Submission instructions:
You must submit your solutions electronically (in ascii, postscript, or PDF format). Electronic solutions should be submitted following the same instructions as last time; these can be found at http://www.seas.upenn.edu/~cis500/homework.html. Do not email your solutions to us.

1 Exercise  As quick self-checks, you should have done exercises 8.2.3, 8.3.5, 9.2.1, 9.2.2, 9.2.3, and 9.4.1 in TAPL when you read Chapter 8 and 9. If you did not, do them now.1 Do not turn in anything for this exercise.

2 Exercise  Exercise 8.3.4 in TAPL.

3 Exercise  Exercise 8.3.6 in TAPL.

4 Exercise  Show the derivations of the following typing judgments:
1. if false then 0 else succ 0 : Nat
2. succ (if iszero 0 then succ 0 else pred 0) : Nat

5 Exercise  If we make the following changes to the Arith language, do the following theorems become false or remain true. If they become false, give a counterexample.

The theorems:
• Uniqueness of types (Theorem 8.2.4)
• Progress (Theorem 8.3.2)
• Preservation (Theorem 8.3.3)

The changes to Arith:
1. Say we remove the rule E-PredZero.
2. Say we add the rule: if t₁ then t₂ else t₃ → t₂
3. Say we add the rules
   
   pred true → false
   pred false → true
   
   and the following typing rule:

   \[ \begin{array}{c}
   t : \text{Bool} \\
   \hline
   \text{pred } t : \text{Nat}
   \end{array} \]

4. Say we add the rule

1In general, you should be doing all the one ⋆ exercises when you read TAPL. They are designed to make sure you have a minimal understanding of what you’re reading.
if 0 then \( \mathit{t_2} \) else \( \mathit{t_3} \) \( \rightarrow \) \( \mathit{t_2} \)

and the following typing rule:

\[
\begin{array}{c}
\text{if } \mathit{t_1} \text{ then } \mathit{t_2} \text{ else } \mathit{t_3} : \mathbb{T} \\
\end{array}
\]

5. Say we add the following typing axiom: \( 0 : \mathbb{B} \)

**6 Exercise** Prove the type soundness theorem for Arith.

**Theorem:** If \( \mathit{t} : \mathbb{T} \) and \( \mathit{t} \rightarrow^* \mathit{t'} \) and \( \mathit{t'} \not\rightarrow \), then \( \mathit{t'} \) is a value.

You should prove this theorem by induction on the following definition of multistep evaluation. In your proof, you may use the Preservation and Progress theorems from TAPL (Theorems 8.3.2 and 8.3.3).

\[
\begin{array}{c}
t \rightarrow \mathit{t'} \quad \mathit{t'} \rightarrow^* \mathit{t''} \quad \text{EV-Step} \\
\end{array}
\]

**7 Exercise** The following exercise concerns the big-step semantics for the simply-typed lambda calculus. For simplicity, consider just the language with a base value \( 0 \) of type \( \mathbb{N} \) and functions.

The big-step semantics of this calculus is:

\[
\begin{array}{c}
\text{B-Value} \\
\hline
\mathit{v} \downarrow \mathit{v} \\
\end{array}
\]

\[
\begin{array}{c}
\mathit{t_1} \downarrow \lambda x : \mathbb{T}_1.\mathit{t_{11}} \quad \mathit{t_2} \downarrow \mathit{v_2} \quad [x \mapsto \mathit{v_2}]\mathit{t_{11}} \downarrow \mathit{v} \\
\hline
\mathit{t_1} \mathit{t_2} \downarrow \mathit{v} \quad \text{B-App} \\
\end{array}
\]

1. The following theorem is true

**Theorem:** If \( \emptyset \vdash \mathit{t} : \mathbb{T} \) then \( \mathit{t} \downarrow \mathit{v} \) and \( \emptyset \vdash \mathit{v} : \mathbb{T} \).

but the following proof is flawed. Find the error in the proof below.

Proof by induction on the structure of \( \mathit{t} \).

- Case \( \mathit{t} = 0 \). This case is trivial because \( 0 \downarrow 0 \) by B-VALUE and \( \emptyset \vdash 0 : \mathbb{N} \).
- Case \( \mathit{t} = \mathit{x} \), a variable. This case is trivial because it is not the case that \( \emptyset \vdash \mathit{x} : \mathbb{T} \) for any \( \mathbb{T} \).
- Case \( \mathit{t} = \lambda \mathit{x} : \mathbb{T}_1.\mathit{t_{11}} \), an abstraction. This case is trivial because \( \lambda \mathit{x} : \mathbb{T}_1.\mathit{t_1} \downarrow \lambda \mathit{x} : \mathbb{T}_1.\mathit{t_{11}} \) by B-VALUE and \( \emptyset \vdash \lambda \mathit{x} : \mathbb{T}_1.\mathit{t_{11}} : \mathbb{T} \) by assumption.
- Case \( \mathit{t} = \mathit{t_1} \mathit{t_2} \).
  - By inversion (Lemma 9.3.1), \( \emptyset \vdash \mathit{t_1} : \mathbb{T}_1 \rightarrow \mathbb{T} \) and \( \emptyset \vdash \mathit{t_2} : \mathbb{T}_1 \).
  - By induction \( \mathit{t_1} \downarrow \mathit{v_1} \) and \( \emptyset \vdash \mathit{v_1} : \mathbb{T}_1 \rightarrow \mathbb{T} \).
  - By induction \( \mathit{t_2} \downarrow \mathit{v_2} \) and \( \emptyset \vdash \mathit{v_2} : \mathbb{T}_1 \).
  - By canonical forms (Lemma 9.3.4), \( \mathit{v_1} = \lambda \mathit{x} \mathbb{T}_1.\mathit{t_{11}} \).
  - By inversion (Lemma 9.3.1), \( \mathit{x} : \mathbb{T}_1 \vdash \mathit{t_{11}} : \mathbb{T} \).
  - By substitution (Lemma 9.3.8), \( \emptyset \vdash [\mathit{x} \mapsto \mathit{v_2}]\mathit{t_{11}} : \mathbb{T} \).
  - By induction \( [\mathit{x} \mapsto \mathit{v_2}]\mathit{t_{11}} \downarrow \mathit{v} \) and \( \emptyset \vdash \mathit{v} : \mathbb{T} \).
  - By B-APP, \( \mathit{t_1} \mathit{t_2} \downarrow \mathit{v} \) and by the above \( \emptyset \vdash \mathit{v} : \mathbb{T} \).

2. An easier-to-prove statement of the theorem reads:
Theorem: If $\emptyset \vdash t : T$ and $t \Downarrow v$ then $\emptyset \vdash v : T$.

Prove this theorem.

8 Exercise Suppose we give an implicitly-typed presentation of the simply-typed lambda-calculus. In this case, we define typing judgments that apply to untyped lambda calculus terms (plus booleans).

1. What is the typing rule for abstractions?

2. This style breaks the uniqueness of types property (Theorem 9.3.3). Give a counterexample.

3. State the canonical forms lemma for this calculus (analogous to 9.3.4).

9 Debriefing

1. How many hours did each person in your group spend on this assignment, including time taken to read TAPL?

2. Would you rate it as easy, moderate, or difficult?

3. Did everyone in your study group participate?

4. How deeply do you feel you understand the material it covers (0%–100%)?

If you have any other comments, we would like to hear them; please send them to cis500@cis.upenn.edu.