COMP4161 T3/2022
Advanced Topics in Software Verification

Assignment 1

This assignment starts on Tuesday 20th September 2022 and is due on Tuesday 27th September 2022 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 (15 marks)

(a) Provide the most general type for the term `λx y z. z x (a y y)`. Show a type derivation tree to justify your answer.

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? (10 marks)

(b) Find a term that has the following type:

`('a ⇒ 'a ⇒ 'b ⇒ 'c) ⇒ 'a ⇒ 'b ⇒ 'c`

(You don't need to provide a type derivation, just the term).

(5 marks)

2 λ-Calculus (23 marks)

Recall the encoding of booleans and booleans operations in lambda calculus seen in the lecture:

```
true    ≡  λx y. x
false   ≡  λx y. y
ifthen  ≡  λz x y. z x y
not     ≡  λx. ifthen x false true
```

(a) Define (in Isabelle) `xor`, the exclusive OR operator, using the definitions of `ifthen` and `not` (3 marks).

(b) Show by beta reduction (by hand, not using Isabelle) that:

`xor` $=_\beta$ `λx y. x (y false true) y`

then show by beta reduction that:

`xor false y` $=_\beta$ `y`.

and

`xor true y` $=_\beta$ `not y`.

Each step should be a single beta reduction or definition unfolding. Alpha conversion is allowed (14 marks).

(c) Prove the above 3 lemmas in Isabelle, using `unfold` and `refl`. Explain (informally) what the `refl` theorem states and explain why it can be used to prove the lemmas (6 marks).

3 Propositional Logic (35 marks)

Prove each of the following statements (after stating them in Isabelle for (c) and (f)), using only the proof methods
`rule`, `erule`, `assumption`, `cases`, `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`, `TrueE`, `conjunct1`, `conjunct2`, and `mp`.
You do not need to use all of these methods and rules.

(a) `A ∨ B ∨ A ⟶ B ∨ A` (3 marks)

(b) `(¬ a ⟷ b) ⟷ ¬ b ⟷ a` (3 marks)

(c) "Saying that if Alice is here then Bob is definitely not here is the same as saying that they can't both be here." (5 marks)

(d) `(A ∧ B ∨ C) = ((A ∨ C) ∧ (B ∨ C))` (5 marks)

(e) `¬ P ∧ Q ⟶ ¬ (R ∧ P) ∧ (R ⟶ Q)` (7 marks)

(f) "If either it is not raining or you have an umbrella, then it is not possible that you do not have an umbrella at a time where it is also raining." (5 marks)

(g) `(((f ⟶ g) ∧ h ⟶ f) ⟶ g) = ((f ⟶ g) ∧ (g ∨ h))` (7 marks)

4 Higher-Order Logic (27 marks)

Prove each of the following statements (after stating them in Isabelle for (d)), using only the proof methods and proof rules stated in the previous question, plus any of the following proof rules:
`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) `(∀x. ¬ P x) = (∄x. P x)` (4 marks)

(b) `(∀x. B x) ∨ (∀y. A y) ⟶ (∀x y. A y ∨ B x)` (4 marks)

(c) `(∀x y. A y ∨ B x) ⟶ (∀x. B x) ∨ (∀y. A y)` (7 marks)

(d) "If any proposition is true then the value True is the same as the value False." (4 marks)

(e) `((∃x. A x) ⟶ ¬ C) ⟶ (∀x. B x ⟶ A x) ⟶ (∃x. B x) ⟶ ¬ C` (4 marks)

(f) `(∀x. ¬ R x x) ⟶ ¬ (∀x y. ¬ R x y ⟶ R y x)` (4 marks)