1) Consider the grammar

\[
E \rightarrow E \text{ OR } T \mid T \\
T \rightarrow T \text{ and } F \mid F \\
F \rightarrow \text{ not } F \mid (E) \mid \text{true} \mid \text{false}
\]

a) Construct a parse tree for sentence

\[
\text{not ( true or false)}
\]
b) THE ANSWER I EXPECTED

This generates all valid Boolean expressions with NOT, AND, OR, (, ).

But I will show you how to prove this (one direction only).

Answer: First show that the languages generates Valid Boolean expressions with NOT, AND, OR, (, ).
Second show that, given any Boolean expression this language can generate it.

FOCUS ON RIGHT MOST DERIVATIONS.

Proof: (First). Induction on the length of the strings derived.

Induction Hypothesis P(n): The grammar generates a string of length <= n then its a Boolean expressions.

Let n = 1 (base case)
The only strings on length 1 are true and false. Hence base case is true.

Let n > 1 assume P(n). Now consider a string of length n+1.

case 1. Either the string ends with a OR true or OR false. If this happens then the derivation was due to the production E --> E OR T; now clearly the string generated by E is length (n-1), by Induction hypothesis the string of length(n+1) is a Boolean expression

case 2. The string end with AND true or AND false. Then the only way this could happen is E--> T--> T AND F
Again since T derives strings of length (n-1) and E-->T we are done.

The only way this can happen is 
F--> ( E ). clearly, E derives strings 
of length n-1, we are done.

Now to show the second part, use induction. That is given a 
string which is Boolean expression, show by induction on 
the length of the string you can derive it.