Review
Church encoding of lists

... will not be on the exam. :-)

Briefly, though, here's the intuition:

\[\nu_1; \nu_2; \nu_3; \nu_4 = \lambda s. \lambda z. s (s (s (s z)))\]
Church encoding of lists

... will not be on the exam. :-)

Briefly, though, here’s the intuition:

\[ c_4 = \lambda s. \lambda z. s (s (s (s z))) \]
Church encoding of lists

... will not be on the exam. :-)

Briefly, though, here’s the intuition:

\[ c_4 = \lambda s. \lambda z. s \ (s \ (s \ (s \ z))) \]

\[ [v_1;v_2;v_3;v_4] = \lambda s. \lambda z. s \ v_1 \ (s \ v_2 \ (s \ v_3 \ (s \ v_4 \ z))) \]
Exercise 9.2.2: Show (by drawing derivation trees) that the following terms have the indicated types:

1. \( f : \text{Bool} \rightarrow \text{Bool} \vdash f \ (\text{if} \ false \ \text{then} \ true \ \text{else} \ false) : \text{Bool} \)

2. \( f : \text{Bool} \rightarrow \text{Bool} \vdash \lambda x : \text{Bool}. \ f \ (\text{if} \ x \ \text{then} \ false \ \text{else} \ x) : \text{Bool} \rightarrow \text{Bool} \)
The two typing relations

Question: What is the relation between these two statements?

1. \( t : T \)
2. \( \vdash t : T \)
The two typing relations

Question: What is the relation between these two statements?

1. $t : T$
2. $\Gamma \vdash t : T$

First answer: These two relations are completely different things.

- We are dealing with several different small programming languages, each with its own typing relation (between terms in that language and types in that language).
- For the simple language of numbers and booleans, typing is a binary relation between terms and types ($t : T$).
- For $\lambda \to$, typing is a ternary relation between contexts, terms, and types ($\Gamma \vdash t : T$).

(When the context is empty — because the term has no free variables — we often write $\vdash t : T$ to mean $\emptyset \vdash t : T$.)
Conservative extension

Second answer: The typing relation for $\lambda \rightarrow$ conservatively extends the one for the simple language of numbers and booleans.

- Write “language 1” for the language of numbers and booleans and “language 2” for the simply typed lambda-calculus with base types $\texttt{Nat}$ and $\texttt{Bool}$.
- The terms of language 2 include all the terms of language 1; similarly typing rules.
- Write $t :_{1} T$ for the typing relation of language 1.
- Write $\Gamma \vdash t :_{2} T$ for the typing relation of language 2.
- Theorem: Language 2 conservatively extends language 1: If $t$ is a term of language 1 (involving only booleans, conditions, numbers, and numeric operators) and $T$ is a type of language 1 (either $\texttt{Bool}$ or $\texttt{Nat}$), then $t :_{1} T$ iff $\emptyset \vdash t :_{2} T$. 
Preservation (and Weaking, Permutation, Substitution)
Let’s quickly review the steps in the proof of the progress theorem:

- inversion lemma for typing relation
- canonical forms lemma
- progress theorem
Lemma:

1. If $\Gamma \vdash \text{true} : R$, then $R = \text{Bool}$.
2. If $\Gamma \vdash \text{false} : R$, then $R = \text{Bool}$.
3. If $\Gamma \vdash \text{if } t_1 \text{ then } t_2 \text{ else } t_3 : R$, then $\Gamma \vdash t_1 : \text{Bool}$ and $\Gamma \vdash t_2, t_3 : R$.
4. If $\Gamma \vdash x : R$, then
Inversion

Lemma:

1. If \( \Gamma \vdash \text{true} : R \), then \( R = \text{Bool} \).
2. If \( \Gamma \vdash \text{false} : R \), then \( R = \text{Bool} \).
3. If \( \Gamma \vdash \text{if } t_1 \text{ then } t_2 \text{ else } t_3 : R \), then \( \Gamma \vdash t_1 : \text{Bool} \) and \( \Gamma \vdash t_2, t_3 : R \).
4. If \( \Gamma \vdash x : R \), then \( x : R \in \Gamma \).
5. If \( \Gamma \vdash \lambda x : T_1 . t_2 : R \), then
Lemma:

