QUESTION 1 Let \( A \) be a formula
\[
((\neg a \Rightarrow \neg b) \Rightarrow c)
\]
and let \( v \) be such that
\[
v(a) = T, \ v(b) = F, v(c) = F.
\]
Evaluate \( A', B_1, ..., B_n \) as defined by the definition:
Let \( A \) be a formula and \( b_1, b_2, ..., b_n \) be all propositional variables that occur in \( A \). Let \( v \) be variable assignment \( v : VAR \rightarrow \{T, F\} \). We define, for \( A, b_1, b_2, ..., b_n \) and \( v \) a corresponding formulas \( A', B_1, B_2, ..., B_n \) as follows: (for \( i = 1, 2, ..., n \))
\[
A' = \begin{cases} 
A & \text{if } v^*(A) = T \\
\neg A & \text{if } v^*(A) = F 
\end{cases}
\]
\[
B_i = \begin{cases} 
b_i & \text{if } v(b_i) = T \\
\neg b_i & \text{if } v(b_i) = F
\end{cases}
\]
Solution \( b_1 = a, b_2 = b, b_3 = c \), and \( v^*(A) = v^*(a \Rightarrow \neg b) = v(a) \Rightarrow \neg v(b) = T \Rightarrow \neg F = T \). The corresponding \( A', B_1, B_2, B_3 \) are:
\[
A' = A \quad (\text{as } v^*(A) = T),
\]
\[
B_1 = a \quad (\text{as } v(a) = T),
\]
\[
B_2 = \neg b \quad (\text{as } v(b) = F),
\]
\[
B_3 = \neg c \quad (\text{as } v(c) = F).
\]

QUESTION 2 Let \( GL \) be the Gentzen style proof system for classical logic defined in chapter 10. Prove, by constructing a proper decomposition tree that
\[
(1) \vdash_{GL}((\neg a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)).
\]
Solution: By definition we have that
\[
\vdash_{GL}((\neg a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)) \text{ if and only if } \vdash_{GL} \rightarrow ((\neg a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)).
\]
We construct the decomposition tree for \( \rightarrow A \) as follows.
\[
\begin{array}{c}
T \rightarrow A \\
\rightarrow ((\neg a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)) \\
| \ (\rightarrow \Rightarrow) \\
(\neg a \Rightarrow b) \rightarrow (\neg b \Rightarrow a)
\end{array}
\]
\[ \neg b, (\neg a \Rightarrow b) \quad \text{---} \quad a \]
\[ (\neg a \Rightarrow b) \quad \text{---} \quad b, a \]
\[ \text{axiom} \]
\[ b \quad \text{---} \quad b, a \]
\[ a \quad \text{---} \quad b, a \]
\[ \text{axiom} \]

All leaves of the tree \( T_{\rightarrow A} \) are axioms, hence we have found the proof of \( A \) in \( \text{GL} \).

(2) \( \not \vdash_{\text{GL}} ((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)) \).

**Solution:** Observe that for any formula \( A \), its decomposition tree \( T_{\rightarrow A} \) in \( \text{GL} \) is not unique. Hence when constructing decomposition trees we have to cover all possible cases.

We construct the decomposition tree for \( \neg \rightarrow A \) as follows.

\( T_{1 \rightarrow A} \)

\[ \neg \rightarrow ((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)) \]
\[ (\text{first of two choices}) \]
\[ (a \Rightarrow b) \quad \text{---} \quad (\neg b \Rightarrow a) \]
\[ (\text{first of two choices}) \]
\[ \neg b, (a \Rightarrow b) \quad \text{---} \quad a \]
\[ (\text{one choice}) \]
\[(a \Rightarrow b) \rightarrow b, a\]
\[
\bigwedge \left( \Rightarrow \rightarrow \right) \quad (one \ choice)\]

\[
\rightarrow a, b, a \quad b \rightarrow b, a\]

*non-axiom*

*axiom*

The tree contains a non-axiom leaf \( \rightarrow a, b, a \), hence it is not a proof of \( ((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)) \) in \( \text{GL} \).

We have only one more tree to construct. Here it is.

\[T_{2 \rightarrow A}\]

\[
\rightarrow ((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a))
\]
\[
\mid (\neg \Rightarrow)
\]

*(one choice)*

\[
(a \Rightarrow b) \rightarrow (\neg b \Rightarrow a)
\]
\[
\bigwedge \left( \Rightarrow \rightarrow \right) \quad (second \ of \ two \ choices)\]

\[
\rightarrow (\neg b \Rightarrow a), a \quad b \rightarrow (\neg b \Rightarrow a)
\]

\[
\mid (\rightarrow \Rightarrow)
\]

*(one choice)*

\[
\neg b \rightarrow a, a
\]
\[
\mid (\neg \rightarrow)
\]

*(one choice)*

\[
\rightarrow a, b
\]

*non-axiom*

\[
b \rightarrow a, b
\]

*axiom*

All possible trees end with an non-axiom leave whet proves that \( \not \vdash_{\text{GL}} ((a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)) \).

