

# PhD Qualifying Exam On Programming Languages

Examiner: Prof. Adriana Compagnoni

29 November 2005

## General Remarks

- *This is an open-books, open-notes, but closed laptops exam.*
- The questions refer to the text book *Essentials of Programming Languages* by Friedman, Wand and Haynes (Second Edition).

## Exercises

1. Consider the language of Section 3.6, and extend it with the following primitives for records:

$$\begin{aligned} \langle \text{expression} \rangle & ::= \{ \{ \langle \text{identifier} \rangle = \langle \text{expression} \rangle \}^{*(\cdot)} \} \\ & \quad | \quad (\langle \text{expression} \rangle . \langle \text{identifier} \rangle) \end{aligned}$$

where the first production corresponds to the record constructor and the second production corresponds to field projection. Assume that the language has boolean primitives `true` and `false`.

### Examples:

- `[a = 4, b = true, c = *(3,7)]` is a record with three fields named `a`, `b`, and `c`.
  - `([a = 4, b = true].a)` evaluates to `4`.
  - `let x = 5`  
`in ([d = 3, e = [a = true, b = 75], f = *(x,2)].f)`  
evaluates to `10`.
- (a) Extend the datatypes and the interpreter of the language to include records.

2. Now consider the typed version of this language as in Sections 4.1 and 4.2, where a new type constructor for record types is added.

$$\langle \text{type} - \text{exp} \rangle ::= [\{ \langle \text{identifier} \rangle : \langle \text{type} - \text{exp} \rangle \}^{*(\cdot)}]$$

### Examples:

- `[a = 4, b = true, c = *(3,7)]`  
has type `[a : int, b : bool, c : int]`.
- `([a = 4, b = true].a)` has type `int`.
- `let x = 5`  
`in ([d = 3, e = [a = true, b = 75], f = *(x,2)].f)`  
has type `int`.

- (a) What kinds of typing errors are introduced by these new primitives?
- (b) Design two typing rules: one for records and one for field projection.
- (c) Extend the typechecker to consider records.
- (d) Trace your typechecker and your interpreter on the following expression:

```
let r = [a = 3, b = 4, c = true]
      negate = proc (bool b) if b then false else true
in (negate (r.c))
```

- (e) What goes wrong while typechecking the following expression?

```
let r = [a = 3, b = 4, c = true]
in (r.d)
```

Transcribe the code in your typechecker that finds this error.

**Good luck!**