1. If $\Gamma \vdash true : R$, then $R = \text{Bool}$.
2. If $\Gamma \vdash false : R$, then $R = \text{Bool}$.
3. If $\Gamma \vdash \text{if } t_1 \text{ then } t_2 \text{ else } t_3 : R$, then $\Gamma \vdash t_1 : \text{Bool}$ and $\Gamma \vdash t_2, t_3 : R$.
4. If $\Gamma \vdash x : R$, then $x : R \in \Gamma$.
5. If $\Gamma \vdash \lambda x : T_1 . t_2 : R$, then $R = T_1 \rightarrow R_2$ for some $R_2$ with $\Gamma, x : T_1 \vdash t_2 : R_2$.
6. If $\Gamma \vdash t_1 \ t_2 : R$, then
Inversion

Lemma:

1. If $\Gamma \vdash \text{true} : R$, then $R = \text{Bool}$.
2. If $\Gamma \vdash \text{false} : R$, then $R = \text{Bool}$.
3. If $\Gamma \vdash \text{if } t_1 \text{ then } t_2 \text{ else } t_3 : R$, then $\Gamma \vdash t_1 : \text{Bool}$ and $\Gamma \vdash t_2, t_3 : R$.
4. If $\Gamma \vdash x : R$, then $x : R \in \Gamma$.
5. If $\Gamma \vdash \lambda x : T_1 . t_2 : R$, then $R = T_1 \rightarrow R_2$ for some $R_2$ with $\Gamma, x : T_1 \vdash t_2 : R_2$.
6. If $\Gamma \vdash t_1 ~ t_2 : R$, then there is some type $T_{11}$ such that $\Gamma \vdash t_1 : T_{11} \rightarrow R$ and $\Gamma \vdash t_2 : T_{11}$. 
Lemma:

1. If \( v \) is a value of type \( \text{Bool} \), then \( v \) is either \( \text{true} \) or \( \text{false} \).

2. If \( v \) is a value of type \( T_1 \rightarrow T_2 \), then \( v \) has the form \( \lambda x: T_1. t_2 \).
Canonical Forms

Lemma:

1. If \( v \) is a value of type \( \text{Bool} \), then \( v \) is either \( \text{true} \) or \( \text{false} \).
2. If \( v \) is a value of type \( T_1 \rightarrow T_2 \), then \( v \) has the form \( \lambda x:T_1 . t_2 \).
Progress

*Theorem:* Suppose $t$ is a closed, well-typed term (that is, $\vdash t : T$ for some $T$). Then either $t$ is a value or else there is some $t'$ with $t \rightarrow t'$. 
Preservation

Theorem: If $\Gamma \vdash t : T$ and $t \rightarrow t'$, then $\Gamma \vdash t' : T$.

Steps of proof:
- Weakening
- Permutation
- Substitution preserves types
- Reduction preserves types (i.e., preservation)
Weakening and Permutation

Weakening tells us that we can *add assumptions* to the context without losing any true typing statements.

*Lemma:* If $\Gamma \vdash t : T$ and $x \notin \text{dom}(\Gamma)$, then $\Gamma, x : S \vdash t : T$.

Moreover, the latter derivation has the same depth as the former.

Permutation tells us that the order of assumptions in (the list) $\Gamma$ does not matter.

*Lemma:* If $\Gamma \vdash t : T$ and $\Delta$ is a permutation of $\Gamma$, then $\Delta \vdash t : T$.

Moreover, the latter derivation has the same depth as the former.
Weakening and Permutation

Weakening tells us that we can *add assumptions* to the context without losing any true typing statements.

*Lemma:* If $\Gamma \vdash t : T$ and $x \notin dom(\Gamma)$, then $\Gamma, x:S \vdash t : T$.

Permutation tells us that the order of assumptions in (the list) $\Gamma$ does not matter.

*Lemma:* If $\Gamma \vdash t : T$ and $\Delta$ is a permutation of $\Gamma$, then $\Delta \vdash t : T$. 
Weakening and Permutation

Weakening tells us that we can *add assumptions* to the context without losing any true typing statements.

