《程序设计语言的形式语义》期中考试参考答案

Problem 1

Free variables.

$$\mathsf{fv}(x) = \{x\} \qquad \mathsf{fv}(\mathsf{true}) = \mathsf{fv}(\mathsf{false}) = \emptyset$$

$$\begin{aligned}
\mathsf{fv}(\lambda y : \tau.M) &= \mathsf{fv}(M) \setminus \{y\} \\
\mathsf{fv}(M_1\ M_2) &= \mathsf{fv}(M_1) \cup \mathsf{fv}(M_2) \\
\mathsf{fv}(\mathsf{if}\ M_1\ \mathsf{then}\ M_2\ \mathsf{else}\ M_3) &= \mathsf{fv}(M_1) \cup \mathsf{fv}(M_2) \cup \mathsf{fv}(M_3) \\
\mathsf{fv}(M_1 \oplus M_2) &= \mathsf{fv}(M_1) \cup \mathsf{fv}(M_2) \\
\mathsf{fv}(\mathsf{fix}\, M) &= \mathsf{fv}(M)
\end{aligned}$$

Capture-avoiding substitution.

$$\begin{aligned}
x[N/x] &= N \qquad y[N/x] = y \quad (y \neq x) \\
\mathsf{true}[N/x] &= \mathsf{true} \qquad \mathsf{false}[N/x] = \mathsf{false}
\end{aligned}$$

$$\begin{aligned}
(M_1\ M_2)[N/x] &= (M_1[N/x])\,(M_2[N/x]) \\
(\mathsf{if}\ M_1\ \mathsf{then}\ M_2\ \mathsf{else}\ M_3)[N/x] &= \mathsf{if}\ M_1[N/x]\ \mathsf{then}\ M_2[N/x]\ \mathsf{else}\ M_3[N/x] \\
(M_1 \oplus M_2)[N/x] &= (M_1[N/x]) \oplus (M_2[N/x]) \\
(\mathsf{fix}\, M)[N/x] &= \mathsf{fix}\,(M[N/x])
\end{aligned}$$

$$(\lambda y : \tau.M)[N/x] = \begin{cases}
\lambda y : \tau.M & y = x, \\
\lambda y : \tau.(M[N/x]) & y \neq x,\ y \notin \mathsf{fv}(N), \\
\lambda z : \tau.((M[z/y])[N/x]) & y \neq x,\ y \in \mathsf{fv}(N), z \text{ fresh}.
\end{cases}$$

In the last case, z is fresh for M, N, x, y.

Problem 2

(a)

The two complete evaluation sequences are

$$
\begin{aligned}
&(\lambda f : \mathsf{Bool} \to \mathsf{Bool}.\mathsf{true})\big(((\lambda g : \mathsf{Bool} \to \mathsf{Bool}.g)(\lambda x : \mathsf{Bool}.x)) \oplus ((\lambda y : \mathsf{Bool}.y)\mathsf{false})\big) \\
&\quad \longrightarrow (\lambda f : \mathsf{Bool} \to \mathsf{Bool}.\mathsf{true})((\lambda g : \mathsf{Bool} \to \mathsf{Bool}.g)(\lambda x : \mathsf{Bool}.x)) \\
&\quad \longrightarrow (\lambda f : \mathsf{Bool} \to \mathsf{Bool}.\mathsf{true})(\lambda x : \mathsf{Bool}.x) \\
&\quad \longrightarrow \mathsf{true},
\end{aligned}
$$

and

$$
\begin{aligned}
&(\lambda f : \mathsf{Bool} \to \mathsf{Bool}.\mathsf{true})\big(((\lambda g : \mathsf{Bool} \to \mathsf{Bool}.g)(\lambda x : \mathsf{Bool}.x)) \oplus ((\lambda y : \mathsf{Bool}.y)\mathsf{false})\big) \\
&\quad \longrightarrow (\lambda f : \mathsf{Bool} \to \mathsf{Bool}.\mathsf{true})((\lambda y : \mathsf{Bool}.y)\mathsf{false}) \\
&\quad \longrightarrow (\lambda f : \mathsf{Bool} \to \mathsf{Bool}.\mathsf{true})\mathsf{false} \\
&\quad \longrightarrow \mathsf{true}.
\end{aligned}
$$

The choice must be selected before either branch is evaluated. The selected branch then reduces deterministically to a value, enabling the final BETA step. Every evaluation sequence is a prefix of one of these.

