Homework 8

Due Friday, 11/5/2010

Please turn in your homework solutions online at `http://www.dci.pomona.edu/tools-bin/cs131upload.php` before the beginning of class on the due date.

Several questions on this homework involve ML program. However, they behave identically to equivalent Haskell programs. If you have any problem understanding them, please let me know. There are also some Ada programs, where Ada is a procedural language designed about 1980, but again the syntax should be pretty understandable.

1. (20 points) **Activation Records for Inline Blocks**

   Please do problem 7.1 from Mitchell, page 191.

2. (10 points) **Time and Space Requirements**

   Please do problem 7.3 from Mitchell, page 193.

3. (15 points) **Ada Parameter Modes**

   The Ada programming language permits parameters to be labeled as `in`, `out`, or `in out`, as in the following procedure definitions, where `T` is some type:

   ```
   procedure test1(in x: T) is begin ... end
   procedure test2(out x: T) is begin ... end
   procedure test3(in out x: T) is begin ... end
   ```

   The modifiers, or modes, have the following meaning:

   - `in`: The value of the parameter `x` cannot be changed inside the procedure. If we call `test1(y)`, the value of `y` is the same before and after the call..
   - `out`: The parameter `x` can be written to, but it cannot be read. If we call `test2(y)`, the value of `y` after the call is the last value written to `x` in the procedure.
   - `in out`: The parameter `x` can be both read and written, and the value of `y` after a call to `test3(y)` is the last value written to `x` in the procedure.

   The language definition does not specify how each mode should be implemented, and the compiler may use any appropriate parameter passing mechanism to implement them.

   (a) Which parameter passing mechanism could be used to implement `test1`, `test2`, and `test3`? The choices are pass-by-reference, pass-by-value, and pass-by-value-result (as described in problem 7.6). If more than one is possible, describe the advantages/disadvantages of each.

   (b) Consider the following procedure that takes two parameters. Does the following program print the same value for all strategies you outlined for `in out` parameters above?

```
procedure incTwo(in out x:integer, in out y:integer) is
  begin
    x := x + 1;
    y := y + 1;
  end

procedure main() is
  w : integer = 3;
  begin
    incTwo(w,w);
    print w;
  end
```

(c) Discuss the advantages and disadvantages of permitting the compiler such flexibility in how it implements parameter modes.

4. (10 points) **Activation records**

   Draw the stack of activation records for the following Ada program (a) after the first call to procedure b; (b) after the second call to procedure b. Show the static (access) and dynamic (control) links for each activation record. (c) Indicate how x is found in procedure c.

```
procedure env is
   integer x = 12;
   procedure a is
      integer y = 2;
      procedure b(z) is
         procedure c is
         begin
            b(x);
         end c;
      begin
         c;
      end b
   begin
      b(x+y);
   end a;
begin
   a;
end env
```

5. (15 points) **Function Calls and Memory Management**

   Please do problem 7.12 from Mitchell, page 198.

6. (15 points) **Function Returns and Memory Management**

   Please do problem 7.13 from Mitchell, page 199.