logical verification 2006-2007
exercises week 7

Exercise 1.

a. Show that $(B \to (A \to B) \to C) \to B \to C$ is a tautology.

b. Give the type derivation in simply typed λ-calculus corresponding to the proof of 1a.

Exercise 2.

a. Show that $(A \to A \to B) \to A \to B$ is a tautology.

b. Give the type derivation in simply typed λ-calculus corresponding to the proof of 2a.

Exercise 3.

a. Show that the formula $((A \to B \to A) \to B) \to B$ is a tautology of first-order minimal propositional logic.

b. Give the type derivation in simply typed λ-calculus corresponding to the proof of 3a.

Exercise 4. Replace in the following terms the ?'s by simple types, such that we obtain typable λ-terms.

a. $\lambda x{:}?.\,\lambda y{:}?.\,x$

b. $\lambda x{:}?.\,\lambda y{:}?.\,x\,y\,y$

c. $\lambda x{:}?.\,\lambda y{:}?.\,x\,(x\,y)$

d. $\lambda x{:}?.\,\lambda y{:}?.\,\lambda z{:}?.\,x\,(y\,z)$

e. $\lambda x{:}?.\,\lambda y{:}?.\,\lambda z{:}?.\,y\,(\lambda u{:}?.\,x)$

f. $\lambda x{:}?.\,\lambda y{:}?.\,\lambda z{:}?.\,z\,((\lambda u{:}?.\,y)\,x)$

Exercise 5.

a. What is the definition of a detour in a natural deduction proof?

b. Give a proof of $A \to A \to A$ in first-order minimal propositional logic that contains a detour.

c. Give the λ-term that corresponds to the proof of 5b.
Which part corresponds to the detour?
Give the normal form of the λ-term.

Exercise 6. Give some different closed normal forms of type $(A \to A) \to A \to A$.

Exercise 7. Show that Peirce's Law implies double negation. That is, show that $(((A \to \bot) \to A) \to A) \to \neg\neg A \to A$ is a tautology.

Exercise 8.

a. Consider the definition of `natlist` for lists of natural numbers:

```
Inductive natlist : Set :=
| nil : natlist
| cons : nat -> natlist -> natlist.
```

Give the type of `natlist_ind`, which is used to give proofs by induction.

b. Give the definition of an inductive predicate `last_element` such that `(last n l)` means that `n` is the last element of `l`.

Exercise 9.

a. Give the inductive definition of the datatype `natbintree` of binary trees with unlabeled nodes and natural numbers at the leafs.

b. The Coq function for appending two lists is defined as follows:

```
Fixpoint append (l k : natlist) {struct l} : natlist :=
  match l with
    nil => k
  | cons n l' => cons n (append l' k)
  end.
```

In what argument is the recursion? Why is the recursive call (intuitively) safe?

c. Give the definition of a recursive function `flatten : natbintree -> natlist` which flattens a tree into a list that contains the nodes from left to right.

You may use append.

Exercise 10. What is the type of the function that can be extracted from the proof of the following theorem:

```
forall l : natlist,
{l' : natlist | Permutation l l' /\ Sorted l'}.
```

Exercise 11.

a. Give an example of a proof that is incorrect because the side-condition for the introduction rule for $\forall$ is violated.

b. The rule for elimination of an existential quantifier is:

$$\frac{\exists x.\, A \qquad \forall x.\, (A \to B)}{B}\ E\exists$$

What is the side-condition for this rule?

Exercise 12. Show that the following formulas are tautologies of first-order intuitionistic predicate logic.

a. $(\forall x.\, \neg P(x)) \to \neg(\exists x.\, P(x))$

Hint: use the existential quantification elimination rule as early as possible.

b. $\forall x.\, (P(x) \to \neg\forall y.\, (\neg P(y)))$.

c. $(\forall x.\, P(x)) \to \neg\exists y.\, \neg P(y)$.

d. $((\exists x.\, P(x)) \to (\forall y.\, Q(y))) \to \forall z.\, (P(z) \to Q(z))$.