UNIVERSITY OF EDINBURGH

COLLEGE OF SCIENCE AND ENGINEERING

SCHOOL OF INFORMATICS

INFR11114 TYPES AND SEMANTICS FOR PROGRAMMING LANGUAGES

Tuesday 10th December 2019

09:30 to 11:30

INSTRUCTIONS TO CANDIDATES

Answer QUESTION 1 and ONE other question.

Question 1 is COMPULSORY. If both QUESTION 2 and QUESTION 3 are answered, only QUESTION 2 will be marked.

All questions carry equal weight.

CALCULATORS MAY NOT BE USED IN THIS EXAMINATION

MSc Courses

Convener: V.Nagarajan
External Examiners: W.Knottenbelt, M.Dunlop, M.Niranjan, E.Vasilaki

THIS EXAMINATION WILL BE MARKED ANONYMOUSLY

1. THIS QUESTION IS COMPULSORY

   The datatype `Sexp` (inspired by *S-expressions* as found in Lisp) has two constructors:

$$\frac{}{\texttt{nil} : \texttt{Sexp}} \qquad \frac{x : \texttt{Sexp} \quad y : \texttt{Sexp}}{x \mathbin{\hat{}} y : \texttt{Sexp}}$$

   Here are informal definitions of two higher-order predicates, `AllS` and `AnyS`, each of which takes a predicate P over `Sexp` as an argument.

$$\texttt{leaf}\,\frac{P\ \texttt{nil}}{\texttt{AllS}\ P\ \texttt{nil}} \qquad \texttt{node}\,\frac{P\ (x \mathbin{\hat{}} y) \quad \texttt{AllS}\ P\ x \quad \texttt{AllS}\ P\ y}{\texttt{AllS}\ P\ (x \mathbin{\hat{}} y)}$$

$$\texttt{here}\,\frac{P\ x}{\texttt{AnyS}\ P\ x} \qquad \texttt{left}\,\frac{\texttt{AnyS}\ P\ x}{\texttt{AnyS}\ P\ (x \mathbin{\hat{}} y)} \qquad \texttt{right}\,\frac{\texttt{AnyS}\ P\ y}{\texttt{AnyS}\ P\ (x \mathbin{\hat{}} y)}$$

   (a) Formalise the definitions of `Sexp`, `AllS`, and `AnyS`. [*12 marks*]

   (b) Show that $\texttt{AnyS}\ (\neg\_ \circ P)\ x$ implies $\neg\ (\texttt{AllS}\ P\ x)$, for all P and x. [*13 marks*]

2. ANSWER EITHER THIS QUESTION OR QUESTION 3

You will be provided with a definition of intrinsically-typed lambda calculus in Agda. Consider constructs satisfying the following rules, written in extrinsically-typed style.

Typing:

$$\texttt{nil}\ \frac{}{\Gamma \vdash \texttt{nil} ⦂ \texttt{Sexp}} \qquad \texttt{\_\^{}\_}\ \frac{\Gamma \vdash M ⦂ \texttt{Sexp} \quad \Gamma \vdash N ⦂ \texttt{Sexp}}{\Gamma \vdash M \mathbin{\hat{}} N ⦂ \texttt{Sexp}}$$

$$\texttt{caseS}\ \frac{\Gamma \vdash L ⦂ \texttt{Sexp} \quad \Gamma \vdash M ⦂ A \quad \Gamma\ ,\ x ⦂ \texttt{Sexp}\ ,\ y ⦂ \texttt{Sexp} \vdash N ⦂ A}{\Gamma \vdash \texttt{caseS}\ L\ [\texttt{nil} \Rightarrow\ M \mid\mid x \mathbin{\hat{}} y \Rightarrow N] ⦂ A}$$

Values:

$$\texttt{V-nil}\ \frac{}{\texttt{Value nil}} \qquad \texttt{V-\^{}}\ \frac{\texttt{Value}\ V \quad \texttt{Value}\ W}{\texttt{Value}\ (V \mathbin{\hat{}} W)}$$

Reduction:

$$\xi\texttt{-\^{}}_1\ \frac{M \longrightarrow M'}{M \mathbin{\hat{}} N \longrightarrow M' \mathbin{\hat{}} N} \qquad \xi\texttt{-\^{}}_2\ \frac{\texttt{Value}\ V \quad N \longrightarrow N'}{V \mathbin{\hat{}} N \longrightarrow V \mathbin{\hat{}} N'}$$

$$\xi\texttt{-caseS}\ \frac{L \longrightarrow L'}{\texttt{caseS}\ L\ [\texttt{nil} \Rightarrow\ M \mid\mid x \mathbin{\hat{}} y \Rightarrow N] \longrightarrow \texttt{caseS}\ L'\ [\texttt{nil} \Rightarrow\ M \mid\mid x \mathbin{\hat{}} y \Rightarrow N]}$$

$$\beta\texttt{-nil}\ \frac{}{\texttt{caseS nil}\ [\texttt{nil} \Rightarrow\ M \mid\mid x \mathbin{\hat{}} y \Rightarrow N] \longrightarrow M}$$

$$\beta\texttt{-\^{}}\ \frac{\texttt{Value}\ V \quad \texttt{Value}\ W}{\texttt{caseS}\ (V \mathbin{\hat{}} W)\ [\texttt{nil} \Rightarrow\ M \mid\mid x \mathbin{\hat{}} y \Rightarrow N] \longrightarrow N\,[x := V]\,[y := W]}$$

(a) Extend the given definition to formalise the above rules, including any other required definitions. [*15 marks*]

(b) Prove progress. You will be provided with a proof of progress for the simply-typed lambda calculus that you may extend. [*10 marks*]

Please delimit any code you add as follows.

```
-- begin
-- end
```

3. ANSWER EITHER THIS QUESTION OR QUESTION 2

You will be provided with a definition of inference for extrinsically-typed lambda calculus in Agda. Consider constructs satisfying the following rules, written in extrinsically-typed style that support bidirectional inference.

Typing:

$$\texttt{nil}\ \frac{}{\Gamma \vdash \texttt{nil} \downarrow \texttt{Sexp}} \qquad \texttt{\_\^{}\_}\ \frac{\begin{array}{c}\Gamma \vdash M \downarrow \texttt{Sexp} \\ \Gamma \vdash N \downarrow \texttt{Sexp}\end{array}}{\Gamma \vdash M \ \hat{}\ N \downarrow \texttt{Sexp}}$$

$$\texttt{caseS}\ \frac{\begin{array}{c}\Gamma \vdash L \uparrow \texttt{Sexp} \\ \Gamma \vdash M \downarrow A \\ \Gamma\ ,\ x ⦂ \texttt{Sexp}\ ,\ y ⦂ \texttt{Sexp} \vdash N \downarrow A\end{array}}{\Gamma \vdash \texttt{caseS}\ L\ [\texttt{nil} \Rightarrow\ M\ \texttt{||}\ x\ \hat{}\ y \Rightarrow N] \downarrow A}$$

(a) Extend the given definition to formalise the typing rules, and update the definition of equality on types. [*10 marks*]

(b) Extend the code to support type inference for the new features. [*15 marks*]

Please delimit any code you add as follows.

```
-- begin
-- end
```