

### ICS 2021 Problem Sheet #6

**Problem 6.1:** *long life diet rules*

(1+3+1+3 = 8 points)

Uwe Schöning published in his book “Logic for Computer Scientists” the following little story:

A 100 year old was asked “What is the secret of having a long life?” and he did respond to stick to three diet rules:

1. If you don't take beer, you must have fish.
2. If you have both beer and fish, don't have ice cream.
3. If you have ice cream or don't have beer, then don't have fish.

The questioner found this answer quite confusing. Can you help to simplify the answer?

We introduce three boolean variables: The variable  $B$  is true if we have beer, the variable  $F$  is true if we have fish, and the variable  $I$  is true if we have ice cream.

- a) Provide a boolean formula for the function  $D(B, F, I)$  that captures the three rules.
- b) Construct a truth table that shows all interpretations of  $D(B, F, I)$ . Break things into meaningful steps so that we can award partial points in case things go wrong somewhere.
- c) Out of the truth table, derive a simpler boolean formula defining  $D(B, F, I)$ .
- d) Take the boolean formula from a) and algebraically derive the simpler boolean formula from c). Annotate each step of your derivation with the equivalence law that you apply so that we can follow along.

**Problem 6.2:** *long life diet rules*

(1+1 = 2 points)

Uwe Schöning published in his book “Logic for Computer Scientists” the following little story:

A 100 year old was asked “What is the secret of having a long life?” and he did respond to stick to three diet rules:

1. If you don't take beer, you must have fish.
2. If you have both beer and fish, don't have ice cream.
3. If you have ice cream or don't have beer, then don't have fish.

The questioner found this answer quite confusing. Can you help to simplify the answer?

We introduce three boolean variables: The variable  $B$  is true if we have beer, the variable  $F$  is true if we have fish, and the variable  $I$  is true if we have ice cream.

- a) Define a function

```
diet :: Bool -> Bool -> Bool -> Bool
```

that implements the three diet rules directly following the description given above and a function

```
diet' :: Bool -> Bool -> Bool -> Bool
```

that implements a simplified diet formula.

b) Define a function

```
truthTable :: (Bool -> Bool -> Bool -> Bool) -> [(Bool, Bool, Bool, Bool)]
```

takes a (diet) function as an argument and returns a list where each element is a 4-tuple representing three input values passed to the (diet) function followed by the function's result.

Submit your Haskell code as a plain text file.

Below is a unit test template that you can use to fill in your code.

```
1  module Main (main) where
2
3  import Test.HUnit
4
5  -- The function diet implements the three diet rules directly.
6  diet :: Bool -> Bool -> Bool -> Bool
7  diet b f i = undefined
8
9  -- The function diet' implements the simplified diet formula.
10 diet' :: Bool -> Bool -> Bool -> Bool
11 diet' b f i = undefined
12
13 -- The function truthTable takes a function as an argument and returns
14 -- a list where each element is a 4-tuple representing three input
15 -- value passed to the function followed by the function's result.
16 truthTable :: (Bool -> Bool -> Bool -> Bool) -> [(Bool, Bool, Bool, Bool)]
17 truthTable f = undefined
18
19 -- Test whether the two truth tables returned are the same (which is
20 -- not a very sharp test but I do not want to reveal too many details).
21 -- You may want to add your own test cases...
22 dietTests = TestList [ truthTable diet ~?= truthTable diet' ]
23
24 main = runTestTT dietTests
```