MCA 18 302
PRINCIPLES OF COMPILERS

MODULE 1

1. **Introduction to compiling**
    1. Definition of compiler, translator, interpreter
    2. Analysis of the source program
    3. The phases of a compiler
    4. Compiler construction tools
2. **Programming language basics**
3. **Lexical analysis**
    1. Role of lexical analyzer
    2. Input buffering
    3. Specification of tokens
    4. Recognition of tokens using finite automata
    5. Regular expressions and finite automata
    6. From NFA to DFA
    7. Regular expression to an NFA

Introduction to compiling

2. Analysis of the source program

- In compiling ,analysis consists of three phases:

1. Linear analysis or Lexical analysis or scanning

2. Hierarchical analysis or Syntax analysis or Parsing

3. Semantic analysis

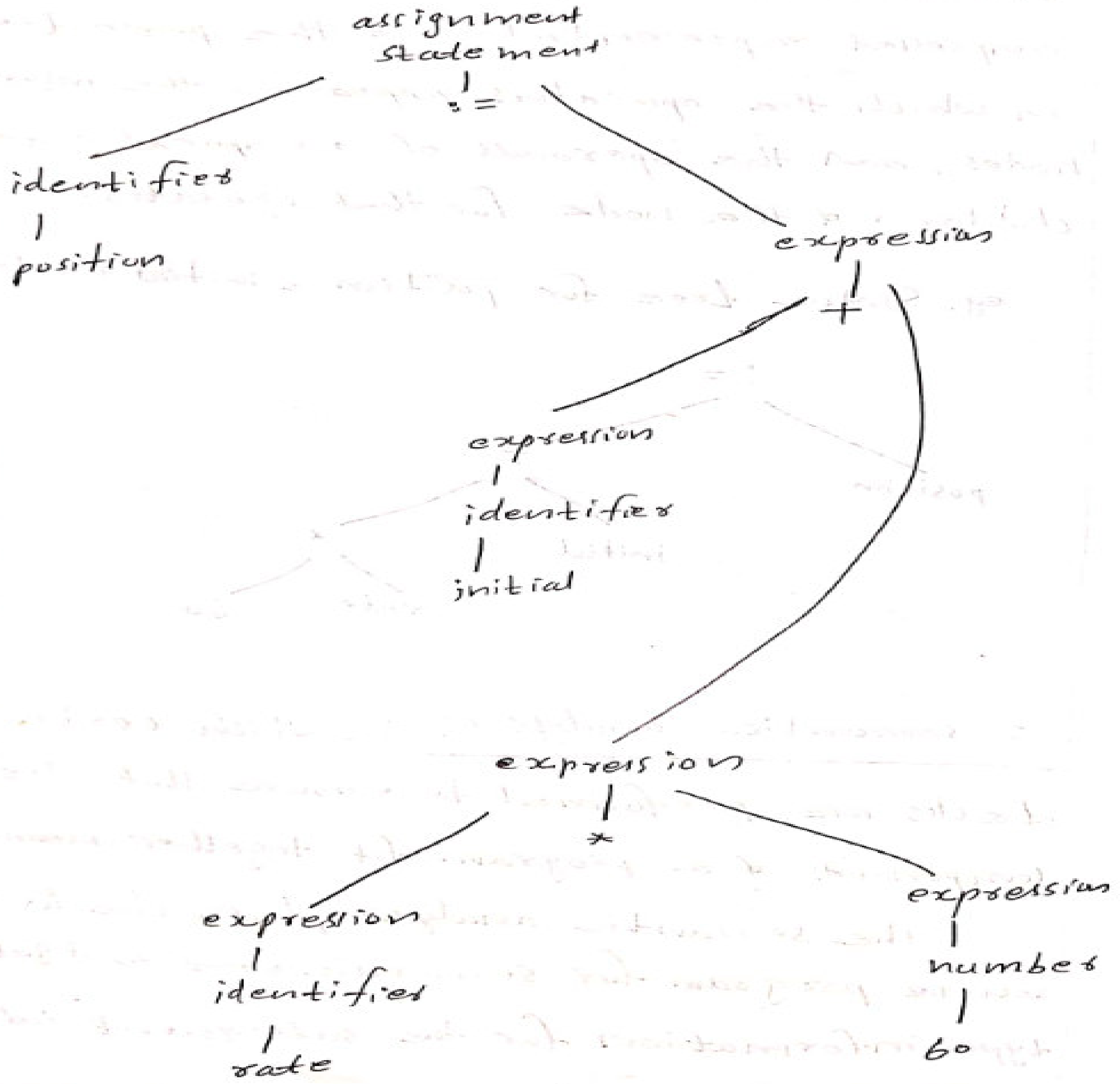
1. **Linear analysis:**

- In which the stream of characters making up the source program is read from left to right and grouped into tokens that are sequences of characters having a collective meaning.
