Exercise 1

Let \( \pi \) be the following program seen in the lecture course:

\begin{verbatim}
begin var \ x := 2;
 proc \ p \ is \ x := 0;
 proc \ q \ is \ begin \ var \ x := 1; \ proc \ p \ is call \ p; \ call \ p; \ end;
 call \ p;
 end
\end{verbatim}

Compute the semantics of this program according to the following variants:

1. with static links for procedures, dynamic links for variables, and the recursive call rule,
2. with static links for procedures, dynamic links for variables and the non-recursive call rule.

Exercise 2

Let \( \pi \) be the following program seen in the lecture course:

\begin{verbatim}
begin var \ x := 0;
 proc \ p \ is \ x := x * 2;
 proc \ q \ is \ call \ p;
 begin
 \ var \ x := 5;
 \ proc \ p \ is \ x := x + 1;
 \ call \ q; \ y := x;
 end;
 end
\end{verbatim}

Compute the semantics of this program according to the following three variants:

1. Dynamic links for procedures and variables.
2. Static links for procedures and dynamic link for variables.
3. Static links for procedures and variables.

Exercise 3

We add the following statement to the \textbf{While} language with blocks and procedures:

\begin{verbatim}
Stm ::= p := S.
\end{verbatim}

Give a semantics to this language. Your semantics should be conservative wrt. the semantics of the \textbf{While} language.

Exercise 4
Complete the semantics of **While** with blocks and procedures with static links for variables and procedures.

**Exercise 5**

We modify the syntax of procedures to allow parameters:

\[
S \in \text{Stm} \\
S ::= x := a | \text{skip} | S_1; S_2 | \\
     \text{if } b \text{ then } S_1 \text{ else } S_2 \\
     \text{while } b \text{ do } S \text{ od} | \text{begin } D_V \text{ } D_P; \text{end} | \text{call } p(a_1, a_2) \\
D_V ::= \text{var } x := a; D_V | \epsilon \\
D_P ::= \text{proc } p(x_1, x_2) \text{ is } S; D_P | \epsilon
\]

We are interested in the semantics with static link for procedures and dynamic link for variables.

- Modify the semantics of procedure declaration and call in order to obtain a semantics with call-by-value.
- Same thing with call-by-reference.