1. **Explain the two grammar characteristics that prohibit them from being used as the basis for a top-down parser.**

* The Left Recursion Problem
* Lack of pairwise disjointness

1. **What is the difference between direct and indirect left recursion?**

Direct recursion occurs when a method invokes itself but indirect recursion occurs when a method invokes another method, eventually resulting in the original method being invoked again.

1. **What is the FIRST set for a given grammar and sentential form?**

FIRST(a) = {a | a =>\* ab }

             (If a =>\* e, e is in FIRST(a))

1. **Describe the pairwise disjointness test.**

For each nonterminal, A, in the grammar that has more than one RHS, for each pair of rules, A -> ai and A -> aj, it must be true that

         FIRST(ai) ⋂ FIRST(aj) = f

1. **What is left factoring?**

It is used to remove the common left factor to resolve the lack of pairwise disjointness problem.

1. **What is a phrase of a sentential form?**

* It is the string of all of the leaves of the partial parse tree that is rooted at one particular internal node of the whole parse tree.

1. **What is a simple phrase of a sentential form?**

* A simple phrase is just a phrase that takes a single derivation step from its root nonterminal node.

1. **What is the handle of a sentential form?**

The handle of any rightmost sentential form is its leftmost simple phrase.

1. **What is the mathematical machine on which both top-down and bottom-up parsers are based?**

PDA (PushDown Automation) machine

1. **How do PDAs work?**

It scans strings of symbols from left to right and uses a pushdown stack as its memory.

1. **Describe three advantages of LR parsers.**

* They will work for nearly all grammars that describe programming languages.
* They work on a larger class of grammars than other bottom-up algorithms but are as efficient as any other bottom-up parser.
* They can detect syntax errors as soon as it is possible

1. **What was Knuth’s insight in developing the LR parsing technique?**

A bottom-up parser could use the entire history of the parse, up to the current point, to make parsing decisions.

1. **Describe the purpose of the ACTION table of an LR parser.**

The ACTION table specifies the action of the parser, given the parser state and the next token.

1. **Describe the purpose of the GOTO table of an LR parser.**

The GOTO table specifies which state to put on top of the parse stack after a reduction action is done.

1. **Is left recursion a problem for LR parsers?**

No, many left recursion grammars are LR, but none are LL.