(b)

No such type exists. Rules T-VAR and T-ABS force $\lambda g : \mathsf{Bool} \to \mathsf{Bool}.g$ to have type $(\mathsf{Bool} \to \mathsf{Bool}) \to (\mathsf{Bool} \to \mathsf{Bool})$, and $\lambda y : \mathsf{Bool}.y$ to have type $\mathsf{Bool} \to \mathsf{Bool}$. Thus T-APP forces

- $\bullet \vdash (\lambda g : \mathsf{Bool} \to \mathsf{Bool}.g)(\lambda x : \mathsf{Bool}.x) : \mathsf{Bool} \to \mathsf{Bool}$,
- $\bullet \vdash (\lambda y : \mathsf{Bool}.y)\mathsf{false} : \mathsf{Bool}$.

These are the only possible types of the two branches. Rule T-CHOICE requires equal types, so their choice is untypable. Hence T-APP cannot type the whole term.

Problem 3

Variant	Preservation	Progress
1	Holds	Holds
2	Holds	Fails
3	Holds	Fails
4	Fails	Holds
5	Fails	Holds

All counterexamples below are closed.

Variant 2: progress fails.

$$\bullet \vdash \mathsf{true} \oplus \mathsf{false} : \mathsf{Bool}.$$

This term is not a value. Both choice selection rules have been removed, and no rule evaluates inside the choice, so the term cannot reduce.

Variant 3: progress fails.

$$\bullet \vdash \big((\lambda x : \mathsf{Bool}.x) \oplus (\lambda x : \mathsf{Bool}.x)\big)\,\mathsf{true} : \mathsf{Bool}.$$

Both operands of this application are values, but $(\lambda x : \mathsf{Bool}.x) \oplus (\lambda x : \mathsf{Bool}.x)$ is not a lambda and has no reduct. The application is neither a value nor able to reduce.

Variant 4: preservation fails.

$$\bullet \vdash \mathsf{true} \oplus (\mathsf{true}\,\mathsf{true}) : \mathsf{Bool},$$

$$\mathsf{true} \oplus (\mathsf{true}\,\mathsf{true}) \longrightarrow \mathsf{true}\,\mathsf{true}, \qquad \bullet \nvdash \mathsf{true}\,\mathsf{true} : \mathsf{Bool}.$$

The right alternative is unchecked; true cannot have a function type.

Variant 5: preservation fails.

$$\bullet \vdash \mathsf{fix}\,(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.x\,\mathsf{true}) : \mathsf{Bool},$$

$$\mathsf{fix}\,(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.x\,\mathsf{true}) \longrightarrow \big(\mathsf{fix}\,(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.x\,\mathsf{true})\big)\,\mathsf{true},$$

$$\bullet \nvdash \big(\mathsf{fix}\,(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.x\,\mathsf{true})\big)\,\mathsf{true} : \mathsf{Bool}.$$

The operand of fix has type $(\mathsf{Bool} \to \mathsf{Bool}) \to \mathsf{Bool}$, so T-FIX-GENERAL assigns $\mathsf{fix}\,(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.x\,\mathsf{true})$ only type Bool. It cannot replace the function x without losing typing.

Problem 4

(a)

Take $r = \mathsf{true}$ and $s = \mathsf{false}$. The following evaluation reaches a value:

$$
\begin{aligned}
& P_{R_{\mathsf{true}},S_{\mathsf{false}}}\ \mathsf{true} \\
& \quad \longrightarrow \Big(\lambda b : \mathsf{Bool}.\big(\mathsf{if}\ b\ \mathsf{then}\ (P_{R_{\mathsf{true}},S_{\mathsf{false}}}\ \mathsf{true})\ \mathsf{else}\ \mathsf{true}\big) \oplus P_{R_{\mathsf{true}},S_{\mathsf{false}}}\ \mathsf{false}\Big)\ \mathsf{true} \\
& \quad \longrightarrow \big(\mathsf{if}\ \mathsf{true}\ \mathsf{then}\ (P_{R_{\mathsf{true}},S_{\mathsf{false}}}\ \mathsf{true})\ \mathsf{else}\ \mathsf{true}\big) \oplus P_{R_{\mathsf{true}},S_{\mathsf{false}}}\ \mathsf{false} \\
& \quad \longrightarrow P_{R_{\mathsf{true}},S_{\mathsf{false}}}\ \mathsf{false} \\
& \quad \longrightarrow \Big(\lambda b : \mathsf{Bool}.\big(\mathsf{if}\ b\ \mathsf{then}\ (P_{R_{\mathsf{true}},S_{\mathsf{false}}}\ \mathsf{true})\ \mathsf{else}\ \mathsf{true}\big) \oplus P_{R_{\mathsf{true}},S_{\mathsf{false}}}\ \mathsf{false}\Big)\ \mathsf{false} \\
& \quad \longrightarrow \big(\mathsf{if}\ \mathsf{false}\ \mathsf{then}\ (P_{R_{\mathsf{true}},S_{\mathsf{false}}}\ \mathsf{true})\ \mathsf{else}\ \mathsf{true}\big) \oplus P_{R_{\mathsf{true}},S_{\mathsf{false}}}\ \mathsf{false} \\
& \quad \longrightarrow \mathsf{if}\ \mathsf{false}\ \mathsf{then}\ (P_{R_{\mathsf{true}},S_{\mathsf{false}}}\ \mathsf{true})\ \mathsf{else}\ \mathsf{true} \\
& \quad \longrightarrow \mathsf{true}.
\end{aligned}
$$

