UFR-IMAG
Université Joseph Fourier

Programming Language and Compiler Design,
2010/2011

Marion Daubignard
Yassine Lakhnech
Laurent Mounier

Exercise 1

Let π be the following program seen in the lecture course:

```
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
```

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 π be the following program seen in the lecture course:

```
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
```

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 **While** language with blocks and procedures:

Stm ::= $p := S$.

Give a semantics to this language. Your semantics should be conservative wrt. the semantics of the **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:

$$
\begin{array}{lcl}
S & \in & \mathbf{Stm} \\
S & ::= & x := a \mid \mathsf{skip} \mid S_1; S_2 \mid \\
 & & \mathsf{if}\ b\ \mathsf{then}\ S_1\ \mathsf{else}\ S_2 \\
 & & \mathsf{while}\ b\ \ \mathsf{do}\ S\ \mathsf{od}\ \mid \mathsf{begin}\ D_V\ D_P;\ S\ \mathsf{end} \mid \mathsf{call}\ p(a_1, a_2) \\
D_V & ::= & \mathsf{var}\ \ x := a;\ D_V \mid \epsilon \\
D_P & ::= & \mathsf{proc}\ p(x_1, x_2)\ \mathsf{is}\ S;\ D_P \mid \epsilon
\end{array}
$$

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.