*Lemma:* If $\Gamma \vdash t : T$ and $x \notin \text{dom}(\Gamma)$, then $\Gamma, x:S \vdash t : T$.

Moreover, the latter derivation has the same depth as the former.

Permutation tells us that the order of assumptions in (the list) $\Gamma$ does not matter.

*Lemma:* If $\Gamma \vdash t : T$ and $\Delta$ is a permutation of $\Gamma$, then $\Delta \vdash t : T$.

Moreover, the latter derivation has the same depth as the former.
Preservation

*Theorem:* If $\Gamma \vdash t : T$ and $t \rightarrow t'$, then $\Gamma \vdash t' : T$.

*Proof:* By induction
Preservation

Theorem: If $\Gamma \vdash t : T$ and $t \rightarrow t'$, then $\Gamma \vdash t' : T$.

Proof: By induction on typing derivations.

Which case is the hard one??
Preservation

Theorem: If $\Gamma \vdash t : T$ and $t \rightarrow t'$, then $\Gamma \vdash t' : T$.

Proof: By induction on typing derivations.
Case $T$-App: Given $t = t_1 \ t_2$

$\Gamma \vdash t_1 : T_{11} \rightarrow T_{12}$
$\Gamma \vdash t_2 : T_{11}$

$T = T_{12}$

Show $\Gamma \vdash t' : T_{12}$
Preservation

Theorem: If $\Gamma \vdash t : T$ and $t \rightarrow t'$, then $\Gamma \vdash t' : T$.

Proof: By induction on typing derivations.
Case $\text{T-APP}$: Given $t = t_1 \ t_2$
\[\Gamma \vdash t_1 : T_{11} \rightarrow T_{12}\]
\[\Gamma \vdash t_2 : T_{11}\]
\[T = T_{12}\]

Show $\Gamma \vdash t' : T_{12}$

By the inversion lemma for evaluation, there are three subcases...
Preservation

*Theorem:* If $\Gamma \vdash t : T$ and $t \rightarrow t'$, then $\Gamma \vdash t' : T$.

*Proof:* By induction on typing derivations.

Case $\text{T-App}$: Given $t = t_1 \ t_2$

$\Gamma \vdash t_1 : T_{11} \rightarrow T_{12}$

$\Gamma \vdash t_2 : T_{11}$

$T = T_{12}$

Show $\Gamma \vdash t' : T_{12}$

By the inversion lemma for evaluation, there are three subcases...

*Subcase:* $t_1 = \lambda x : T_{11} \ . \ t_{12}$

$t_2$ a value $v_2$

$t' = [x \mapsto v_2]t_{12}$
Preservation

**Theorem:** If $\Gamma \vdash t : T$ and $t \rightarrow t'$, then $\Gamma \vdash t' : T$.

**Proof:** By induction on typing derivations.

**Case T-App:** Given $t = t_1 t_2$

$\Gamma \vdash t_1 : T_{11} \rightarrow T_{12}$

$\Gamma \vdash t_2 : T_{11}$

$T = T_{12}$

Show $\Gamma \vdash t' : T_{12}$

By the inversion lemma for evaluation, there are three subcases...

**Subcase:** $t_1 = \lambda x : T_{11}. \ t_{12}$

$t_2$ a value $v_2$

$t' = [x \mapsto v_2] t_{12}$

Uh oh.
Preservation

Theorem: If $\Gamma \vdash t : T$ and $t \rightarrow t'$, then $\Gamma \vdash t' : T$.

Proof: By induction on typing derivations.

Case $T$-App: Given $t = t_1 \ t_2$

$\Gamma \vdash t_1 : T_{11} \rightarrow T_{12}$

$\Gamma \vdash t_2 : T_{11}$

$T = T_{12}$

Show $\Gamma \vdash t' : T_{12}$

By the inversion lemma for evaluation, there are three subcases...