The first and fourth steps use APP1 with FIX; the second and fifth use BETA. The third, sixth, and seventh use CHOOSE-R, CHOOSE-L, and IF-FALSE, respectively. Thus this is a seven-step evaluation, proving $P_{R_{\mathsf{true}},S_{\mathsf{false}}}\ \mathsf{true} \Downarrow_{\mathrm{may}}$.

Problem 4(a), continued: failure of may termination.

The only immediate reducts of $Q_{R_{\mathsf{true}},S_{\mathsf{false}}}\,\mathsf{true}$ are

$$Q_{R_{\mathsf{true}},S_{\mathsf{false}}}\,\mathsf{true} \longrightarrow \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ (x\,\mathsf{true})\ \mathsf{else}\ \mathsf{true})\big)\,\mathsf{true},$$
$$Q_{R_{\mathsf{true}},S_{\mathsf{false}}}\,\mathsf{true} \longrightarrow \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.x\,\mathsf{false})\big)\,\mathsf{true}.$$

Each step uses APP1, FIX-STEP, and the corresponding choice rule. The first reduct repeats the deterministic three-step cycle

$$\begin{aligned}
&\big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ (x\,\mathsf{true})\ \mathsf{else}\ \mathsf{true})\big)\,\mathsf{true}\\
&\quad\longrightarrow \Big(\lambda b : \mathsf{Bool}.\\
&\qquad\quad \mathsf{if}\ b\ \mathsf{then}\ \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ (x\,\mathsf{true})\ \mathsf{else}\ \mathsf{true})\big)\,\mathsf{true}\ \mathsf{else}\ \mathsf{true}\Big)\,\mathsf{true}\\
&\quad\longrightarrow \mathsf{if}\ \mathsf{true}\ \mathsf{then}\ \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ (x\,\mathsf{true})\ \mathsf{else}\ \mathsf{true})\big)\,\mathsf{true}\ \mathsf{else}\ \mathsf{true}\\
&\quad\longrightarrow \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ (x\,\mathsf{true})\ \mathsf{else}\ \mathsf{true})\big)\,\mathsf{true}.
\end{aligned}$$

These three steps are APP1 with FIX, BETA, and IF-TRUE. The second reduct evaluates as follows:

$$\begin{aligned}
&\big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.x\,\mathsf{false})\big)\,\mathsf{true}\\
&\quad\longrightarrow \Big(\lambda b : \mathsf{Bool}.\big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.x\,\mathsf{false})\big)\,\mathsf{false}\Big)\,\mathsf{true}\\
&\quad\longrightarrow \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.x\,\mathsf{false})\big)\,\mathsf{false}.
\end{aligned}$$

It then repeats the two-step cycle

$$\begin{aligned}
&\big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.x\,\mathsf{false})\big)\,\mathsf{false}\\
&\quad\longrightarrow \Big(\lambda b : \mathsf{Bool}.\big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.x\,\mathsf{false})\big)\,\mathsf{false}\Big)\,\mathsf{false}\\
&\quad\longrightarrow \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.x\,\mathsf{false})\big)\,\mathsf{false}.
\end{aligned}$$

The two steps in each display use APP1 with FIX, followed by BETA. Every intermediate term in these deterministic continuations is a nonvalue. Neither initial choice can reach a value, so $\neg(Q_{R_{\mathsf{true}},S_{\mathsf{false}}}\,\mathsf{true} \Downarrow_{\mathrm{may}})$.

