PS Functional Programming | Midterm Test | December 4, 2019
LVA 703025 | Version D

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 $\boxed{12}$

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 = if x then (foo (not x) (succ y) z) else (y < z)
```

(a) Mark all valid evaluations. (3)

☐ `foo 0 2 False` = `True`
☐ `foo False 15 15` = `False`
☐ `foo True 'a' 'b'` = `False`

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

☐ `foo :: Bool -> Bool -> Bool -> Bool`
☐ `foo :: (Enum a,Bool a,Ord a) => a -> a -> a -> a`
☐ `foo :: Integral a => Bool -> a -> a -> Bool`

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

☐ `foo :: (Enum a,Enum b,Ord a,Ord b) => Bool -> (a,b) -> (a,b) -> Bool`
☐ `foo :: Bool -> (Float -> Float -> Bool)`
☐ `foo :: Bool -> Char -> Char -> Bool`

(d) Mark all declarations that compile without errors (the third consists of two declarations). (3)

☐ `three2 x y z = (x y) (z 2)`
☐ `eqRotate x y z = if x == y then True else (eqRotate $ z) x y`
☐
```
testEq x x = True
testEq x _ = False
```

Exercise 2: Programming 10

(a) Define a data type `Fruit` with three constructors corresponding to a banana, a cherry (Kirsche) and a citron deriving `Eq`, and a data type `Color` with two constructors corresponding to yellow and red, deriving `Show`. (2)

(b) Define a function `ripe` which maps a `Fruit` to the `Color` it has when ripe (reif). Give both the type definition and the defining equations. (2)

(c) Write a type-class instantiation such that `Fruit` becomes an instance of `Ord` such that fruits are ordered according to your preference, with greater fruit being more preferred. (3)

(d) Define a function `twice` that applies a given function twice to all elements of a list. For instance, evaluating `twice succ "haca"` should result in `"jcec"` and evaluating `twice sqrt [0,16,1]` should result in `[0.0,2.0,1.0]`. Give both the type definition and the defining equations. (3)