CIS425 - Midterm 2 Exam

Your Name : ____________________________________

Your Student ID : ______________________________

40 Points. In this problem, you will develop an ML interpreter for a small functional language called 425PL. The language has boolean expressions and functions. The syntax of 425PL programs is given by the following grammar:

```
e ::= x | true | false  | e and e | e or e | fn x=>e  | e e | let x = e in e
```

where the expression `e and e` is true if and only if both expressions are true, and the expression `e or e` is false if and only both expressions are false.

As we know, the interpreter works on a representation of programs, therefore you will have to complete the following datatype definition:

```
datatype term =
```

The interpreter returns a value of type `result`, which in our case means booleans and functions. You will have to complete the following datatype definition:

```
datatype result =
```

The interpreter makes use of an environment defined as

```
datatype env = Env of (string -> result)
```

You can assume that the functions working on the environment are provided to you with the following types:

```
look_up :  env * string -> result
update_env : env*string*result -> env
```

We are now ready to explain the semantics of 425. We assume that boolean expressions are evaluated left-to-right, and we return a result as soon as we can. For example, in the expression `e1 and e2` if the evaluation of `e1` returns `false` then `e2` is not evaluated and `false` is returned immediately; for example, the program

```
let f = fn x => f x in (false and (f 1))
```

returns `false`, rather than looping forever. Likewise, if we have `e1 or e2` and the evaluation of `e1` returns `true` then `e2` is not evaluated and `true` is returned immediately; for example, the program

```
let f = fn x => f x in (true or (f 1))
```

returns `true`, rather than looping forever.

Function calls are evaluated right-to-left; given `e1 e2` you evaluate `e1` first, and then you evaluate `e2`. Let expressions are evaluated according to the rule:

```
env |- e1 --> v1    env[x=v1] |- e2 --> v2
---------------------------------------------
env |- let x=e1 in e2 --> v2
```

If an error in encountered, like trying to execute something of the form (`true false`) or (`(fn x => x) and true`), you should raise the predefined exception ERROR with `raise ERROR`.

The boolean operators in SML are `orelse` and `andalso`, as given below:

```
fun f x = f x;
val a = true  orelse (f 1);
val b = false andalso (f 1)
val f = fn : 'a -> 'b
val a = true : bool
val b = false : bool
```

- Complete the datatype definitions of `term` and `result`
- Write the interpreter implementing dynamic scope.
- Write the interpreter implementing static scope.
- Give an example of a 425PL program that returns `true` in dynamic scope and `false` in static scope

20 points Consider the following Pseudo-ML expression.

```
let x = 9
    fun f y = if y>x then 99 else 4
    fun g h = let x = 7 in h 12 end
in
    let x = 20 in  g f
```

1. Fill in the missing information in the following depiction of the run-time stack after the call to `h 12` inside the body of `g`, assuming you have static scope.

Activation Records

(1)		static link	( )
		x	
		f	•
		g	•
(2)		static link	( )
		x	
(3)	g(f)	static link	( )
		h	•
		x	
(4)	h(12)	static link	( )
		y	

Closures

⟨( ), • ⟩

⟨( ), • ⟩

Compiled Code

code for f
… …

code for g
… …

2. What is the value of this expression? Briefly explain your answer.

20 points Consider the following program written in pseudo-ML

```
let x = 2 in
  let f = fn z => x + z in
        let  x = 100 in
              (f x)
```

Draw the run-time stack after the call to `f x`, just before you are returning to the caller. Draw both the dynamic (or control) link and the access (or static) link.

- What is the result in static scope? Explain how you obtain your result following the run-time stack.

– What is the result in dynamic scope. Explain how you obtain your result following the run-time stack.

20 points Consider the following Pseudo-ML expression.

```
let fun f x = fn y => x
    g = f 1
    in g 2
```

1. What is the result of this program according to dynamic scope? Explain how you obtain your answer.
2. What is the result of this program according to static scope? Explain how you obtain your answer.