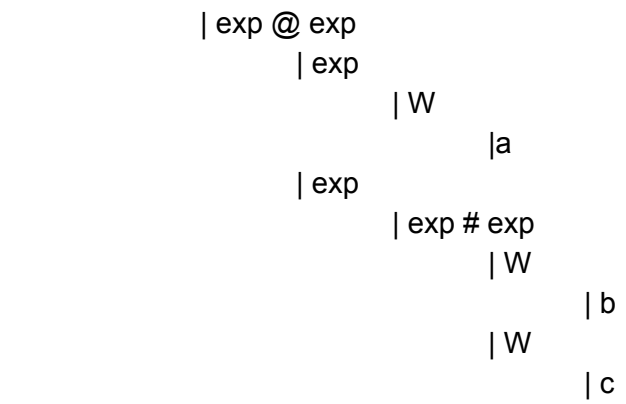
1. The difference between a high and low level programming language is the level of abstraction. A high level language supports data abstraction and structured programming; things like loops, if statements, and classes. A low level language wouldn't have any of these.

2. '(a | z)[a - z | A - Z]*'

3.
    a. It is not LL1 parsable because multiple parse trees can be created that result in the same output. For instance, if the input is "exp @ exp @ exp".
    b. Yes, because we can make a parse tree for it.



    c. Make a separate expression that contains the @ operator ensuring that an @ operator cannot be placed into the parse tree after the # symbol.

4. Yes, it is an LL1 grammar. A grammar cannot have a terminal without a nonterminal before it. The grammar given has only one nonterminal. Both lines from that nonterminal only lead to terminals. Therefore our parse tree only has a height of one in any scenario. There's no other node to be considered ambiguous to sen and there's no looping structure.