(b)

Both $(r, s) = (\mathsf{false}, \mathsf{true})$ and $(\mathsf{false}, \mathsf{false})$ work. For either choice of s, the only immediate reducts are

$$Q_{R_{\mathsf{false}}, S_s}\ \mathsf{true} \longrightarrow \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ (x\,\mathsf{false})\ \mathsf{else}\ \mathsf{true})\big)\ \mathsf{true},$$

$$Q_{R_{\mathsf{false}}, S_s}\ \mathsf{true} \longrightarrow \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ \mathsf{true}\ \mathsf{else}\ (x\,s))\big)\ \mathsf{true}.$$

Each step uses APP1, FIX-STEP, and the corresponding choice rule. Their complete continuations are deterministic:

$$\begin{aligned}
&\big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ (x\,\mathsf{false})\ \mathsf{else}\ \mathsf{true})\big)\ \mathsf{true}\\
&\quad\longrightarrow \Big(\lambda b : \mathsf{Bool}.\\
&\qquad\ \mathsf{if}\ b\ \mathsf{then}\ \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ (x\,\mathsf{false})\ \mathsf{else}\ \mathsf{true})\big)\ \mathsf{false}\ \mathsf{else}\ \mathsf{true}\Big)\ \mathsf{true}\\
&\quad\longrightarrow \mathsf{if}\ \mathsf{true}\ \mathsf{then}\ \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ (x\,\mathsf{false})\ \mathsf{else}\ \mathsf{true})\big)\ \mathsf{false}\ \mathsf{else}\ \mathsf{true}\\
&\quad\longrightarrow \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ (x\,\mathsf{false})\ \mathsf{else}\ \mathsf{true})\big)\ \mathsf{false}\\
&\quad\longrightarrow \Big(\lambda b : \mathsf{Bool}.\\
&\qquad\ \mathsf{if}\ b\ \mathsf{then}\ \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ (x\,\mathsf{false})\ \mathsf{else}\ \mathsf{true})\big)\ \mathsf{false}\ \mathsf{else}\ \mathsf{true}\Big)\ \mathsf{false}\\
&\quad\longrightarrow \mathsf{if}\ \mathsf{false}\ \mathsf{then}\ \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ (x\,\mathsf{false})\ \mathsf{else}\ \mathsf{true})\big)\ \mathsf{false}\ \mathsf{else}\ \mathsf{true}\\
&\quad\longrightarrow \mathsf{true},
\end{aligned}$$

and

$$\begin{aligned}
&\big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ \mathsf{true}\ \mathsf{else}\ (x\,s))\big)\ \mathsf{true}\\
&\quad\longrightarrow \Big(\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ \mathsf{true}\ \mathsf{else}\ \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ \mathsf{true}\ \mathsf{else}\ (x\,s))\big)\ s\Big)\ \mathsf{true}\\
&\quad\longrightarrow \mathsf{if}\ \mathsf{true}\ \mathsf{then}\ \mathsf{true}\ \mathsf{else}\ \big(\mathsf{fix}(\lambda x : \mathsf{Bool} \to \mathsf{Bool}.\lambda b : \mathsf{Bool}.\mathsf{if}\ b\ \mathsf{then}\ \mathsf{true}\ \mathsf{else}\ (x\,s))\big)\ s\\
&\quad\longrightarrow \mathsf{true}.
\end{aligned}$$

In each unfolding, APP1 with FIX is followed by BETA and the applicable conditional rule. Every intermediate term has exactly one successor. Starting with the MUST-VALUE derivation of $\mathsf{true} \Downarrow_{\mathrm{must}}$, apply MUST-STEP backwards along each individual step: six applications for the first reduct and three for the second. Both premises hold at every application because the next step exists and its result is the only successor. Thus both immediate reducts must terminate. One further application of MUST-STEP, covering these two reducts, gives

$$Q_{R_{\mathsf{false}}, S_s}\ \mathsf{true} \Downarrow_{\mathrm{must}}.$$

Problem 4(b), continued: failure of must termination.

