CS 410/510 Compiling Advanced Languages Homework 3 – due Tuesday, May 9, 2000

Pencil and Paper

1. Write down a well-typed expression in the simply-typed λ-calculus which has as its erasure the following untyped λ-calculus expression:

λ`g.`λ`h.`λ`x.g(h x)`

Give a complete typing derivation tree for you simply-typed expression using the axioms and rules of inference presented in class.

Programming

The files on the web page give a revised interpreter for a core language, together with a type-checker for it.

2. Give a new implementation for the `Env` module that uses the implementation type:

```
'a Env = string -> 'a
```

Hint: The implementation of `lookup` is simply

```
lookup (Env e) id = e id
```

3. (a) Give two interesting examples of core calculus expressions that cause the exception `Bad_Select` to be raised and that fail to pass the typechecker.

(b) Do the same for `Bad_Select`.

4. Figure out what "backwards" means and fill in the correct definition of `meet_types` in `check0.ml`. (Assume for now that the definition of `join_types` is correct!)

5. Give a one-line definition of `subtype` based on `join_types`.

Extra Credit: Examine the implementation of `Fix` in the interpreter. Explain why this implementation method not work for lists of mutually recursive functions like

```
fun f x = g x and g x = f x
```