[Due Date: May, 25, 5pm]

CS 381, Spring 2020 — Homework 5 (Runtime Stack, Scoping, Parameter Passing)

Please note:
Submit ***one*** solution per group (each group can have up to 5 members) through Canvas.
Late submissions will ***not*** be accepted. Do ***not*** send solutions by email.

Exercise 1. Runtime Stack

Consider the following block. Assume static scoping and call-by-value parameter passing.

```
{ int x;
  int y;
  y := 1;
  { int f(int x) {
      if x=0 then {
         y := 1 }
      else  {
         y := f(x-1)*y+1 };
      return y;
    };
    x := f(2);
  };
}
```

Illustrate the computations that take place during the evaluation of this block, that is, draw a sequence of pictures each showing the complete runtime stack with all activation records after each statement or function call.

Note. Do ***not*** use the alternative model of "temporary stack evaluation" that was briefly illustrated on slides 20 and 25 to explain the implementation given in `FunStatScope.hs` and `FunRec.hs`. Rather use one stack onto which a new activation record is pushed on each recursive function call.

Exercise 2. Static and Dynamic Scope

Consider the following block. Assume call-by-value parameter passing.

```
{ int x;
  int y;
  int z;
  x := 3;
  y := 7;
  { int f(int y) { return x*y };
    int y;
    y := 11;
    { int g(int x) { return f(y) };
      { int y;
        y := 13;
        z := g(2);
      };
    };
  };
}
```

(a) Which value will be assigned to `z` in line 12 under static scoping?

(b) Which value will be assigned to `z` in line 12 under dynamic scoping?

It might be instructive to draw the runtime stack for different times of the execution, but it is not strictly required.

Exercise 3. Parameter Passing

Consider the following block. Assume dynamic scoping.

```
{ int y;
  int z;
  y := 7;
  { int f(int a) {
      y := a+1;
      return (y+a)
    };
    int g(int x) {
      y := f(x+1)+1;
      z := f(x-y+3);
      return (z+1)
    }
    z := g(y*2);
  };
}
```

What are the values of `y` and `z` at the end of the above block under the assumption that both parameters `a` and `x` are passed:

(a) Call-by-Name

(b) Call-by Need

It might be instructive to draw the runtime stack for different times of the execution, but it is not strictly required.