THE UNIVERSITY OF CALGARY

FACULTY OF SCIENCE

FINAL EXAMINATION

COMPUTER SCIENCE 417

December, 2006

Time: 2 hrs.

Instructions

The exam contains questions totalling 100 points. Answer all questions. This exam is closed book.
1. Given the following datatype for terms:

```haskell
data Term = Var Int
          | Node String [Term]
```

and following datatype for exceptions:

```haskell
data Possibly a = Error String | Value a
```

(a) Explain what a unifier of two terms is.
(b) Explain what the most general unifier of two terms is.
(c) Create an instance of the exception monad for the possibly type.
(d) Write a function to implement the occurs check:

```haskell
check :: Int -> Term -> Possibly(Term)
```

using the Possibly monad to return an informative error.
(e) Write a function to find the most general unifier of two terms.
2. (a) In λ-calculus (with respect to α, β equality) let:

\[ S = \lambda xyz. xz(yz) \]
\[ K = \lambda xy. x \]
\[ I = \lambda x. x \]

Show that
i. \( SKK = I \);
ii. \((SI)(SI)\) does not have a normal form;
iii. \( \lambda xy.y \) can be expressed using K and I.

(b) What is a fixed point combinator? Given an example of a fixed point combinator and show that it has the desired property.

(c) Explain how one may represent binary trees in the λ-calculus.

(d) Assuming that one has a representation of numbers and of their basic functions (such as addition and multiplication) describe how to encode the following recursive program in the λ-calculus:

\[
gdc n m = \begin{cases} 
  n + m & \text{if } n \cdot m = 0 \\
  \text{if } n < m \text{ then } gdc(m-n) n \\
  \text{else if } n > m \text{ then } gdc(m-n) m 
\end{cases}
\]
(e) Which of the following are true? Explain your reasoning.

- It is decidable whether a term in the $\lambda$-calculus is in normal form.
- It is decidable whether a term in the $\lambda$-calculus has a normal form.
- If a term in the $\lambda$-calculus does not have a normal form it is cannot be solvable.
- It is decidable whether a term in the $\lambda$-calculus which is in normal form is solvable.
- It is undecidable whether a term in the $\lambda$-calculus has a head normal form.
- It is even undecidable whether a term in the $\lambda$-calculus, which has a normal form, is equal to true.
- A rewriting system is always terminating.
- A rewriting system in which all critical divergences can be resolved is always confluent.
- In an orthogonal rewriting system a leftmost outermost reduction strategy will always find the normal form if one exists.
- In the simply typed $\lambda$-calculus (without fixed points) one can express all computable functions.
35 marks

3. (a) Given the type judgements in table 1 give the lambda term which corresponds to the following proof (in which $\Gamma := A, A \to B, B \to C$):

$$\frac{\frac{\Gamma \vdash B \to C}{\text{proj}} \quad \frac{\frac{\Gamma \vdash A}{\text{proj}} \quad \frac{\frac{\Gamma \vdash A \to B}{\text{proj}} \quad \frac{\frac{\Gamma \vdash t : Q}{\text{app}}}{\Gamma \vdash (ft) : Q}}{\Gamma \vdash (ft) : Q}}{\Gamma \vdash C}$$

(b) Using the judgements for type inference in table 2:

i. Show that the term, ($\lambda x.xx$), cannot be typed in the simply typed lambda calculus.

ii. Show that $S = \lambda xyz.xz(yz)$ can be typed in the simply typed lambda calculus and provide the type.
Table 2: Type judgements with type equations

\[
\begin{array}{l}
\frac{x : P, \Gamma \vdash x : Q}{\triangleright P = Q} \\
\frac{x : P, \Gamma \vdash t : R}{\Gamma \vdash \lambda x.t : Q} \quad P, R \triangleright Q = P \rightarrow R \\
\frac{\Gamma \vdash f : R \quad \Gamma \vdash t : P}{\Gamma \vdash (ft) : Q} \quad P, R \triangleright R = P \rightarrow Q
\end{array}
\]