UNIVERSITY OF EDINBURGH

COLLEGE OF SCIENCE AND ENGINEERING

SCHOOL OF INFORMATICS

INFR10040TYPES AND SEMANTICS FOR PROGRAMMING LANGUAGES

Monday 19 th May 2014

14:30 to 16:30

INSTRUCTIONS TO CANDIDATES

Answer QUESTION 1 and ONE other question.

Question 1 is COMPULSORY.

All questions carry equal weight.

CALCULATORS MAY NOT BE USED IN THIS EXAMINATION

Year 4 Courses

Convener: I. Stark
External Examiners: A. Cohn, T. Field

THIS EXAMINATION WILL BE MARKED ANONYMOUSLY

1. **You MUST answer this question.**

This question uses the library definition of `list` in Coq, which includes the function ++.

Here is an informal definition of the predicate `last`.

$$\texttt{last\_end}\frac{}{\texttt{last}\ (x :: nil)\ x} \qquad \texttt{last\_step}\frac{\texttt{last}\ xs\ y}{\texttt{last}\ (x :: xs)\ y}$$

- Formalise the definition above. [*12 marks*]
- Prove the following.

$$\texttt{Theorem last\_app} : \forall (X : \texttt{Type})\ (x : X)\ (xs : \texttt{list}\ X),$$
$$\texttt{last}\ xs\ x \longrightarrow \exists xs'.\, xs = xs' \mathbin{++} [x]$$

[*13 marks*]

2. You will be provided with a definition of a simple imperative language in Coq.

   Consider a construct satisfying the following rules:

   Evaluation:

$$\texttt{E\_LoopEnd}\frac{\begin{array}{c} c_1/st \Downarrow st' \\ \texttt{beval}\ st'\ b = \texttt{false} \end{array}}{\texttt{LOOP}\ c_1\ \texttt{WHILE}\ b\ \texttt{DO}\ c_2\ \texttt{END}/st \Downarrow st'}$$

$$\texttt{E\_LoopLoop}\frac{\begin{array}{c} c_1/st \Downarrow st' \\ \texttt{beval}\ st'\ b = \texttt{true} \\ c_2/st' \Downarrow st'' \\ \texttt{LOOP}\ c_1\ \texttt{WHILE}\ b\ \texttt{DO}\ c_2\ \texttt{END}/st'' \Downarrow st''' \end{array}}{\texttt{LOOP}\ c_1\ \texttt{WHILE}\ b\ \texttt{DO}\ c_2\ \texttt{END}/st \Downarrow st'''}$$

   Hoare logic:

$$\texttt{hoare\_loop}\frac{\begin{array}{c} \{\{P\}\}\ c_1\ \{\{Q\}\} \\ \{\{Q \wedge b\}\}\ c_2\ \{\{P\}\} \end{array}}{\{\{P\}\}\ \texttt{LOOP}\ c_1\ \texttt{WHILE}\ b\ \texttt{DO}\ c_2\ \texttt{END}\ \{\{Q \wedge \neg b\}\}}$$

   - Extend the given definition to formalise the evaluation rules. *[12 marks]*
   - Prove the Hoare rule. You will be provided with proofs of Hoare rules for the simple imperative language that you may modify. *[13 marks]*

3. Problem 3

You will be provided with a definition of simply-typed lambda calculus in Coq.

Consider constructs satisfying the following rules:

Evaluation:

$$\texttt{ST\_Succ}\ \frac{t \Longrightarrow t'}{(\texttt{succ}\ t) \Longrightarrow (\texttt{succ}\ t')}$$

$$\texttt{ST\_Ncase}\ \frac{t_1 \Longrightarrow t_1'}{\begin{array}{c}(\texttt{ncase}\ t_1\ \texttt{of zero} \Rightarrow t_2 \mid \texttt{succ}\ x \Rightarrow t_3) \\ \Longrightarrow (\texttt{ncase}\ t_1'\ \texttt{of zero} \Rightarrow t_2 \mid \texttt{succ}\ x \Rightarrow t_3)\end{array}}$$

$$\texttt{ST\_NcaseZero}\ \frac{}{(\texttt{ncase zero of zero} \Rightarrow t_2 \mid \texttt{succ}\ x \Rightarrow t_3) \Longrightarrow t_2}$$

$$\texttt{ST\_NcaseSucc}\ \frac{\texttt{value}\ v_1}{(\texttt{ncase}\ (\texttt{succ}\ v_1)\ \texttt{of zero} \Rightarrow t_2 \mid \texttt{succ}\ x \Rightarrow t_3) \Longrightarrow [x := v_1]t_3}$$

Typing

$$\texttt{T\_Zero}\ \frac{}{\Gamma \vdash \texttt{zero} \in \texttt{Nat}}$$

$$\texttt{T\_Succ}\ \frac{\Gamma \vdash t \in \texttt{Nat}}{\Gamma \vdash (\texttt{succ}\ t) \in \texttt{Nat}}$$

$$\texttt{T\_Ncase}\ \frac{\begin{array}{c}\Gamma \vdash t_1 \in \texttt{Nat} \\ \Gamma \vdash t_2 \in T \\ \Gamma, x \in \texttt{Nat} \vdash t_3 \in T\end{array}}{\Gamma \vdash (\texttt{ncase}\ t_1\ \texttt{of zero} \Rightarrow t_2 \mid \texttt{succ}\ x \Rightarrow t_3) \in T}$$

- Extend the given definition to formalise the evaluation and typing rules. [*12 marks*]
- Prove progress. You will be provided with a proof of progress for the simply-typed lambda calculus that you may extend. [*13 marks*]