Technische Universität München *Summer Semester 08*
Fakultät für Informatik *Vesal Vojdani*
Prof. Dr. H. Seidl *vojdanig@in.tum.de*

Compiler Construction

Exercise Sheet 10

Deadline: 9. July 2008, at the lecture, in room 02.07.053, or by e-mail.

Exercise 1: *Polymorphic type-inference* *10 Points*

Infer the types of the polymorphic functions **twice** and **map**:

a) let **twice** = fn (f, x) ⇒ f (f x) in
let v = **twice** (**inc**, 4) in **twice** (**inc**, v)

b) letrec **map** = fn (f, l) ⇒ case l of [] → []; x:xs → f x : **map** (f, xs) in
let v = **map** (**id**, [1;2;3]) in **map** (**string_of_int**, v)

assuming the following types:

string_of_int	: *int* -> *string*
inc	: *int* -> *int*
id	: α -> α

If an expression doesn't type, suggest how to fix it.

Exercise 2: *Type classes* *15 Points*

Recall the type class `Eq` from the lectures:

```
class Eq where
```

$$(=_{Eq}) \ \mathbf{:} \forall\alpha \in Eq.\alpha \to \alpha \to \mathbf{bool}$$

Consider the function **isElem**:

```
let  isElem  = letrec $e$ = fn $l$ ⇒ fn $y$ ⇒
               case $l$ of [ ] → false;
                     $x : xs$ → if $x =_{Eq} y$ then true else ($e\ xs\ y$)
               in $e$
in isElem    ([ 1, 2, 5, 3, 1, 2 ], 3)
```

a) Infer the type of the function **isElem**.

b) Show how $\mathcal{W}$ **isElem** $(\Gamma_0, \emptyset, \emptyset)$ with $\Gamma_0 = \{ =_{Eq} \mapsto \sigma_{Eq}\}$ is computed.

c) Infer the type for the function call above.