For the divergent evaluation, we have

$$
\begin{aligned}
&P_{R_{\mathsf{false}},S_s}\ \mathsf{true}\\
&\quad\longrightarrow \Big(\lambda b : \mathsf{Bool}.\big(\mathsf{if}\ b\ \mathsf{then}\ (P_{R_{\mathsf{false}},S_s}\ \mathsf{false})\ \mathsf{else}\ \mathsf{true}\big) \oplus \big(\mathsf{if}\ b\ \mathsf{then}\ \mathsf{true}\ \mathsf{else}\ (P_{R_{\mathsf{false}},S_s}\ s)\big)\Big)\ \mathsf{true}\\
&\quad\longrightarrow \big(\mathsf{if}\ \mathsf{true}\ \mathsf{then}\ (P_{R_{\mathsf{false}},S_s}\ \mathsf{false})\ \mathsf{else}\ \mathsf{true}\big) \oplus \big(\mathsf{if}\ \mathsf{true}\ \mathsf{then}\ \mathsf{true}\ \mathsf{else}\ (P_{R_{\mathsf{false}},S_s}\ s)\big)\\
&\quad\longrightarrow \mathsf{if}\ \mathsf{true}\ \mathsf{then}\ (P_{R_{\mathsf{false}},S_s}\ \mathsf{false})\ \mathsf{else}\ \mathsf{true}\\
&\quad\longrightarrow P_{R_{\mathsf{false}},S_s}\ \mathsf{false},
\end{aligned}
$$

and

$$
\begin{aligned}
&P_{R_{\mathsf{false}},S_s}\ \mathsf{false}\\
&\quad\longrightarrow \Big(\lambda b : \mathsf{Bool}.\big(\mathsf{if}\ b\ \mathsf{then}\ (P_{R_{\mathsf{false}},S_s}\ \mathsf{false})\ \mathsf{else}\ \mathsf{true}\big) \oplus \big(\mathsf{if}\ b\ \mathsf{then}\ \mathsf{true}\ \mathsf{else}\ (P_{R_{\mathsf{false}},S_s}\ s)\big)\Big)\ \mathsf{false}\\
&\quad\longrightarrow \big(\mathsf{if}\ \mathsf{false}\ \mathsf{then}\ (P_{R_{\mathsf{false}},S_s}\ \mathsf{false})\ \mathsf{else}\ \mathsf{true}\big) \oplus \big(\mathsf{if}\ \mathsf{false}\ \mathsf{then}\ \mathsf{true}\ \mathsf{else}\ (P_{R_{\mathsf{false}},S_s}\ s)\big)\\
&\quad\longrightarrow \mathsf{if}\ \mathsf{false}\ \mathsf{then}\ \mathsf{true}\ \mathsf{else}\ (P_{R_{\mathsf{false}},S_s}\ s)\\
&\quad\longrightarrow P_{R_{\mathsf{false}},S_s}\ s.
\end{aligned}
$$

In each display, the first step uses APP1 with FIX, and the second uses BETA. The next two steps select the left alternative and apply IF-TRUE in the first display, or select the right alternative and apply IF-FALSE in the second. If $s = \mathsf{true}$, these segments form an eight-step cycle at $P_{R_{\mathsf{false}},S_{\mathsf{true}}}\ \mathsf{true}$. If $s = \mathsf{false}$, the first segment reaches $P_{R_{\mathsf{false}},S_{\mathsf{false}}}\ \mathsf{false}$, where the second segment is a four-step cycle. In either case, following this infinite evaluation through a finite must-termination derivation would force an infinite descent in derivation height. Hence $\neg(P_{R_{\mathsf{false}},S_s}\ \mathsf{true} \Downarrow_{\mathrm{must}})$.

(c)

No. The closed term $\mathsf{true} \oplus (\mathsf{true}\,\mathsf{true})$ is a counterexample. Its only complete evaluation sequences are

$$\mathsf{true} \oplus (\mathsf{true}\,\mathsf{true}) \longrightarrow \mathsf{true}, \qquad \mathsf{true} \oplus (\mathsf{true}\,\mathsf{true}) \longrightarrow \mathsf{true}\,\mathsf{true}.$$

Thus $(\mathsf{true} \oplus (\mathsf{true}\,\mathsf{true})) \Downarrow_{\mathrm{may}}$, and no evaluation sequence is infinite. The term $\mathsf{true}\,\mathsf{true}$ is a nonvalue with no evaluation step, so neither must-termination rule applies to it. Since $\mathsf{true} \oplus (\mathsf{true}\,\mathsf{true})$ is not a value and has this successor, the universal premise of MUST-STEP cannot hold. Hence $\neg((\mathsf{true} \oplus (\mathsf{true}\,\mathsf{true})) \Downarrow_{\mathrm{must}})$.