20 for each question

1. What is the difference between a lexeme and a token?

A lexeme is a group of characters that comprise the smallest unit of a program language, or a word.

A token is a category of a lexeme.

For example, a + could be a lexeme in a language. The token for the + lexeme could be plus\_op, which is simply a name describing the category for the lexeme +. Another example could be a category (or token) called “id” that is used to represent all variable names like var1, var2.

1. What are two grammar issues that are difficult for top-down parsers to deal with?

Left-recursion (when a rule calls itself recursively either directly or indirectly through another rule), and pairwise disjointness (this is when multiple definitions of a single rule start with the same symbol (terminal or non-terminal).

1. Show all steps of a left-most derivation for the following sentence given the grammar below:

E -> E + T | T

T - > id

id + id + id

E => E + T

=> E + T + T

=> T + T + T

=> id + T + T

=> id + id + T

=> id + id + id

They don’t “have” to underline the symbol that is being derived like I did. But the derivation should look the same.

1. What is handle pruning?

In a bottom-up parse, handle pruning refers to the act of finding the “handle” or root(lhs) of a rhs or symbol, then replacing the rhs with the appropriate symbol. This describes the way that a bottom-up parser reduces a “sentence” back to the starting symbol in a grammar.

1. In what order does a bottom-up parser parse a sentence (sentence being the final sentencial form of a derivation)?

From the bottom-up and it produces the reverse of a right-most derivation.