**QUESTION 3**  Show that tree below do not constitute a proof in \( \text{GL} \) defined in chapter 10.
The tree is not a proof in \( GL \) because a rule corresponding to the decomposition step below does not exists in it.

\[
\neg \neg (\neg a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)
\]

\[
\neg (\neg a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)
\]

\[
(\neg a \Rightarrow b) \Rightarrow (\neg b \Rightarrow a)
\]

\[
(\neg a \Rightarrow b), \neg b \Rightarrow a
\]

\[
(\neg a \Rightarrow b) \Rightarrow b, a
\]

\[
\neg a, b, a
\]

\[
\neg \neg \neg a, b, a
\]

\[
b \Rightarrow b, a
\]

\[
\neg \neg a \Rightarrow b, a
\]

\[
a \Rightarrow b, a
\]

\[
\neg \neg a \Rightarrow b, a
\]

\[
\neg \neg a \Rightarrow b, a
\]

The tree is a proof in some system \( GL1 \) that has all the rules of \( GL \) except its \( \neg \Rightarrow \). This rule has to be replaced by the rule:

\[
(\neg \Rightarrow)_1 \Gamma, \Gamma' \Rightarrow \Delta, A, \Delta'
\]

\[
\Gamma, \neg A, \Gamma' \Rightarrow \Delta, \Delta'
\]
Observe that the completeness of the system \(GL\) may not imply the completeness of \(GL1\), i.e. we don’t know if the new system \(GL1\) is complete (in fact, it is!).

**QUESTION 4** Let \(GL\) be the Gentzen style proof system for classical logic defined in chapter 10. Prove, by constructing a counter-model defined by a proper decomposition tree that

\[
\not\models ((a \Rightarrow (\neg b \land a)) \Rightarrow (\neg b \Rightarrow (a \cup b))).
\]

**Solution**

\[
\begin{align*}
T \rightarrow A \\
\rightarrow ((a \Rightarrow (\neg b \land a)) \Rightarrow (\neg b \Rightarrow (a \cup b))) \\
\text{one of two choices} \\
\neg b, (a \Rightarrow (\neg b \land a)) \rightarrow (a \cup b) \\
\text{one of two choices} \\
\neg b, (a \Rightarrow (\neg b \land a)) \rightarrow a, b \\
\land (\Rightarrow \rightarrow)
\end{align*}
\]

\[
\begin{align*}
\rightarrow a, b, a, b \\
\neg \rightarrow
\end{align*}
\]

\[
\begin{align*}
\neg (\neg \land \rightarrow) \\
\neg a, a, a, b \\
\text{non-axiom}
\end{align*}
\]

\[
\begin{align*}
\neg (\Rightarrow \land \rightarrow) \\
\neg b, a, a, b \\
\text{axiom}
\end{align*}
\]

The counter-model model determined by the non-axiom leaf \(\rightarrow a, b, a, b\) is any truth assignment that evaluates it to \(F\).

Observe that (we use a shorthand notation) \(\rightarrow a, b, a, b\) represents semantically \(T \rightarrow a, b, a, b\) and hence \(\rightarrow a, b, a, b = F\) iff \(T \rightarrow a, b, a, b = F\), what happens only if \(T \Rightarrow a \cup b \cup a \cup b = F\), i.e. when \(a = F\) and \(b = F\).
GENERAL REMARK  We are using the word "PROOF" in two distinct senses.

In the first sense, we use it as a *formal proof* within a fixed proof system, namely LI and is represented as a proof tree, or sequence of expressions of the language $\mathcal{L}$ of LI.

In the second sense, it also designates certain sequences of sentences of English language (supplemented by some technical terms, if needed) that are supposed to serve as an argument justifying some assertions about the language $\mathcal{L}$, or proof system based on it.

In general, the language we are studying, in this case $\mathcal{L}$, is called an *OBJECT LANGUAGE*. The language in which we formulate and prove the results about the object language is called the *META-LANGUAGE*. The metalanguage might also be formalized and made the object of study, which we would carry in a *meta-metalanguage*.

We use English as our not formalized metalanguage, although, we use only a mathematically weak portion of the English language. The contrast between the language and metalanguage is also present in study for example, a foreign language. In French study class, French is the object language, while the metalanguage, the language we use, is English.

The distinction between *proof* and *meta-proof*, i.e. a proof in the metalanguage, is now clear. We construct (in the metalanguage) a decomposition tree which is a *formal proof* in the object language. By doing so, we prove in the metalanguage, that the proof in the object language exists. Such proof is called a *meta-proof*, and the fact thus proved is called a *meta-theorem*.

QUESTION 5  Consider a system RS$_2$ obtained from RS by changing the sequence $\Gamma'$ into $\Gamma$ and $\Delta$ into $\Delta'$ in all of the rules of inference of RS.

1. Construct a decomposition tree of $A$ from the QUESTION 2 in RS$_2$.

   **Solution** construction is similar to RS, except that now we traverse sequences from RIGHT to left.

2. Define in your own words, for any $A$, the decomposition tree $T_A$ in RS$_2$.

   **Solution** the definition is similar as in the book.