COMP3161/9164 22T3 Assignment 0
Proofs

Marks :	15% of overall marks for the course. A mark of x (out of 100) on this assignment will translate to $.15x$ course marks.
Due date:	**Thursday, 6th of October 2022, 12 noon (Sydney Time)**

1 Task

In this assignment, you will formally model a language of boolean computations using a variety of semantic techniques, including its syntax and sematics, and its compilation to various target languages.

Prepare your answers in one PDF file, preferably using LaTeX, where all prose is typeset. Figures may be drawn, but make sure they are legible. Please ensure all mathematics is formatted correctly. Some guidance will be posted on the course website.

Submit your PDF using the CSE give system, by typing the command


```
give cs3161 Proofs Proofs.pdf
```

or by using the CSE give web interface.

Part A (25 marks)

Consider the language of boolean expressions $\mathcal{P}$ containing just literals ($\texttt{True}$, $\texttt{False}$), parentheses, conjunction ($\wedge$) and negation ($\neg$):

$$\mathcal{P} = \{\texttt{True}, \texttt{False}, \neg\texttt{True}, \neg\texttt{False}, \texttt{True} \wedge \texttt{False}, \neg(\texttt{True} \wedge \texttt{False}), \dots\}$$

1. Write down a set of inference rules that define the set $\mathcal{P}$. The rules may be ambiguous. *(5 marks)*

2. The operator $\neg$ has the highest precedence, and conjunction is right-associative. Define a set of simultaneous judgements to define the language without any ambiguity. *(5 marks)*

3. Here is an *abstract syntax* $\mathcal{B}$ for the same language:

$$\mathcal{B} ::= \mathsf{Not}\ \mathcal{B} \mid \mathsf{And}\ \mathcal{B}\ \mathcal{B} \mid \mathsf{True} \mid \mathsf{False}$$

   Write an inductive definition for the *parsing* relation connecting your unambiguous judgements to this abstract syntax. *(5 marks)*

4. Here is a big-step semantics for the language $\mathcal{B}$

$$\frac{x \Downarrow \mathsf{True}}{\mathsf{Not}\ x \Downarrow \mathsf{False}} \qquad \frac{x \Downarrow \mathsf{False}}{\mathsf{Not}\ x \Downarrow \mathsf{True}} \qquad \frac{x \Downarrow \mathsf{False}}{\mathsf{And}\ x\ y \Downarrow \mathsf{False}} \qquad \frac{x \Downarrow \mathsf{True} \quad y \Downarrow v}{\mathsf{And}\ x\ y \Downarrow v}$$

$$\frac{}{\mathsf{True} \Downarrow \mathsf{True}} \qquad \frac{}{\mathsf{False} \Downarrow \mathsf{False}}$$

   a) Show the evaluation of $\mathsf{And}\ (\mathsf{Not}\ (\mathsf{And}\ \mathsf{True}\ \mathsf{False}))\ \mathsf{True}$ with a derivation tree. *(5 marks)*

   b) Consider the following inference rule:

$$\frac{y \Downarrow \mathsf{False}}{\mathsf{And}\ x\ y \Downarrow \mathsf{False}}$$

   If we assume that $x\ \mathcal{B}$ holds, is this rule derivable? Is it admissible? And if we *don't* assume that $x\ \mathcal{B}$ holds, how does this change your answers? Justify your answers. *(5 marks)*

Part B (20 marks)

Here is a small-step semantics for a language $\mathcal{L}$ with `True`, `False` and `if` expressions:

$$\frac{c \mapsto c'}{(\mathtt{If}\ c\ t\ e) \mapsto (\mathtt{If}\ c'\ t\ e)}(1) \qquad \frac{}{(\mathtt{If}\ \mathtt{True}\ t\ e) \mapsto t}(2) \qquad \frac{}{(\mathtt{If}\ \mathtt{False}\ t\ e) \mapsto e}(3)$$

1. Show the full evaluation of the term (`If True` (`If True False True`) `False`). *(5 marks)*

2. Define an equivalent *big-step* semantics for $\mathcal{L}$. *(5 marks)*

3. Prove that if $e \Downarrow v$ then $e \overset{\star}{\mapsto} v$, where $\Downarrow$ is the big-step semantics you defined in the previous question, and $\overset{\star}{\mapsto}$ is the reflexive and transitive closure of $\mapsto$. Use rule induction on $e \Downarrow v$. *(10 marks)*

Part C (15 marks)

1. Define a recursive *compilation function* $c : \mathcal{B} \to \mathcal{L}$ which converts expressions in $\mathcal{B}$ to expressions in $\mathcal{L}$. *(5 marks)*

