COMP 317: Semantics of Programming Languages

Problem Sheet 3

1. Extend the syntax and semantics of the programming language with for-loops.

2. Some languages (such as the OO language Eiffel) have "assertions": these are commands that assert that some condition holds. If the condition does hold, then the assertion has no effect, and computation proceeds to the next following command; if the condition does not hold, then the program crashes (and for the sake of this exercise, this can be considered to have the same effect as a non-terminating loop). The conditions that are asserted are just Boolean expressions (<BooleanExpression>s), and an assertion could be written as: `assert` *T* for some Boolean expression *T*. Add assertions to our language by specifying their syntax in BNF, and by giving them a denotational semantics.

3. We want to extend the language with "case conditionals" of the form

```
case E of
  N1 : P1 ;;
  N2 : P2 ;;
    ...   ;;
  Nm : Pm
endcase
```

   where *E* is an expression, each *Ni* is an integer, and each *Pi* is a program. This program is executed by first evaluating the expression *E* to obtain an integer *N*; if the first occurrence of *N* in the list *N1, ..., Nm* is *Ni* (we allow that the list *N1, ..., Nm* may contain duplicates), then program *Pi* is executed; if *N* doesn't occur in the list *N1, ..., Nm* then the program immediately terminates (i.e., is equivalent to skip). For example, the program

```
case  'x + 1  of
  0 : 'z := 5; ;;
  3 : 'z := 6; ;;
  4 : 'y := 0;
endcase
```

   will set `'z` to 5 if `'x` has the value -1; it will set `'z` to 6 if `'x` has the value 2; it will set `'y` to 0 if `'x` has the value 3; and it will have no effect if `'x` has any other value.

   1. Give a BNF definition of a syntactic category <CaseList> for the list of cases, where the list either consists of a single case, of the form *N : P*, with *N* an integer and *P* a program, or is of the form *N : P* `;;` *CL*, where *CL* is a CaseList. For example,

```
0 : 'z := 5;  ;;  3 : 'z := 6;  ;;  4 : 'y := 0;
```

      is a <CaseList>.
   2. Extend the BNF syntax of the programming language with a clause stating that Programs (<Pgm>) may also consist of case conditionals of the form

```
case E of CL endcase
```

      where *E* is an expression and *CL* a CaseList.
   3. Define a semantic function for CaseLists

```
[[ CL ]] : Int State -> State
```

such that for a CaseList *CL*, integer *N* and State *S*, [[CL]](N,S) gives the State that results from choosing the first program in CL with label *N* and running it in state *S*. For example, it should follow from your definition that

```
[[ 0 : 'z := 5; ;;  3 : 'z := 6; ;;  4 : 'y := 0; ]](3, S)
```

will return the state that results from running the program `'z := 6;` in the State *S*.

4. Extend the program-denotational semantics to give a semantics for case conditionals; i.e., define

   ```
   [[ case E of CL endcase ]]
   ```

   where *E* is an expression and *CL* a CaseList.
5. Use your answers to calculate the semantics of the following program:

   ```
   'x := 2 ;
   case  'x + 1  of
     0 : 'z := 5; ;;
     3 : 'z := 6; ;;
     4 : 'y := 0;
   endcase
   ```

Grant Malcolm