Homework #6

Q1. Program the semantics of the language whose definition is given below (the language is similar to the language in HW 4, except for two productions added for Expressions.

Prolog code for a subset of a language can be found in this paper: Horn Logic Denotations and Their Applications, appears in Future Directions in Logic Programming, Springer Verlag, 1998.

P in Program
K in Block
D in Declaration
C in Command
E in Arithmetic Expression
B in Boolean Expression
I in Identifier
N in Number

P ::= K.
K ::= begin D; C end
D ::= D ; D | const I = N | var I
C ::= C ; C | I := E | if B then C else C endif | while B do C endwhile | K
B ::= true | false | E = E | not B
E ::= E + E | E - E | E * E | E / E | (E) | I:= E | I | N
I ::= x | y | z | u | v
N ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

Assume that all programs written in this language take 2 inputs that are found in variables x and y before the program begins execution, and produce one output found in variable z after the program execution is over.

Notice that both arithmetic and boolean expressions have side-effects due to the addition of the rule \( E ::= I := E \).
Notes for Q1:

(1) Please make sure that the program is your own work (and do not share your code).

(2) Use program/3 to create your parse tree and program_eval/4 to run it. For example, your program should accept the following query:

```
?- program(P,[begin, var, z, ; , var, x, ;, z, :=, x, end, .],[]),
    write(P), program_eval(P,2,3,Z).
```

and return $Z = 2$.

(3) Test your prolog program with the following 8 cases of simple programs and attach your result for each case. You may use the initial values for $x$ and $y$ to be $x=2$, $y=3$, thus to output $z$.

1. `begin var z; var x; z:=x end`.
2. `begin var x; var y; var z; z:=x+y end`.
3. `begin var x; var y; var z; z:=(z:=x+2)+y end`.
4. `begin var x; var y; var z;
   if x=y then z:=1 else z:=0 endif end`.
5. `begin var x; var y; var z;
   if x = 0 then z:=x else z:=y endif end`.
6. `begin var x; var y; var z;
   if not(x=y) then z:=x else z:=y endif end`.
7. `begin var x; var z;
   z:=0; while not x=0 do z := z+1; x:=x-1 endwhile end`.
8. `begin var x; var y; var z;
   z:=1; w:=x; while not w = 0 do z :=z*y; w:=w-1 endwhile end`.

(4) Submit your prolog code and output through eLearning. (Send only your complete prolog program code. If TA cannot load or run your program, then your program is not working! Put some comments if TA needs any special instruction to load and run).
Q2. Use Mixtus to generate code for a sample program. Remember you'll have to comment out the code for update and access routines before partial evaluation.

To run Mixtus type 'mixtus' on apache, jupiter, cs1, cs2, .. (any of the Sun machines).

To load a program file code.pl in mixuts type 'pconsult('code.pl').' at the mixtus prompt.

To partial evaluate a goal under mixtus type the goal main(...), type 'pe(main(...))' at the mixtus prompt, where main/3 is:

\[
\text{main}(\text{ValX}, \text{ValY}, \text{A}) \leftarrow \text{program}(\text{P}, [\text{begin}, \text{var}, \text{x}, ;, \text{var}, \text{y}, ;, \text{var}, \text{z}, ;, \text{var}, \text{w}, ;, \text{z}, :=, 1, ;, \text{w}, :=, \text{x}, ;, \text{while}, \text{w}, >, 0, \text{do}, \text{z}, :=, \text{z}, *, \text{y}, ;, \text{w}, :=, \text{w}, -, 1, \text{endwhile}, \text{end}, .],[]), \text{write}(\text{P}), \text{prog_eval}(\text{P}, \text{ValX}, \text{ValY}, \text{A}).
\]