Semantics

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

\[
Com ::= \text{skip} \mid P\text{var} ::= I\text{exp} \mid Com \mid Com ;\text{Com} \mid \text{if } B\text{exp then Com else Com} \mid \text{while } B\text{exp do Com}
\]

where \(I\text{exp}\) and \(B\text{exp}\) are syntactic categories of integer and boolean expressions and \(P\text{var}\) is a set of program variables. Let \(\text{States}\) be \([P\text{var} \to \mathbb{Z}]\) and \(\text{Cont}\), the cpo of continuations, be \([\text{States} \to A_{\bot}]\), 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:

\[
[-] : \text{Com} \to (\text{Cont} \to (\text{States} \to A_{\bot}))
\]

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

\[
[-] : I\text{exp} \to (\text{States} \to \mathbb{Z}) \\
[-] : B\text{exp} \to (\text{States} \to \mathbb{B}) \quad \text{where } \mathbb{B} = \{\text{true}, \text{false}\}
\]

have already been defined. \[9 \text{ marks}\]

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

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 \textit{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 \textit{exit}. If \(C_1\) does not encounter an \textit{exit} then \(C_2\) is ignored. An \textit{exit} command without an enclosing \textit{orelse} behaves like \textbf{abort}.
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 \textit{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{exit}s. [9 marks]