- Eg: **position:=initial + rate * 60** would be grouped into the following tokens:
    1. The identifier position
    2. The assignment symbol :=
    3. The identifier initial
    4. The plus sign
    5. The identifier rate
    6. The multiplication sign
    7. The number 60
- The blank separating the characters of these tokens would normally be eliminated during lexical analysis.

2. Hierarchical analysis:

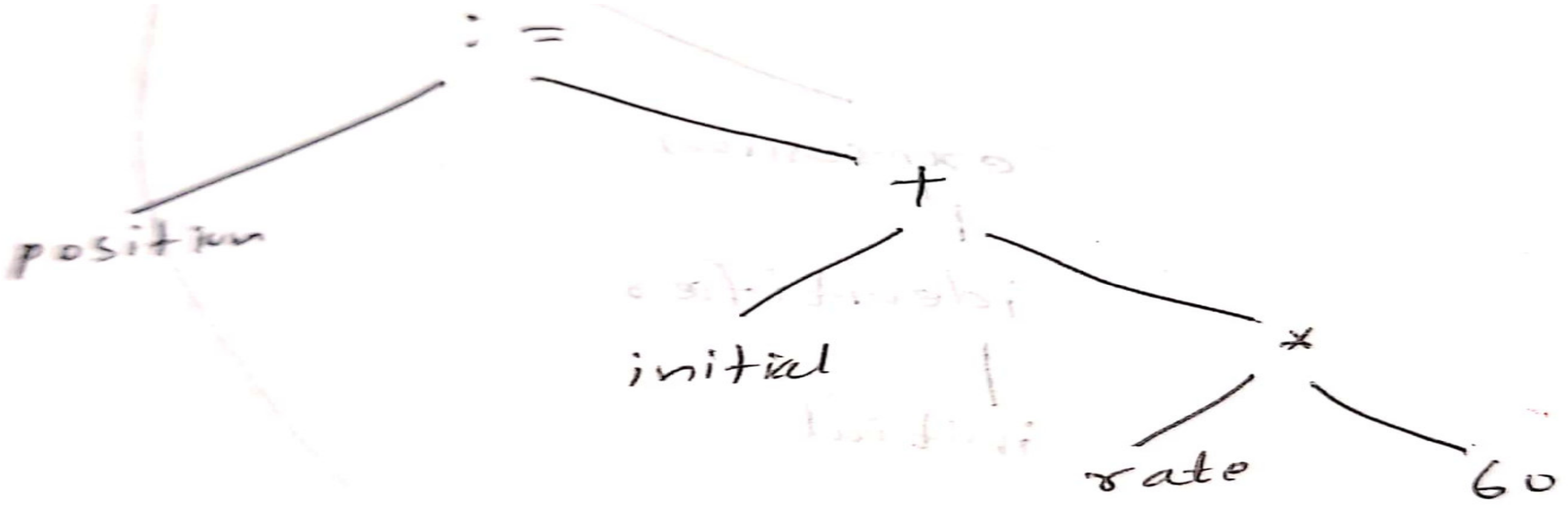
- In which characters or tokens are grouped hierarchically into nested collections with collective meaning.
- It involves grouping the tokens of the source program into grammatical phrases that are used by the compiler to synthesize output.Usually,the grammatical phrases of the source program are represented by a parse tree.
- The parse tree describes the syntactic structure of the input.

eg: parse tree for position := initial + rate * 60



- A syntax tree is a compressed representation of the parse tree in which the operators appear as the interior nodes, and the operands of an operator are the children of the node for that operator.

