Semânticas de Linguagens de Programação
Exercícios 5
Semântica para procedimentos e variáveis II

1. Considera o seguinte programa **c** em **Proc**:

$$\begin{array}{l}\texttt{begin var } z \ := \ x; \\ \qquad x \ := \ y; \\ \qquad y \ := \ z \\ \texttt{end}\end{array}$$

   Usando a semântica natural com âmbito estático de **Proc** considera um ambiente env_P e um ambiente ev_0 onde $ev_0(x) = 1$, $ev_0(y) = 2$. Supõe ainda uma memória m_0, com $m_0(1) = 3$, $m_0(2) = 5$ e $m_0(next) = 3$. Constrói a árvore de derivação para execução de **c** com

$$env_P, ev_0 \vdash \langle \mathtt{c}, m_0 \rangle \ \longrightarrow \ m_1,$$

   indicando o valor de m_1.

2. Considera o seguinte programa **c** em **Proc**:

$$\begin{array}{l}\texttt{begin} \\ \texttt{proc fac is} \\ \texttt{begin var } z \ := \ x; \\ \texttt{if } \ x \ = \ 1 \texttt{ then skip} \\ \texttt{else } (x \ := x - 1; \texttt{call fac}; \ y \ := \ z * y) \\ \texttt{end}; \\ (y \ := 1; \texttt{call fac}) \\ \texttt{end}\end{array}$$

   Considera a semântica natural com âmbito estático para procedimentos e variáveis, sendo ep_0 um ambiente de procedimentos, $ev_0(x) = 1, ev_0(y) = 2$, $m_0(1) = 3$ e $m_0(next) = 3$. Calcula

$$ev_0, ep_0 \vdash \langle \mathbf{c}, m_0 \rangle \ \longrightarrow \ m_1,$$

   onde **c** é o comando (programa) dado.

3. Considera os seguintes comandos da linguagem **Proc**, $\mathtt{c_0}$:

$$\begin{array}{l}\texttt{begin var } w \ := \ 0; \texttt{var } z \ := \ 0; \\ \qquad \texttt{proc } inc \texttt{ is } w \ := \ w + 1; \\ \qquad \texttt{while } \neg \ w \ = \ y \texttt{ do} \\ \qquad\qquad (z \ := \ z + x; \\ \qquad\qquad \texttt{call } inc) \\ \qquad \texttt{end}\end{array}$$

   e $\mathbf{c_1}$:

$$w := 0; z := 0; \ \texttt{while } \neg w = y \texttt{ do } (z := z + x; w := w + 1)$$

(a) Seja $env_V(x) = 1$, $env_V(y) = 2$, $m_0(1) = 3$ e $m_0(2) = 2$, $m_0(next) = 3$ com $env_V : \mathbf{Var} \to \mathbf{Loc}$, $m_0 : \mathbf{Loc} \cup \{next\} \to \mathbb{Z}$ e $new(\ell) = \ell + 1$. Usando a semântica natural para a linguagem **Proc** com âmbito estático (para procedimentos e variáveis) determina m_1 tal que

$$env_V, env_P \vdash \langle \mathbf{c_0}, m_0 \rangle \longrightarrow m_1$$

construindo a árvore de derivação.

(b) Seja $s_0 : \mathbf{Var} \to \mathbb{Z}$ com $s_0(x) = 3$ e $s_0(y) = 2$. Usando a semântica natural para a linguagem **While** calcula $\langle \mathbf{c_1}, s_0 \rangle \longrightarrow s_1$ indicando a árvore de derivação.

(c) Qual é a relação entre s_1 e m_1?

(d) Como poderias concluir que $\mathbf{c_1}$ e $\mathbf{c_0}$ são equivalentes para a semântica natural?

4. Repete os exercícios 5. da folha prática 2 usando a semântica natural de Proc com âmbito estático para variáveis. Divide o comando em subcomandos atribuindo um nome a cada um deles.

5. Modifica a sintaxe da declaração de procedimentos para aceitarem dois parâmetros *call-by-value* (passagem de parâmetros por valor).

$$\begin{aligned} D_P &::= \texttt{proc}\ p(x_1, x_2)\ \texttt{is}\ c;\ D_P \mid \\ c &::= \ldots \mid \texttt{call}\ p(a_1, a_2) \end{aligned}$$

O ambiente dos procedimentos é agora

$$\mathbf{Env_P} = \mathbf{Pname} \hookrightarrow \mathbf{Com} \times \mathbf{Var} \times \mathbf{Var} \times \mathbf{Env_V} \times \mathbf{Env_P}$$

Modifica a semântica de **Proc** para estes constructores e em especial a semântica de `call`. Escreve um pequeno programa que ilustre o seu funcionamento.

