Programming Languages

Spring 2004 Qualifiers

November 24, 2003

Question 1

Consider a simple **while** language with the following syntax:

Program	→	*Statement*.
Statement	→	*Statement*; *Statement* |
		Identifier := *Expression* |
		if *Expression* **then** *Statement* **else** *Statement* |
		while *Expression* **do** *Statement*
Expression	→	*Identifier* |
		Numeral |
		Expression + *Expression* |
		Expression ∗ *Expression* |
		(*Expression*) |
		Expression > *Expression*

Let us assume that now we want to extend the syntax with the notion of *macro*, with the following changes:

Program	→	*Macros Statement*.
Macros	→	ϵ |
		Macro Macros
Macro	→	**define** *Identifier Statement*;

and we add one extra case to the statements:

Statement	→	**call** *Identifier*

The following is an example:

```
define MultiplyY  y:=y*y ;
define DecrementX  x:=x-1 ;

y:=1;
while (x > 0) do
   MultiplyY ;
   DecrementX
```

Extend Hoare's partial correctness axiomatization by adding a Rule to handle the **call** statement.

Question 2

Let us assume that we are interested in developing a language for programming agents. The agent is moving in a 2-d chessboard and it is trying to reach a target. The agent is supplied with one sensor, which returns the values:

$$sensor = \begin{cases} 1 & \textit{the target is same row on the right} \\ -1 & \textit{the target is same row on the left} \\ 0 & \textit{target reached} \\ 2 & \textit{target is in the same column up} \\ -2 & \textit{target is in the same column down} \\ 3 & \textit{not localized} \end{cases}$$

The behavior of the agent is reactive: it reads the sensor and immediately decides what to do next. The agent is programmed to repeatedly perform a cycle: read sensor, select move, perform move. You can program the behavior of the agent by providing the rules that the agent will apply to decide the move based on the sensor. The syntax for a program is

Program	$\rightarrow$	**sense-plan-act cycle:** *Sensor Rules*.
Sensor	$\rightarrow$	**Get Sensor** *Identifier*
Rules	$\rightarrow$	*Rule* |
		Rule Rules
Rule	$\rightarrow$	*Identifier* = *Numeral* **causes** *Action*
Action	$\rightarrow$	**go left** | **go right** | **go up**
		go down | **go random** | **stop**

For example, a sample program:

```
sense-plan-act cycle:
    Get Sensor  x
    x = 1 causes go right
    x = 0 causes stop
    x = -1 causes go left
```

Provide the denotational semantics for this language; the arity of the top-level valuation function should be:

$$\mathcal{P} : \textit{Program} \rightarrow \textit{Position}$$

where the semantic algebra *Position* has a domain $Int \times Int$ (Int is the set of integers). Consider the following assumptions:

- when started, the agent is in the location $(0, 0)$; the chessboard is infinite in each direction;
- you can assume the existence of a function

  $$\textit{locate} : \textit{Position} \rightarrow \{0, 1, -1, 2, -2, 3\}$$

  which locates the target based on the current position.

Make sure to provide all the necessary semantic algebras as well as valuation functions.