CS224 – Lab 2

Purpose: We have seen different ways that variables can be bound to memory locations in activation records: static activation records, a stack of activation records, activation records with nesting links, and activation records that can't be deallocated from a stack after execution ends. The more sophisticated the language, the more sophisticated a method for binding is required.

Knowledge: This lab will help you become familiar with the following content knowledge:

- The limits of static activation records
- The limits of activation records on a stack

Task: Follow the steps in this lab carefully to complete the assignments.

Assignment 1:
Draw the stack of activation records for the expression `f 3` at the point when the recursive call `f 1` has just been called. Explain why static activation records would fail for this function call.

```
f x = if x <= 0 then 5 else x + f (x-1)
```

Criteria for Success: You have a diagram of three activation records. The diagram box for an activation record should contain the names and values of all the parameters and locals (in this case there are no locals). Between the records you need to draw that the dynamic links that indicate who called that activation record. You also need an explanation for why a stack is required for this problem.

Assignment 2:
Draw the activation records for the expression `f 3` at the point when `x+z` inside `h` is being evaluated. Explain why nesting links are required and what value is returned.

```
f x = let h z = x + z
          g x = h (x+1) in
            g (x+1)
```

Criteria for Success: You have a diagram of three activation records that includes both dynamic and nesting links that are clearly labeled. The nesting link indicates in which activation record the function was defined. You also need an explanation for why and how the nesting link is used.

Assignment 3:
Draw the activation records for the expression `f 3` at the point when `g 5` is just about to return. Explain why the activation record for `g` can not be deallocated at that point.

```
f x = let g y = (\z-> z+y) in
          (g 5)x
```

Criteria for Success: You have a diagram of two activation records and a function object that includes the dynamic and nesting links and an explanation for why `g`'s activation record can not be deallocated immediately after `g`'s execution is complete.

Submit your answers in Canvas for grading or turn in a paper assignment if you prefer.