COMP4161 T3/2019
Advanced Topics in Software Verification

Assignment 1

This assignment starts on Monday 30th September 2019 at 9am and is due on Monday 7th October 2019 6pm. We will accept plain text (.txt) files, PDF (.pdf) files, and Isabelle theory (.thy) files.

The assignment is take-home. This does NOT mean you can work in groups. Each submission is personal. For more information, see the plagiarism policy: https://student.unsw.edu.au/plagiarism

Submit using `give` on a CSE machine:

```
give cs4161 a1 files ...
```

For example:

```
give cs4161 a1 a1.thy a1.pdf
```

1 Types (17 marks)

Construct a type derivation tree for the term $\lambda a\ b\ c.\ x\ (a\ c\ c)\ (b\ a)$.

Each node of the tree should correspond to the application of a *single* typing rule, indicating which typing rule is used at each step.

Under which contexts is the term type correct?

2 λ-Calculus (23 marks)

Recall the encoding of natural numbers and addition in lambda calculus (Church Numerals) seen in the lecture:

`0`	$\equiv$	$\lambda f\ x.\ x$
`1`	$\equiv$	$\lambda f\ x.\ f\ x$
`2`	$\equiv$	$\lambda f\ x.\ f\ (f\ x)$
`3`	$\equiv$	$\lambda f\ x.\ f\ (f\ (f\ x))$
`add`	$\equiv$	$\lambda m\ n.\ \lambda f\ x.m\ f\ (n\ f\ x)$

(a) Show that the β normal form for `add 1 2` is `3`. Justify your answer by providing the β reduction steps leading from the term to its normal form. Each step should only reduce *one* redex (i.e. one reduction per step). Ideally, you would underline the redex being reduced. (11 marks)

(b) Provide a type for `3`. Justify your answer by providing a derivation tree. (6 marks)

(c) Assuming `add 1 2` is of type T, then what can T be? Justify your answer. (6 marks)

3 Propositional Logic (36 marks)

Prove each of the following statements, using only the proof methods `rule`, `erule`, `assumption`, and `cases`; and using only the proof rules `impI`, `impE`, `conjI`, `conjE`, `disjI1`, `disjI2`, `disjE`, `notI`, `notE`, `iffI`, `iffE`, `iffD1`, `iffD2`, `ccontr`, `classical`, `FalseE`, `TrueI`, `conjunct1`, `conjunct2`, and `mp`. You do not need to use all of these methods and rules.

(a) $B \longrightarrow A \vee B \vee C$ (3 marks)

(b) $(A \longrightarrow B) \wedge (B \longrightarrow C) \wedge (C \longrightarrow D) \longrightarrow A \longrightarrow D$ (7 marks)

(c) $((A \vee B) \wedge C) = (A \wedge C \vee C \wedge B)$ (7 marks)

(d) $(\neg\ P) = (\neg\ \neg\ \neg\ P)$ (7 marks)

(e) $(A \longrightarrow B) = (\neg\ (\neg\ B \wedge A))$ (6 marks)

(f) $(A \vee C) \wedge (C \longrightarrow A) \wedge (A \longrightarrow D) \longrightarrow D$ (6 marks)

4 Higher-Order Logic (24 marks)

Prove the following statements, using only the proof methods: `rule`, `erule`, `assumption`, `frule`, `drule`, `rule_tac`, `erule_tac`, `frule_tac`, `drule_tac`, `rename_tac`, and `case_tac`; and using only the proof rules: `impI`, `impE`, `conjI`, `conjE`, `disjI1`, `disjI2`, `disjE`, `notI`, `notE`, `iffI`, `iffE`, `iffD1`, `iffD2`, `ccontr`, `classical`, `FalseE`, `TrueI`, `allI`, `allE`, `exI`, and `exE`. You do not need to use all of these methods and rules. You may use rules proved in earlier parts of the question when proving later parts.

(a) $(\forall x.\ P\ x) \wedge (\forall x.\ Q\ x) \longrightarrow (\forall x.\ Q\ x \wedge P\ x)$ (6 marks)

(b) $(\forall x.\ P\ x \longrightarrow Q\ x) \wedge (\forall x.\ P\ x) \longrightarrow (\forall x.\ Q\ x)$ (6 marks)

(c) $(\forall x.\ P\ x \vee Q\ x) \wedge ((\exists x.\ Q\ x) \longrightarrow W) \wedge ((\exists x.\ P\ x) \longrightarrow Z) \longrightarrow W \vee Z$ (6 marks)

(d) $(\forall x.\ R\ x\ x) \longrightarrow \neg\ (\forall x\ y.\ R\ y\ x \longrightarrow \neg\ R\ x\ y)$ (6 marks)