1. Concepts of Programming Languages, Chapter 6

Problems

1. Function Calls and Memory Management

This question asks about memory management in the evaluation of the following statically-scoped ML expression.

```ml
let val x = 2
  fun f(y) = if y>x then 5 else 0
  fun g(h) = let val x = 1 in h(3) end
in
  let val x = 0 in g(f) end
end;
```

(a) Fill in the missing information in the following depiction of the run-time stack after the call to `h(3)` inside the body of `g`. Remember that function values are represented by closures, and that a closure is a pair consisting of an environment (pointer to an activation record) and compiled code.

In this drawing, a bullet (●) indicates that a pointer should be drawn from this slot to the appropriate closure or compiled code. Since the pointers to activation records cross and could become difficult to read, each activation record is numbered at the far left. In each activation record, place the number of the activation record of the statically enclosing scope in the slot labeled “static link.” The first two are done for you. Also use activation record numbers for the environment pointer part of each closure pair.

<table>
<thead>
<tr>
<th>Activation Records</th>
<th>Closures</th>
<th>Compiled Code</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Activation Record Diagram" /></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b) What is the value of this expression? Why?
Consider the following code:

```ml
fun g(f) = 
  let val x : int ref = ref (f(1)) 
  in 
    fn(y) => (x := (!x) * y; !x) 
  end;
val x = 1;
fun s(y) = y + x;
val h = g(s);
val z = h(3);
```

(a) What are the types of `g`, `h`, and `z`?
(b) What is the value of `z`?
(c) Draw the run-time structures that result from the execution of this code, by adding to the following diagram. (You may or may not have to draw extra closure pairs, etc.)