b. You are told that, in this grammar, ?, ! and @ are right-associative binary operators and are given in order of ascending precedence (i.e., ? has the lowest precedence, followed by ! and @ has the highest). Unary operators < and > have even higher precedence. Transform the CFG given above by eliminating ambiguity (if needed) and left recursion, as well as performing left factoring. List the resulting grammar. Is this grammar suitable for top-down parsing?

This is handled easiest by breaking the process down into steps, as follows:

1. The obvious source of the the ambiguity is the production \( E \rightarrow EBE \), since you can expand either the left or right \( E \) to generate the same strings. To eliminate the ambiguity, allow for the recursion to be on one side or the other of \( EBE \), but not both. Since we want to avoid left recursion, make this production a right recursive rule instead, to yield the following grammar:

\[
E \rightarrow WBE \\
W \rightarrow UV \mid V \mid [E] \\
V \rightarrow a \mid b \\
U \rightarrow < \mid > \\
B \rightarrow ? \mid ! \mid @
\]

Note that we collected the right-hand sides for \( E \) that stop the recursion and made them the right-hand sides of a new non-terminal, \( W \). This is going to make things a lot easier in the steps to come.

Now, the only parse tree for a string such as the one given for the solution to part (a), is:

2. The previous step eliminated both ambiguity and left recursion. The next thing to deal with is the precedence of operators. The operators with the highest precedence should appear lowest in the parse tree, and the operator with the lowest precedence should be forced to appear highest in the tree.
The operator with the lowest precedence is $. To force it to be highest in the tree, we make sure that it is generated first when expanding $E$, followed by ! and then @. Finally, we want to generate the operators produced by the non-terminal $U$, along with the variables ($a$ and $b$) and bracketed expressions ($[E]$), which are already the right-hand sides for non-terminal $W$. The resulting grammar is:

\[
\begin{align*}
E & \rightarrow TE' \\
E' & \rightarrow ?E \mid \epsilon \\
T & \rightarrow FT' \\
T' & \rightarrow !T \mid \epsilon \\
F & \rightarrow WF' \\
F' & \rightarrow @F \mid \epsilon \\
W & \rightarrow UV \mid V \mid [E] \\
V & \rightarrow a \mid b \\
U & \rightarrow < \mid >
\end{align*}
\]