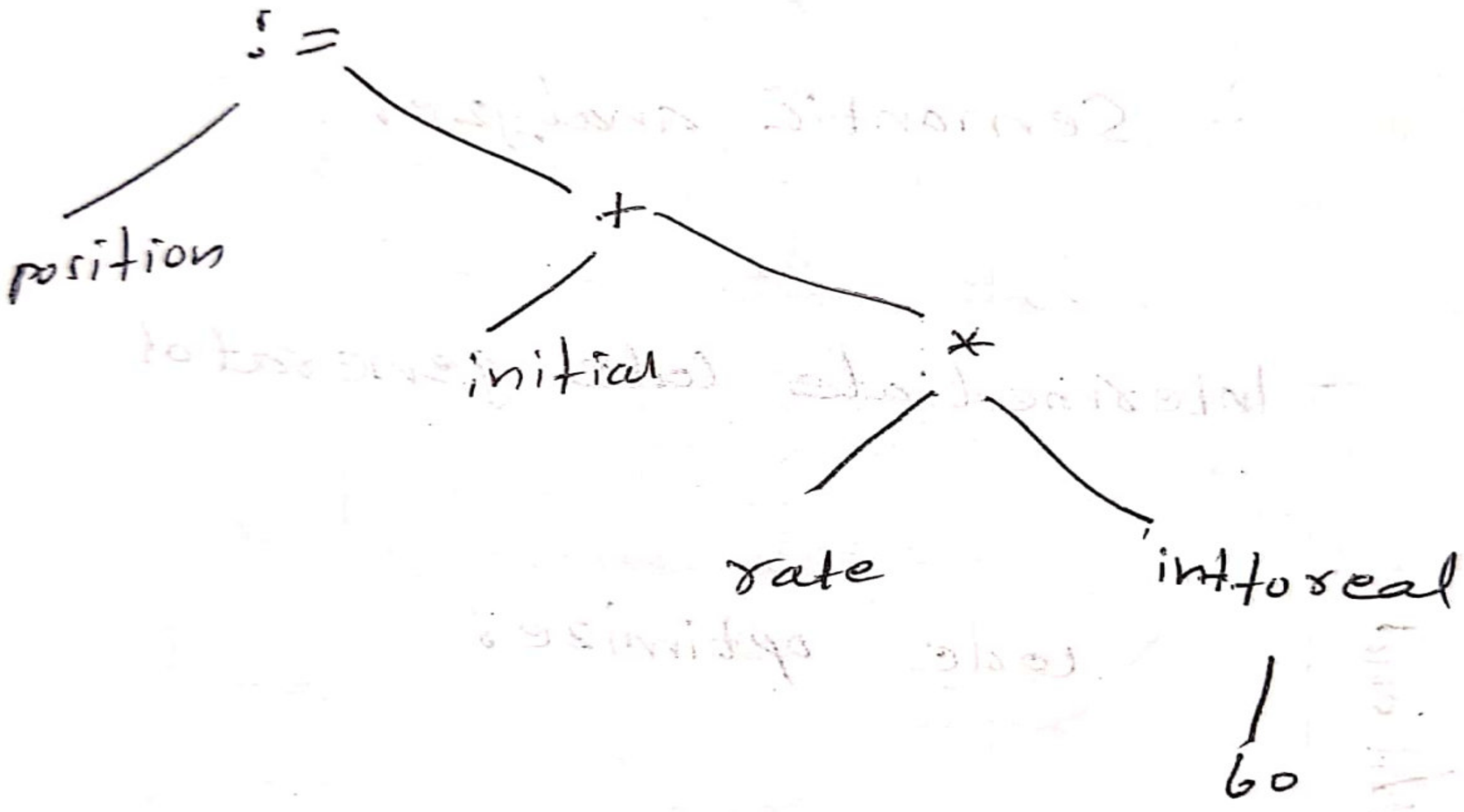
  Example: Syntax tree for position:=initial+rate*60



3. Semantic analysis:

- In which certain checks are performed to ensure that the components of a program fit together meaningfully.
- The semantic analysis phase checks the source program for semantic errors and gathers type information for the subsequent code generation phase. It uses the hierarchical structure determined by the syntax analysis phase to identify the operators and operands of expressions and statements.
- An important component of semantic analysis is type checking.

Example: Semantic analysis inserts a conversion from integer to real


:=
position
+
initial
*
rate
inttoreal
60