Chapter 4: Lexical and Syntax Analysis

1. Differentiate between top-down and bottom-up parsing. What is left factoring?
Ans:

Top-Down Parsing	Bottom-Up Parsing
In Top-Down Parsing, the parse tree is built from the root downward to the leaves.	In Bottom-Up Parsing, the parse tree is built from the leaves upward to the root.
This parsing technique uses Left Most Derivation.	This parsing technique uses Right Most Derivation.
A top-down parser can be easily structured and formed.	It is difficult to produce a bottom-up parser.

Left factoring: Left factoring is a useful grammar transformation used in parsing. It removes the common left factor that appears in two productions of the same non-terminal.

2. What is Left Recursion? Define lexeme and token.
Ans:

Left Recursion: A production of grammar is said to have left recursion if the leftmost variable of its RHS is same as variable of its LHS.

Lexeme: A lexeme is the lowest level syntactic unit of a language (e.g., *, sum, begin)
Token: A token is a category of lexemes (e.g., identifier)

3. Explain the three reasons why lexical analysis is separated from syntax analysis.
Ans:
There are three reasons why lexical analysis is separated from syntax analysis:
- **Simplicity** - less complex approaches can be used for lexical analysis; separating them simplifies the parser
- **Efficiency** - separation allows optimization of the lexical analyzer
- **Portability** - parts of the lexical analyzer may not be portable, but the parser always is portable.

4. Describe three advantages of LR parsers.
Ans:
- They will work for nearly all grammars that describe programming languages.
- They can detect syntax errors as soon as it is possible.
- They work on a larger class of grammars than other bottom-up algorithms, but are as efficient as any other bottom-up parser.

5. Perform the pairwise disjointness test for the following grammar rules.

a. $A \rightarrow aB \mid b \mid cBB$

b. $B \rightarrow aB \mid bA \mid aBb$

c. $C \rightarrow aaA \mid b \mid caB$

Ans:

(a) FIRST (aB) = {a}, FIRST (b) = {b}, FIRST (cBB) = {c}

These are disjoint and so, pass the pairwise disjoint test.

(b) FIRST (aB) = {a}, FIRST (bA) = {b}, FIRST (aBb) = {a}

These aren't disjoint and so, fail the pairwise disjoint test.

(c) FIRST (aaA) = {a}, FIRST (b) = {b}, FIRST (caB) = {c}

These are disjoint and so, pass the pairwise disjoint test.

6. Perform the pairwise disjointness test for the following grammar rules.

a. $S \rightarrow aSb \mid bAA$

b. $A \rightarrow b\{aB\} \mid a$

c. $B \rightarrow aB \mid a$

Ans:

(a) FIRST(aSb) = {a}, FIRST (bAA) = {b}

These are disjoint and so, pass the pairwise disjoint test

(b) FIRST (b{aB}) = {b}, FIRST (a) = {a}

These are disjoint and so, pass the pairwise disjoint test.

(c) FIRST (aB) = {a}, FIRST (a) = {a}

These aren't disjoint and so, fail the pairwise disjoint test.

7. Given the following grammar and the right sentential form, draw a parse tree and show the phrases and simple phrases, as well as the handle.

S → aAb | bBA A → ab | aAB B → aB | b

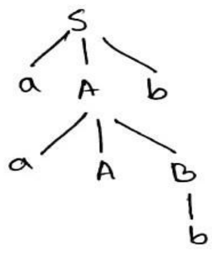
a. aaAbb

b. bBab

c. aaAbBb

Ans: (a) aaAbb

Parse Trees



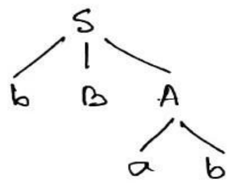
Phrases: aaAbb, aaABb, aAb

Simple Phrase: b

Handles : b, [redacted]

(b) bBab

Parse Tree:



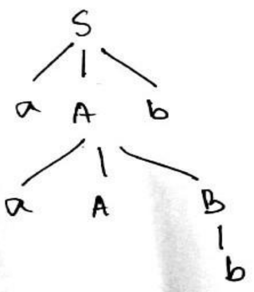
Phrases: bBab, bBA

Simple Phrase: ab

Handles ; ab

(c) aaAbBb

Parse Tree:



so, the last string can't be derived from the given grammar.

8. Given the following grammar and the right sentential form, draw a parse tree and show the phrases and simple phrases, as well as the handle.

S → AbB | bAc   A → Ab | aBB   B → Ac | cBb | c

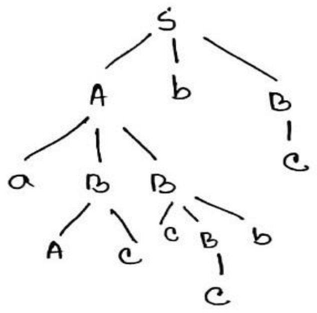
a. aAcccbbc

b. AbcaBccb

c. baBcBbbc

Ans: (a) aAcccbbc

Parse Tree:



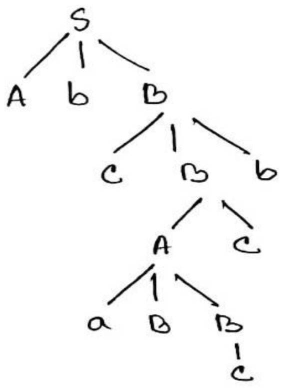
Phrases: aAcccbbc, aAcccbbB, aAccBbbB, aAcBbB, aBBbB, AbB

Simple phrase: c, Ac

Handles: Ac

(b) AbcaBccb

Parse Tree:



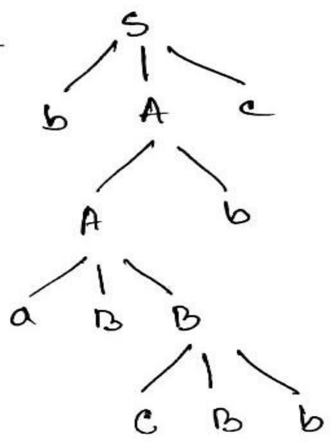
Phrases: AbcaBccb, AbcaBBcb, AbcAcb, AbcBb, AbB

Simple phrase: c

Handles: c

[illegible] baBcBbbc

Parse Tree:



Phrases: baBcBbbc, baBBbc, bAbc, bAc

Simple phrase: cBb

Handle: cBb

9. Describe the Advantages of Using BNF to Describe Syntax
Ans:
Advantages of Using BNF to Describe Syntax:

- Provides a clear and concise syntax description
- The parser can be based directly on the BNF
- Parsers based on BNF are easy to maintain

10. Describe briefly the three approaches to building a lexical analyzer.
Ans:
Three approaches to building a lexical analyzer:

- Firstly, write a formal description of the tokens. Then use a software tool that constructs a table-driven lexical analyzer from the description.
- Secondly, design a state diagram that describes the tokens. Then write a program that implements the state diagram.
- Finally design a state diagram that describes the tokens. Then hand-construct a table-driven implementation of the state diagram.