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

$\text{Nil} = \Lambda X. \lambda \text{nil} : X. \lambda \text{cons} : T \rightarrow X \rightarrow X. \text{nil}$

$\text{Cons}(x : T)(xs : \text{List}(T)) = \Lambda X. \lambda \text{nil} : X. \lambda \text{cons} : T \rightarrow X \rightarrow X. \text{cons}(x)(xs[|X|](\text{nil})(\text{cons}))$

**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 $'a \text{ gtree}$ and its associated constructors $E$ and $B$ in system F (30 points).

$$\text{datatype } 'a \text{ gtree} = E | B \text{ of } ('a \to 'a \text{ gtree})$$

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

Please implement the following function $\text{leftGtree}$ in system F (30 points).

$$\text{fun leftGtree } E = E \mid \text{leftGtree } (B \text{ of } f) = f \text{ (true)}$$

Note that you may not use fixed point operator in your implementation of $\text{leftGtree}$.

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

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

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.

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

This restriction is often called value restriction.

**Exercise 3** (40 points) Assume $D :: \vec{X} ; \Gamma \vdash t : T$ is derivable in System F with value restriction and $|t| \rightarrow u$. Show that $t \rightarrow^* t'$ for some term $t'$ such that $|t'| = u$. 1