PS Functional Programming
LVA 703025

Midterm Test
Version E

January 8, 2020

Name: ______________________________

Matriculation Number: ______________________________

Points: ______________________________

This exam consists of 2 exercises, for a total of 22 points (so there is roughly 1 point per 2 minutes). The available points per exercise are written in the margin.

### Exercise 1: Multiple Choice 12

In each multiple choice question, you get

- 3 points for 3 correct answers,
- 1 point for 2 correct answers,
- 0 points, otherwise.

Consider the following Haskell function.

```
foo x y z
  | null x || null y = z
  | head x == head y = foo (tail x) (tail y) (z ++ [head x])
  | otherwise        = foo (tail x) (tail y) (tail z ++ [head y])
```

(a) Mark all valid evaluations. (3)

□ `foo "ab" "ab" "cd" = "cdab"`

□ `foo "ab" "ba" "dc" = "ba"`

□ `foo "ab" "ba" "" = "ba"`

(b) Mark all valid type declarations of `foo`. (3)

□ `foo :: [Char] -> [Char] -> [Char] -> [Char]`

□ `foo :: Char a => [a] -> [a] -> [a] -> [a]`

□ `foo :: String -> String -> String -> String`

(c) Mark all valid type declarations of `foo`. (3)

□ `foo :: [Integer, Integer , Integer] -> [Integer]`

□ `foo :: (Num a) => [a] -> [a] -> [a] -> [a]`

□ `foo :: [Integer] -> [Integer] -> ([Integer] -> [Integer])`

(d) Mark all declarations that compile without errors. (3)

□ `data Foo a b = Foo String | Bar a b`

□ `comb x y z = (x z) (x z)`

□ `bar x = x ++ head x`

please turn around

## Exercise 2: Programming

10

(a) Define a datatype `Drink` with three constructors corresponding to coffee, tea, and cola, such that the type-class `Eq` is automatically derived. (1)

(b) Define a function `caffeine` which converts drink into the amount of caffeine in one cup in mg. One cup of coffee, tea or cola contains 105 mg, 40 mg or 44 mg caffeine, respectively. Give both the type definition and the defining equations. (3)

(c) Write a type-class instantiation such that `Drink` becomes an instance of `Ord`, ordered by the amount of caffeine in a cup. You can assume that the function `caffeine` defined above exists. (3)

(d) Define a higher-order function `total` which takes a function – that assigns each drink a number such as `caffeine` – and a list of `Drink` and then returns the total value, e.g., the sum of the caffeine of all drinks in the list. Give both the type definition and the defining equations. (3)

- You can use the predefined functions `head`, `tail`, and `null`.
- It is not allowed to use any other predefined function on lists.
- It is not allowed to use pattern matching.
- The type should be as general as possible.