1. BEGIN: Think of this as the starting point. It tells the lexical analyzer (lexer) to start at a specific state, usually state 0.
2. ECHO: This action simply repeats or “echoes” the input as it is. It’s like a mirror reflecting the input back.
3. yytext():
    - yytext holds the characters that form a token (like a word or number) when the lexer recognizes it.
    - Whenever a new token is found, yytext gets updated with that token’s content.
4. yylex(): This function is crucial. It’s called when the lexer begins scanning the source program. Essentially, it’s the main action that drives the lexing process.
5. yywrap():
    - yywrap() is triggered when the lexer reaches the end of the file.
    - If yywrap() returns 0, the lexer keeps scanning.
    - If yywrap() returns 1, it indicates that the end of the file has been reached.
6. yyin: This is where the input source program is stored. It’s like the source document that the lexer reads from.
7. yyleng: This simply keeps track of how many characters are in the current input string being processed.

Related posts:

1. What are the types of passes in compiler ?
2. Discuss the role of compiler writing tools. Describe various compiler writing tools.
3. What do you mean by regular expression ? Write the formal recursive definition of a regular expression.
4. How does finite automata useful for lexical analysis ?
5. Explain the implementation of lexical analyzer.

6. Write short notes on lexical analyzer generator.
7. Explain the automatic generation of lexical analyzer.
8. Explain the term token, lexeme and pattern.
9. Describe grammar.
10. Explain formal grammar and its application to syntax analyzer.
11. Define parse tree. What are the conditions for constructing a parse tree from a CFG ?
12. Describe the capabilities of CFG.
13. What is parser ? Write the role of parser. What are the most popular parsing techniques ? OR Explain about basic parsing techniques. What is top-down parsing ? Explain in detail.
14. What are the common conflicts that can be encountered in shift-reduce parser ?
15. Differentiate between top-down and bottom-up parser.Under which conditions predictive parsing can be constructed for a grammar ?
16. Differentiate between recursive descent parsing and predictive parsing.
17. What is the difference between S-attributed and L-attributed definitions ?
18. What is intermediate code generation and discuss benefits of intermediate code ?
19. Define parse tree. Why parse tree construction is only possible for CFG ?
20. Discuss symbol table with its capabilities ?
21. What are the symbol table requirements ? What are the demerits in the uniform structure of symbol table ?