Introduction to Computer Science

Assignment 08

Constructor University,
Bremen, Germany

April 10, 2024

- Marks are only awarded for producing the correct solutions and justifying how you arrived at them.
- Please write legibly. Illegible answers will NOT be marked.

Question 1

Using the Definition of the syntax of Boolean formulas, show that the expression: $\neg(x \vee (y \wedge 1))$ is a well-formed formula.

[1 mark]

Question 2

Consider the set of Boolean variables $D = \{x_1, x_2, x_3\}$ and the interpretation $\mathcal{I}(x_1) = 1$, $\mathcal{I}(x_2) = 0$, and $\mathcal{I}(x_3) = 1$. Determine the semantics of the formula: $x_1 \vee \neg(x_2 \vee 0) \wedge x_3$ using the Definition of the extended interpretation for formulas $\mathcal{I}^*$.

[1 mark]

Question 3

By means of a truth table, demonstrate the validity of the De Morgan's laws for three variables, i.e.:

$$\neg(\varphi \wedge \psi \wedge \chi) \equiv \neg\varphi \vee \neg\psi \vee \neg\chi$$

[1 mark]

Question 4

By means of algebraic manipulation (Boolean equivalence laws), show that the following Boolean identity holds:

$$(\neg x \wedge y) \vee (y \wedge \neg z) \vee (y \wedge z) \vee (x \wedge \neg y \wedge \neg z) \equiv y \vee (x \wedge \neg z)$$

[1 mark]

Question 5

For the formula: $\neg x \lor \neg y \lor (x \land y \land \neg z)$ obtain the *Disjunctive Normal Form* (DNF).

[1 mark]

Question 6

A binary T has the following traversal sequences:

preorder: $J, G, A, E, V, L, M, W, Z$
inorder: $A, E, G, J, L, M, V, W, Z$

Present the binary tree T.

[2 marks]

Question 7

Natural numbers are an inductive data type. The set of natural numbers is infinite, yet it can be defined in Haskell by an Algebraic Data Type:

```
data Natural = Zero | Succ Natural

instance Show Natural where
    -- show :: Natural -> String
    show n = show (iterate n) where
        iterate Zero        = 0
        iterate (Succ m)    = 1 + (iterate m)

-- some constant
zero, one, two, three, four :: Natural
zero = Zero
one = Succ Zero
two = Succ one
three = Succ two

-- addition
infixl 6 <+>
(<+>) :: Natural -> Natural -> Natural
n <+> Zero      = n
n <+> (Succ m)  = Succ (n <+> m)
```

In this representation, implement the multiplication (represented by `<.>`) as an infix left-associative operator with higher precedence than addition.

[3 marks]