Subcase: $t_1 = \lambda x : T_{11}. \ t_{12}$

$t_2$ a value $v_2$

$t' = [x \mapsto v_2] t_{12}$

Uh oh. What do we need to know to make this case go through??
The “Substitution Lemma”

*Lemma:* If $Γ, x:S ⊢ t : T$ and $Γ ⊢ s : S$, then $Γ ⊢ [x \mapsto s]t : T$.
I.e., “Types are preserved under substitution.”
The “Substitution Lemma”

Lemma: If $\Gamma, x:S \vdash t : T$ and $\Gamma \vdash s : S$, then $\Gamma \vdash [x \mapsto s]t : T$.

Proof: By induction on the depth of a derivation of $\Gamma, x:S \vdash t : T$. Proceed by cases on the final typing rule used in the derivation.
The “Substitution Lemma”

Lemma: If $\Gamma, x:S \vdash t : T$ and $\Gamma \vdash s : S$, then $\Gamma \vdash [x \mapsto s]t : T$.

Proof: By induction on the depth of a derivation of $\Gamma, x:S \vdash t : T$. Proceed by cases on the final typing rule used in the derivation.
The “Substitution Lemma”

Lemma: If $\Gamma, x:S \vdash t : T$ and $\Gamma \vdash s : S$, then $\Gamma \vdash [x \mapsto s]t : T$.

Proof: By induction on the depth of a derivation of $\Gamma, x:S \vdash t : T$. Proceed by cases on the final typing rule used in the derivation.

Case T-App:

$t = t_1 \; t_2$

$\Gamma, x:S \vdash t_1 : T_2 \rightarrow T_1$

$\Gamma, x:S \vdash t_2 : T_2$

$T = T_1$

By the induction hypothesis, $\Gamma \vdash [x \mapsto s]t_1 : T_2 \rightarrow T_1$ and $\Gamma \vdash [x \mapsto s]t_2 : T_2$. By T-App, $\Gamma \vdash [x \mapsto s]t_1 \; [x \mapsto s]t_2 : T$, i.e., $\Gamma \vdash [x \mapsto s](t_1 \; t_2) : T$. 
The “Substitution Lemma”

**Lemma:** If $\Gamma, x:S \vdash t : T$ and $\Gamma \vdash s : S$, then $\Gamma \vdash [x \mapsto s]t : T$.

**Proof:** By induction on the *depth* of a derivation of $\Gamma, x:S \vdash t : T$. Proceed by cases on the final typing rule used in the derivation.

**Case T-VAR:** $t = z$

with $z:T \in (\Gamma, x:S)$

There are two sub-cases to consider, depending on whether $z$ is $x$ or another variable. If $z = x$, then $[x \mapsto s]z = s$. The required result is then $\Gamma \vdash s : S$, which is among the assumptions of the lemma. Otherwise, $[x \mapsto s]z = z$, and the desired result is immediate.
The “Substitution Lemma”

**Lemma:** If $\Gamma, x: S \vdash t : T$ and $\Gamma \vdash s : S$, then $\Gamma \vdash [x \mapsto s]t : T$.

**Proof:** By induction on the depth of a derivation of $\Gamma, x: S \vdash t : T$. Proceed by cases on the final typing rule used in the derivation.

**Case** T-Abs: $t = \lambda y : T_2. t_1$; $T = T_2 \rightarrow T_1$

$\Gamma, x : S, y : T_2 \vdash t_1 : T_1$

By our conventions on choice of bound variable names, we may assume $x \neq y$ and $y \notin FV(s)$. Using permutation on the given subderivation, we obtain $\Gamma, y : T_2, x : S \vdash t_1 : T_1$. Using weakening on the other given derivation ($\Gamma \vdash s : S$), we obtain $\Gamma, y : T_2 \vdash s : S$. Now, by the induction hypothesis, $\Gamma, y : T_2 \vdash [x \mapsto s]t_1 : T_1$. By T-Abs,

$\Gamma \vdash \lambda y : T_2. \ [x \mapsto s]t_1 : T_2 \rightarrow T_1$, i.e. (by the definition of substitution), $\Gamma \vdash [x \mapsto s] \lambda y : T_2. t_1 : T_2 \rightarrow T_1$. 