and returns an answer:

\[
[\cdot] : \text{Com} \rightarrow (\text{Cont} \rightarrow (\text{States} \rightarrow A_\bot)).
\]

One clause of the definition of \( [C] \) is

\[
[\text{skip}] k S = k(S).
\]

Complete the definition of the continuation semantics of IMP commands (expressing their usual behaviour). You may assume that the functions

\[
[\cdot] : \text{Iexp} \rightarrow (\text{States} \rightarrow \mathbb{Z})
\]
\[
[\cdot] : \text{Bexp} \rightarrow (\text{States} \rightarrow \mathbb{B}) \text{ where } \mathbb{B} = \{\text{true}, \text{false}\}
\]

have already been defined. [9 marks]

Now add a new command \textbf{abort} to \text{Com} and a new error value \textbf{Err} to \( A \). The intended behaviour of \textbf{abort} is immediately to terminate the entire program, returning \textbf{Err}. Extend the continuation semantics of IMP by giving the definition of \( [\text{abort}] \). [2 marks]

Now add two further new command forms:

\[
Com ::= \ldots | \textbf{abort} | \textbf{exit} | \text{Com orelse Com}
\]

The intended behaviour of \((C_1 \text{ orelse } C_2)\) is that it executes exactly like \( C_1 \) unless \( C_1 \) hits an \textbf{exit} command, in which case further execution of \( C_1 \) is abandoned and \( C_2 \) is executed starting in the state at which \( C_1 \) encountered the \textbf{exit}. If \( C_1 \) does not encounter an \textbf{exit} then \( C_2 \) is ignored. An \textbf{exit} command without an enclosing \textbf{orelse} behaves like \textbf{abort}. 1
Give a revised continuation semantics to every command of IMP with \texttt{abort}, \texttt{exit} and \texttt{orelse} which reflects this behaviour and in which the denotation of $C \in \text{Com}$ is a function which takes two continuations and a state and returns an element of $A_{\bot}$:

$$[-] : \text{Com} \rightarrow (\text{Cont} \rightarrow (\text{Cont} \rightarrow (\text{States} \rightarrow A_{\bot}))).$$

Hint: The first continuation is the ordinary default continuation and the second is the continuation to be applied if the command \texttt{exits}. [9 marks]