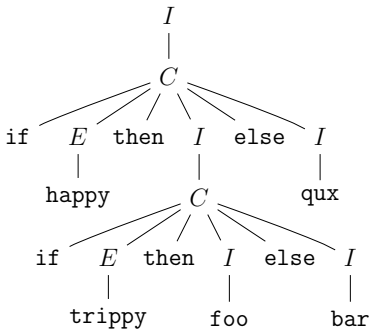
TD6 - Grammaires (2) (éléments de réponses)

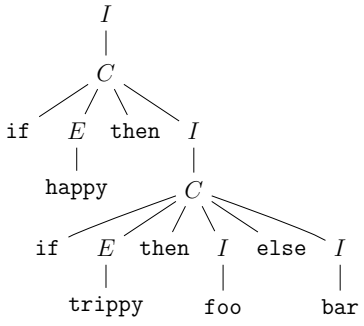
Exercice 1
Exercice 2
Exercice 3
Exercice 4
Exercice 5

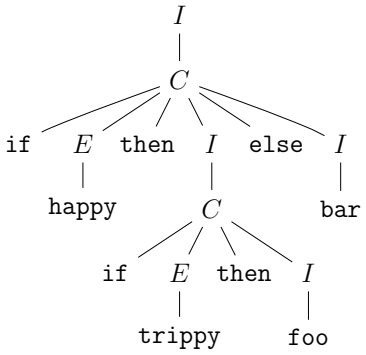
Question 1. L'arbre de dérivation de `if happy then if trippy then foo else bar else qux` est la suivant.



Il est le seul possible car une fois acquis que les deux `if` comportent chacun un `else`, il se construit ensuite en descendant de façon unique. L'instruction est forcément une condition qui s'analyse en `if E then I else I` de façon unique et chacun des morceaux s'analyse de nouveau de façon unique.

Question 2. Un arbre de dérivation possible (figure de gauche ci-dessous) consiste à associer le `else bar` avec `if trippy then foo`. Un autre arbre possible (figure de droite ci-dessous) consiste à associer le `else bar` avec `if happy then ...`.





La grammaire présentée est ambiguë.

Question 3. Pour forcer la première interprétation (le `else bar` se rapporte au `if trippy`), on peut écrire : `if happy then begin if trippy then foo else bar end`.
Pour forcer la seconde interprétation (le `else bar` se rapporte au `if happy`), on peut écrire : `if happy then begin if trippy then foo end else bar`.

Question 4. Pour forcer la première interprétation (le `else` se rapporte au `if` le plus proche possible), on peut modifier la grammaire comme suit.

$$
\begin{aligned}
\textit{Instr} &\to \texttt{foo} \mid \texttt{bar} \mid \texttt{qux} \mid \textit{Cond} \mid \texttt{begin}\ \textit{InstrList}\ \texttt{end} \\
\textit{InstrNoSC} &\to \texttt{foo} \mid \texttt{bar} \mid \texttt{qux} \mid \textit{CondNoSC} \mid \texttt{begin}\ \textit{InstrList}\ \texttt{end} \\
\textit{Cond} &\to \texttt{if}\ \textit{Expr}\ \texttt{then}\ \textit{InstrNoSC}\ \texttt{else}\ \textit{Instr} \mid \texttt{if}\ \textit{Expr}\ \texttt{then}\ \textit{Instr} \\
\textit{CondNoSC} &\to \texttt{if}\ \textit{Expr}\ \texttt{then}\ \textit{InstrNoSC}\ \texttt{else}\ \textit{InstrNoSC} \\
\textit{InstrList} &\to \textit{Instr} \mid \textit{Instr}\ \textit{InstrList} \\
\textit{Expr} &\to \texttt{true} \mid \texttt{false} \mid \texttt{happy} \mid \texttt{trippy}
\end{aligned}
$$

L'idée est d'obliger une instruction conditionnelle qui apparaîtrait après le `then` d'une conditionnelle complète à être elle-même complète (elle ne peut pas être courte car alors le `else` devrait se rattacher à elle), et ce, récursivement. On peut montrer que la grammaire ci-dessus est inambiguë et faiblement équivalente à celle de départ.

On peut aussi fabriquer une grammaire inambiguë, faiblement équivalente à celle de départ, qui force l'autre interprétation (le `else` se rapporte au `if` le plus lointain possible), mais c'est nettement plus complexe (l'idée générale pour apparier un `else` avec un `if... else` dans cette logique est de demander que soit le `else` n'est suivi d'aucun autre `else`, soit toute instruction conditionnelle entre le `then` et le `else` est elle-même complète).