To accommodate floating constants add the following:

START → A → 0-9

0-9 → t, - → 0-9

0-9 → t, F, i, t → H

Decimal float constant

F → C → P, P → 0-9

0-9 → t, F, i, t → H

Hex floating constant
(2) a) The grammar's ambiguity can be proven by finding an example of a sentence generated by the grammar in two or more distinct derivations (parse trees).

Consider the statement

\[
\text{if (expr) then}
\]

\[
\text{if (expr) then \textit{<stmt>}}
\]

\[
\text{else}
\]

\[
\textit{<stmt>}
\]

The following distinct parse trees are generated by the grammar:

```
<stmt>  
  |  
  <if_stmt>  
  / 
  |  
  if ( expr ) then <stmt>  
  / 
  |  
  <if_stmt>  
  / 
  |  
  if ( expr ) then <stmt>  
  / 
  |  
  else <stmt>
```

Parse tree I
This well known ambiguity is commonly called the dangling else problem. PASCAL resolves the ambiguity by associating the "else" with the closest unmatched "then". Therefore, these languages prefer parse tree II.

We now provide a new grammar that only allows parse tree I.
\[
\text{start} \rightarrow \text{balance\_start} \\
\vspace{0.5cm}
| \text{unbalance\_start} \\
\vspace{0.5cm}
\text{balance\_start} \rightarrow \text{if } (<\text{expr}> \text{) then } \text{balance\_start} \text{ else } \text{balance\_start} \\
\vspace{1cm}
| \text{black} \\
\vspace{1cm}
| ... \\
\vspace{1cm}
\text{unbalance\_start} \rightarrow \text{if } (\text{<expr>}) \text{ then } \text{<stmt>} \\
\vspace{1cm}
| \text{if } (\text{<expr>}) \text{ then } \text{balance\_stmt} \\
| \text{else } \text{unbalance\_stmt} \]
3) Sebesta 3.13

\( S \Rightarrow a S b \mid ab \)

4) Sebesta 3.14

(a) Parse tree for \( aabb \)

(b) Parse tree for \( aaaaabbbbb \)