CAP - Exercises: static semantics (chapter 4: typing)

Laure Gonnord

Oct. 2016

Abstract syntax

Recall the abstract syntax of the course for expressions :

$$
\begin{array}{rcll}
e & ::= & c & \textit{constant} \\
 & | & x & \textit{variable} \\
 & | & e + e & \textit{addition} \\
 & | & e \times e & \textit{multiplication} \\
 & | & \ldots &
\end{array}
$$

and the mini-while language :

$$
\begin{array}{rcll}
S(Smt) & ::= & x := expr & \text{assign} \\
 & | & skip & \text{do nothing} \\
 & | & S_1; S_2 & \text{sequence} \\
 & | & \texttt{if}\ b\ \texttt{then}\ S_1\ \texttt{else}\ S_2 & \text{test} \\
 & | & \texttt{while}\ b\ \texttt{do}\ S\ \texttt{done} & \text{loop}
\end{array}
$$

We expand the language with variable declarations :

$$
D(decl) \quad ::= \quad \texttt{var}\ x : t \quad \text{type declaration}
$$

We recall that environnements associate a type to variables (Γ). Here, the environnement is constructed by the following rules :

Declarations

$$
\frac{}{\texttt{var}\ x : t \rightarrow_d [x \mapsto t]}
$$

$$
\frac{D_1 \rightarrow_d \Gamma_1 \quad D_2 \rightarrow_d \Gamma_2 \quad Dom(\Gamma_1) \cap Dom(\Gamma_2) = \emptyset}{D1; D_2 \rightarrow_d \Gamma_1 \cup \Gamma_2}
$$

Expressions Like in the course, for instance :

$$
\frac{\Gamma \vdash e_1 : \texttt{int} \quad \Gamma \vdash e_2 : \texttt{int}}{\Gamma \vdash e_1 + e_2 : \texttt{int}}
$$

Commands Like in the course, for instance :

$$
\frac{\Gamma \vdash b : \texttt{boolean} \quad \Gamma \vdash S : \texttt{void}}{\Gamma \vdash \texttt{while}\ b\ \texttt{do}\ S\ \texttt{done} : \texttt{void}}
$$

And a program

$$\frac{D \to \Gamma \quad \Gamma \vdash C : \texttt{void}}{\Gamma \vdash DC : \texttt{void}}$$

Exercise ▶ Well typed

Type the program :

```
var x1 : integer ; var x2 : integer ; var x3 : integer
x1 := 3 ;
while (not x3) do
x1 := x2 + 1 ;
x3 := x3 and true
done
```

Exercise ▶ Expand expressions

Complete the abstract syntax and the static semantics (typing) of expressions with the new construction `e1? e2: e3` : if e_1 is true then the expression has value e_2 else e_3.

Exercise ▶ Expand the statements

Complete the abstract syntax and the static semantics (typing) of statements with an extended for :

```
for i in e1 .. e2 S
```

2 cases :

- The instruction declares the i variable (like in Ada)
- i should be declared before.