2. Prove that for all e, $e \Downarrow v$ implies $c(e) \Downarrow v$, by rule induction on the assumption that $e \Downarrow v$. *(10 marks)*

Part D (40 marks)

1. Here is a term in λ-calculus:

$$(\lambda n.\ \lambda f.\ \lambda x.\ (n\ f\ (f\ x)))\ (\lambda f.\ \lambda x.\ f\ x)$$

   a) Fully β-reduce the above λ-term. Show all intermediate beta reduction steps. *(5 marks)*

   b) Identify an η-reducible expression in the above (unreduced) term. *(5 marks)*

2. Recall that in λ-calculus, booleans can be encoded as binary functions that return one of their arguments:

$$\mathbf{T} \equiv (\lambda x.\ \lambda y.\ x)$$

$$\mathbf{F} \equiv (\lambda x.\ \lambda y.\ y)$$

   Either via $\mathcal{L}$ or directly, define a function $\mathsf{d} : \mathcal{B} \to \lambda$ which converts expressions in $\mathcal{B}$ to λ-calculus. *(5 marks)*

3. Prove that for all e such that $e \Downarrow v$ it holds that $d(e) \equiv_{\alpha\beta\eta} v'$, where v' is the λ-calculus encoding of v. *(10 marks)*

4. Suppose we added *unary local function definitions* to our language $\mathcal{P}$. Here's an example in concrete syntax:

$$\begin{array}{l} \mathsf{let} \\ \quad g(x) = \neg x \\ \mathsf{in} \\ \quad g(\mathsf{True}) \\ \mathsf{end} \end{array}$$

   We limit ourselves to non-recursive bindings (meaning functions can't call themselves), and first-order functions (meaning functions require boolean arguments).

   a) Extend the *abstract* syntax for $\mathcal{B}$ from question A.3 so that it supports the features used in the above example. Use first-order abstract syntax with explicit strings. You don't have to extend the parsing relation. *(5 marks)*

   b) Define a scope-checking judgement, similar to the **ok** judgement from the lectures. It should check (a) that all names of variables and functions are used only within their scopes; and (b) that names used in variable (or function) position are indeed the names of variables (or functions). Hence, the following expressions should both be rejected:

$$\begin{array}{l} \mathsf{let} \\ \quad f(x) = \neg x \\ \mathsf{in} \\ \quad f(x) \\ \mathsf{end} \end{array} \qquad \begin{array}{l} \mathsf{let} \\ \quad f(x) = \neg x \\ \mathsf{in} \\ \quad f(f) \\ \mathsf{end} \end{array} \qquad \begin{array}{l} \mathsf{let} \\ \quad f(x) = x(\mathsf{True}) \\ \mathsf{in} \\ \quad f(\mathsf{False}) \\ \mathsf{end} \end{array}$$

The following are examples of things that should be accepted: nested definitions, and shadowed definitions.

```
let
  f(x) =
    let
      g(y) = ¬x ∧ y
    in
      g(x) ∧ ¬g(x)
    end
in
  f(False)
end
```

```
let
  f(x) = x
in
  let
    f(x) = f(x)
  in
    f(True)
  end
end
```

Note that the latter example is *not* a recursive call. *(10 marks)*

2 Late Penalty

You may submit up to five days (120 hours) late. Each day of lateness corresponds to a 5% reduction of your total mark. For example, if your assignment is worth 88% and you submit it two days late, you get 78%. If you submit it more than five days late, you get 0%.

Course staff cannot grant assignment extensions—if you need an extensions, you have to apply for special consideration through the standard procedure. More information here: `https://www.student.unsw.edu.au/special-consideration`

3 Plagiarism

Many students do not appear to understand what is regarded as plagiarism. This is no defense. Before submitting any work you should read and understand the UNSW plagiarism policy `https://student.unsw.edu.au/plagiarism`.

All work submitted for assessment must be entirely your own work. We regard unacknowledged copying of material, in whole or part, as an extremely serious offence. In this course submission of any work derived from another person, or solely or jointly written by and or with someone else, without clear and explicit acknowledgement, will be severely punished and may result in automatic failure for the course and a mark of zero for the course. Note this includes including unreferenced work from books, the internet, etc.

Do not provide or show your assessable work to any other person. Allowing another student to copy from you will, at the very least, result in zero for that assessment. If you knowingly provide or show your assessment work to another person for any reason, and work derived from it is subsequently submitted, you will be penalized, even if the work was submitted without your knowledge or consent. This will apply even if your work is submitted by a third party unknown to you. You should keep your work private until submissions have closed.

If you are unsure about whether certain activities would constitute plagiarism ask us before engaging in them!