

---

# HW 1 – Evaluation and Environments

CS 421 – Spring 2007

Revision 1.0

**Assigned** January 24, 2006

**Due** January 31, 2006, 9:00 AM, in class

**Extension** 48 hours (20% penalty)

---

## 1 Change Log

1.0 Initial Release.

## 2 Turn-In Procedure

Your answers to the following questions are to be hand-written, or printed, neatly on one or more sheets of paper, each with your name in the upper right corner. The homework is to be turned in in class at the start of class. Alternately, you may hand it to Prof. Elsa Gunter in person before the deadline.

## 3 Objectives and Background

The purpose of this HW is to test your understanding of

- the order of evaluation of expressions in OCaml;
- the scope of variables, and the state of environments used during evaluation

Another purpose of HWs is to provide you with experience answering non-programming written questions of the kind you may experience on the midterms and final.

## 4 Problems

1. (15 pts) Below is a fragment of OCaml code, with various program points indicated by numbers with comments.

```
let a = "hi";;
let b = 3;;
(* 1 *)
let c =
  let b = " there" in
(* 2 *)
    (a ^ b);;
(* 3 *)
let x = 5;;
let f x y = x + y + b;;
(* 4 *)
let g z = f x z;;
let w = g b;;
(* 5 *)
```

For each of program points 1, 2, 3, 4 and 5, please describe the environment in effect after evaluation has reached that point. You may assume that the evaluation begins in an empty environment, and that the environment is cumulative thereafter. The program points are supposed to indicate points at which all complete preceding declarations have been fully evaluated.

2. (\* 15 pts \*) Consider the following three OCaml functions, `dots1`, `dots2`, and `dots3`:

```
let rec dots1 () = (print_string "."; dots1());;
let rec dots2 () = (dots2(); print_string ".");;
let rec dots3 () = dots3(print_string ".");;
val dots1 : unit -> 'a = <fun>
val dots2 : unit -> unit = <fun>
val dots3 : unit -> 'a = <fun>
```

Suppose you were to run `dots1();;`, `dots2();;` and `dots3();;` in OCaml, (pressing CTRL + C after at least a minute to terminate infinite loops when necessary).

1. For each program, what behavior would you expect to see?
  2. What is the difference between `dots1`, `dots2`, and `dots3` that causes this different behavior and why does it cause it?
  3. For each program state if it is:
    - (a) recursive,
    - (b) forward recursive,
    - (c) tail-recursive.
3. (\* 10 pts \*) Consider the following two definitions

```
let rec fib1 n = if n < 2 then 1 else f (n - 1) + f (n - 2);;
let rec fib2 n =
  let fib_aux f1 f2 n =
    if n = 0 then f1 else fib_aux (f1 + f2) f1 (n - 1)
  in
  fib_aux 1 0 n;;
```

Explain the difference in the run time (e.g. linear, quadratic, exponential) between the executions of the two.

4. (Extra Credit) (5 pts) Consider the following body of code:

```
let f = (print_string "f"; (fun x -> x + 1));;
(* 1 *)
let g = (fun y -> (print_string "g"; y + 2));;
(* 2 *)
let a = f 3;;
(* 3 *)
let b = g 7;;
(* 4 *)
let c = (f 2, g 4);;
(* 5 *)
```

Indicate what observable behavior (*i.e.* everything you see displayed on the screen) will occur with the evaluation of each declaration. Explain why the evaluation of each declaration generates the described behavior.