Written Assignment 5  
Due November 16, 2015 at 12:30pm

You may discuss this assignment with other students and work on the problems together. However, your write-up should be your own individual work and you must acknowledge the students you work with. Written assignments must be turned in at the start of lecture on the indicated due date. If you choose to use any late days on this assignment, you must email your assignment to the TA.

1. Consider the following array language:

\[ P \rightarrow v_1 = v_2[v_3] \mid v_1[v_2] = v_3 \mid v = \text{int} \mid P_1; P_2 \mid \epsilon \]

Here, array read and write are standard and \( c = \text{int} \) assigns variable \( v \) to an integer constant. You can assume that arrays are declared upon use, all arrays contain only integers and can only be indexed by integers. Also assume that arrays are of unlimited size.

(a) Give three distinct non-trivial, valid programs in this language  
(b) Write large-step operational semantics for this array language  
(c) List all run-time errors possible in this language

2. Assume that we change the language IMP1 from lecture 15 so that variables must be declared before use with a \texttt{declare id} statement.

(a) List all run-time errors now possible in this modified version of IMP1  
(b) Design a type system that prevents all run-time errors in this version of IMP1  
(c) Show a derivation for your type system for one non-trivial valid program

3. After a long night partying, you think of a new construct you want to add to your favorite imperative programming language, the \texttt{COMEFROM} keyword. You can think of its meaning as \texttt{GOTO} in reverse. For your reference, here is the same program with \texttt{GOTO} and \texttt{COMEFROM}

```plaintext
int i = 0;
int n = 1;
again:
i++;
n=n*n;
if(i <= 3) goto again:
return n;

int i = 0;
int n = 1;
if(i <=3) comefrom again:
i++;
n=n*n;
again:
return n;
```

Fall 2015/2016
Specifically, COMEFROM \texttt{l} will change control to the current location from where the label \texttt{l} is located.

(a) Do you think that this construct will lead to readable code? Explain your answer.
(b) Write an small example program using COMEFROM that either illustrates how hard to read code becomes or how lucid and clear code can be written with COMEFROM

4. After learning about imperative languages, you decide to add some imperative features to L. Specifically, you plan to add a \texttt{while} loop to L. You have the following syntax in mind:

\texttt{while(e1) do e2 od}

The loop body will execute as long as \texttt{e1} evaluates to a non-zero integer and the entire loop will evaluate to the value of \texttt{e2} in the last iteration and 0 if the loop never executes.

(a) Give large-step operational semantics for this loop in L
(b) Is this a sensible construct to add to L? Specifically, how many times must every loop execute?