LUND UNIVERSITY
Department of Computer Science

Functional programming
20 October 2003, 8-13

Exam

Mark each answer with your initials. Write clearly and comment what you do, that might give you points even if the result is wrong.. Each question is worth five points.

1. Rewrite the following definition so that the argument x does not appear to the left of the equal sign:

```
applyPlusOne f x = (f x) + 1
```

2. The standard prelude contains the function `replicate :: Int -> a -> [a].` A potential definition of this function would be:

```
replicate n x = take n [x,x..]
```

Why is this definition insufficient?

3. What is the type of the following function, and what does it do?

```
h f = fst . head . dropWhile (uncurry (/=)) . ps (iterate f)
    where
    ps g x = zip (tail (g x)) (g x)
```

4. The standard prelude contains the following function:

```
lookup key []   =  Nothing
lookup key ((x,y):xys)
    | key == x   =  Just y
    | otherwise  =  lookup key xys
```

Define `f` and `i` so that the expression `foldl (f key) i` is an equivalent definition of `lookup key`.

5. Define a function `swap` such that

```
flip (curry f) = curry (f.swap)
```

6. Describe as detailed as you can what has to be done in order to use values of type Maybe in arithmethic expressions. Examples:

```
x = Just 5
y = Just 3
z = Nothing

x + y                          -- Just 8
x + z                          -- Nothing
sum [ n+n | n <- [x,y]]        -- Just 16
```