CS3723 Homework #9, Due: 9am, 11/3/10

Procedure Calls and Activation Records

1. For each of the following activation record fields describe what is stored in that field: (a) control link (b) access link (c) return address (d) return-result address (e) actual parameters (f) local variables (g) temporary storage.

2. (a) For each of the following programming language features, list the key implementation constructs that are used to implement them: (i) blocks (ii) nested blocks and recursive functions and procedures (iii) static scoping in recursive functions and procedures (iv) functions as values (v) functions values returned from nested scope.
(b) Why are closures and access links needed in ML but not in C?

3. Do exercise 7.1 in Mitchell (p. 191).

4. Explain what the difference between pass-by-name (p. 96), pass-by-value, and pass-by-reference parameter passing.


7. Suppose that activation records (AR) for a C implementation are stack allocated with the form:

```
control link
return-result address
return address
parameter 1
...
parameter n
local variable 1
...
local variable n
temporary 1
...
temporary n
```

Draw the activation records that would be on the stack just after the execution of line 6 during the call to `fac()` with the parameter 2 in the C program shown to the left below. Use the line numbers for return addresses. Draw directed arcs for the control links. Clearly label the values of local variables and parameters. Label each activation record with the procedure name. Also show which activation record is pointed to by the environmental pointer. The stack should grow towards the bottom of the page. To get you started, the activation record for main is shown to the right of the program code.

```
1  int fac(int n)
2  {
3      if (n <= 1) {
4          return 1;
5      } else {
6          int x = -2;
7          x = fac(n - 1);
8          return n * x;
9      }
10  }
11
12 main()
13 {
14    int val = -1;
15    val = fac(3);
16    printf("%d\n", n);
17  }
```

```
  control link
  return-result address
  return address
  main()
```
8. Consider the execution of the following ML function:

```ml
fun main() = 
  let 
    fun mk_counter() = 
      let val counter = ref 0
        fun inc() = counter := !counter + 1;
        fun curr() = !counter;
      in 
        (inc, curr)
      end;
    val (inc, curr) = mk_counter();
  in 
    inc(); inc(); curr()
  end;

Diagram the activation records and closures for `main`, `make_counter`, and `curr` as they would exist during an execution of `main` just prior to `curr` returning. Pattern your answer after the template shown below.

<table>
<thead>
<tr>
<th>Activation Records</th>
<th>Heap Data/Closures</th>
<th>Compiled Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) main</td>
<td>access link</td>
<td>(0)</td>
</tr>
<tr>
<td></td>
<td>control link</td>
<td></td>
</tr>
<tr>
<td>mk_counter</td>
<td>(0)</td>
<td></td>
</tr>
<tr>
<td>inc</td>
<td>(0)</td>
<td></td>
</tr>
<tr>
<td>curr</td>
<td>(0)</td>
<td></td>
</tr>
<tr>
<td>(2) mk_counter</td>
<td>access link</td>
<td>( )</td>
</tr>
<tr>
<td></td>
<td>control link</td>
<td></td>
</tr>
<tr>
<td></td>
<td>( )</td>
<td></td>
</tr>
<tr>
<td>counter</td>
<td>( )</td>
<td></td>
</tr>
<tr>
<td>inc</td>
<td>( )</td>
<td></td>
</tr>
<tr>
<td>curr</td>
<td>( )</td>
<td></td>
</tr>
<tr>
<td>(3) curr</td>
<td>access link</td>
<td>( )</td>
</tr>
<tr>
<td></td>
<td>control link</td>
<td></td>
</tr>
<tr>
<td></td>
<td>( )</td>
<td></td>
</tr>
</tbody>
</table>

Activation records are labeled (on the far left) with numbers in parentheses and pointers to these activation records should be expressed using these labels. The activation record (0) represents the environment in which `main` was declared and called.

You should place 0, 1, 2, or 3 in each empty set of parentheses ( ) and draw an arrow connecting each bullet (●) to the entity that pointer would point to.

9. Why is it that even though C has function pointers, it is stack allocated and doesn’t require access links / closures?

Note: If you collaborated with your classmates or used their notes, please note which classmates you collaborated with. If you use an external source, besides the text book, lectures, notes provided by the instructor, and your own intellect, please cite that source. Use quote marks if you are quoting material word-for-word from any source (including the text book).