6. Implementação em Haskell: implementa a linguagem **Proc** considerando agora semântica operacional natural com âmbitos estáticos $env_V, env_P \vdash \langle c, mem \rangle \longrightarrow mem'$, seguindo as ideias acima e define uma função

```
snProcE :: EnvV -> EnvP -> Com -> Mem -> Mem
```

sendo que Mem deve corresponder às funções de $\mathbf{Loc} \cup \{next\} \to \mathbb{Z}$.

Semântica natural para Proc com âmbitos estáticos

Sendo $s = mem \circ env_V$, as regras $env_V, env_P \vdash \langle c, mem\rangle \longrightarrow mem'$ são:

$$\texttt{att}_{sn}\ \ env_V, env_P \vdash \langle x := a, mem\rangle \longrightarrow mem[\ell \mapsto v]$$
onde $\ell = env_V(x), v = \mathcal{A}[\![a]\!]\, s$

$$\texttt{skip}_{sn}\ \ env_V, env_P \vdash \langle \texttt{skip}, mem\rangle \longrightarrow mem$$

$$\texttt{comp}_{sn}\ \ \frac{env_V, env_P \vdash \langle c_1, mem\rangle \longrightarrow mem' \quad env_V, env_P \vdash \langle c_2, mem'\rangle \longrightarrow mem''}{env_V, env_P \vdash \langle c_1; c_2, mem\rangle \longrightarrow mem''}$$

$$\texttt{if}^{v}{}_{sn}\ \ \frac{env_V, env_P \vdash \langle c_1, mem\rangle \longrightarrow mem'}{env_V, env_P \vdash \langle \texttt{if}\ b\ \texttt{then}\ c_1\ \texttt{else}\ c_2, mem\rangle \longrightarrow mem'}$$
se $\mathcal{B}[\![b]\!]\, s = \mathsf{true}$

$$\texttt{if}^{f}{}_{sn}\ \ \frac{env_V, env_P \vdash \langle c_2, mem\rangle \longrightarrow mem'}{env_P \vdash \langle \texttt{if}\ b\ \texttt{then}\ c_1\ \texttt{else}\ c_2, mem\rangle \longrightarrow mem'}$$
se $\mathcal{B}[\![b]\!]\, s = \mathsf{false}$

$$\texttt{while}^{v}{}_{sn}\ \ \frac{env_V, env_P \vdash \langle c, mem\rangle \longrightarrow mem' \quad env_V, env_P \vdash \langle \texttt{while}\ b\ \texttt{do}\ c, mem'\rangle \longrightarrow mem''}{env_V, env_P \vdash \langle \texttt{while}\ b\ \texttt{do}\ c, mem\rangle \longrightarrow mem''}$$
se $\mathcal{B}[\![b]\!]\, s = \mathsf{true}$

$$\texttt{while}^{f}{}_{sn}\ \ env_V, env_P \vdash \langle \texttt{while}\ b\ \texttt{do}\ c, mem\rangle \longrightarrow mem$$
se $\mathcal{B}[\![b]\!]\, s = \mathsf{false}$

$$\texttt{block}_{sn}\ \ \frac{\langle D_V, env_V, mem\rangle \longrightarrow_D (env'_V, mem') \quad env'_V, env'_P \vdash \langle c, mem'\rangle \longrightarrow mem''}{env_V, env_P \vdash \langle \texttt{begin}\ D_V\ D_P\ c\ \texttt{end}, mem\rangle \longrightarrow mem''}$$
onde $env'_P = upd_P(D_P, env'_V, env_P)$

$$\texttt{call}_{sn}\ \ \frac{env'_V, env'_P \vdash \langle c, mem\rangle \longrightarrow mem'}{env_V, env_P \vdash \langle \texttt{call}\ p, mem\rangle \longrightarrow mem'}$$
onde $env_P(p) = (c, env'_V, env'_P)$

$$\texttt{call}^{rec}{}_{sn}\ \ \frac{env'_V, env'_P[p \mapsto (c, env'_V, env'_P)] \vdash \langle c, mem\rangle \longrightarrow mem'}{env_V, env_P \vdash \langle \texttt{call}\ p, mem\rangle \longrightarrow mem'}$$
onde $env_P(p) = (c, env'_V, env'_P)$

$$\texttt{var}_{sn}\ \ \frac{\langle D_V, env_V[x \mapsto \ell], mem[\ell \mapsto v][next \mapsto m]\rangle \longrightarrow_D (env'_V, mem')}{\langle \texttt{var}\ x := a;\ D_V, env_V, mem\rangle \longrightarrow_D (env'_V, mem')}$$
onde $v = \mathcal{A}[\![a]\!]\, mem \circ env_V, \ell = mem(next), m = new(\ell)$

$$\texttt{empty}_{sn}\ \ \langle \varepsilon, env_V, mem\rangle \longrightarrow_D (env_V, mem)$$

$$upd_P(\texttt{proc}\ p\ \texttt{is}\ c;\ D_P, env_V, env_P) = upd_P(D_P, env_V, env_P[p \mapsto (c, env_V, env_P)])$$
$$upd_P(\varepsilon, env_V, env_P) = env_P$$