

BU CAS CS 520 (FALL SEMESTER, 2005)  
**PRINCIPLES OF PROGRAMMING LANGUAGES**

## Assignment 4

**Out: Tuesday, 24 October 2005**  
**Due: Wednesday, 02 November 2005**

**Total:** 140 points

Let us define  $List(T)$  as  $\forall X. X \rightarrow (T \rightarrow X \rightarrow X) \rightarrow X$ . Then the two list constructors  $Nil$  and  $Cons$  can be defined as follows.

$$\begin{aligned} Nil &= \Lambda X. \lambda nil : X. \lambda cons : T \rightarrow X \rightarrow X. nil \\ Cons(x : T)(xs : List(T)) &= \Lambda X. \lambda nil : X. \lambda cons : T \rightarrow X \rightarrow X. cons(x)(xs[X])(nil)(cons) \end{aligned}$$

**Exercise 1** (40 points) Given the above list encoding, please implement the *append* and *reverse* functions on lists in System F. Notice that you may not use the fixed point operator in exercise.

**Exercise 2** (60 points) Please encode the following datatype `'a gtree` and its associated constructors `E` and `B` in system F (30 points).

```
datatype 'a gtree = E | B of ('a -> 'a gtree)
```

The type constructor `gtree` can be used to form general tree types. For instance, `bool gtree` can be considered as a type for binary trees; `B (fn (x:bool) => E)` represents a binary tree  $t_1$  whose left and right subtrees are empty; `B (fn (x:bool) => if x then E else B (fn (x: bool) => E))` represents a binary tree  $t_2$  whose left subtree is empty and right subtree is  $t_1$ .

Please implement the following function `leftGtree` in system F (30 points).

```
fun leftGtree E = E
  | leftGtree (B (f)) = f (true)
```

Note that you may not use fixed point operator in your implementation of `leftGtree`.

We can define a type erasure function  $|\cdot|$  as follows, which translates a term in System F into an untyped  $\lambda$ -term.

$$|x| = x \quad |\lambda x : T. t| = \lambda x. |t| \quad |t_1(t_2)| = |t_1|(|t_2|) \quad |\Lambda X. t| = |t| \quad |t[T]| = |t|$$

Notice that for a value  $v$  in System F,  $|v|$  may not necessarily be a value. Therefore, given a term  $t$  in System F,  $|t| \rightarrow^* v_0$  does not necessarily imply that we have  $t \rightarrow^* v$  for some value  $v$  such that  $|v| = v_0$ . To have this property, we can impose a restriction on **(T-Tabs)** by requiring that  $t$  be a value whenever the following rule is applied.

$$\frac{\vec{X}, X; \Gamma \vdash t : T \quad \vec{X} \vdash \Gamma [ctx]}{\vec{X}; \Gamma \vdash \Lambda X. t : \forall X. T} \text{ (T-Tabs)}$$

This restriction is often called *value restriction*.

**Exercise 3** (40 points) Assume  $\mathcal{D} :: \vec{X}; \Gamma \vdash t : T$  is derivable in System F with value restriction and  $|t| \rightarrow u$ . Show that  $t \rightarrow^* t'$  for some term  $t'$  such that  $|t'| = u$ .