Token:

- Think of a token as a categorized unit in a program's code.
- It consists of two parts:
    - The token name, which is like a label representing a type of unit (like "number" or "operator").
    - An optional attribute value, which gives extra information about the token (like the specific value of a number).
- In a program, tokens can be things like words (keywords), numbers, symbols (like "+" or "("), and so on.

Lexeme:

- A lexeme is the actual sequence of characters in the source code that matches the pattern for a token.
- It's like the specific instance of a token found in the code.
- For example, if "if" is a keyword token, then in the code, the word "if" itself would be the lexeme that matches this token.

Pattern:

- A pattern is a description of the form that the lexemes of a token can take.
- It's like a blueprint or rule that defines what a valid token looks like.
- Regular expressions are commonly used to define patterns. These are special sequences of characters that help match patterns in text.
- For instance, if we have a keyword token pattern, the pattern might simply be the sequence of characters that make up that keyword, like "if" or "while".

Explain the term token, lexeme and pattern.

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. What are the various LEX actions that are used in LEX programming ?
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

Explain the term token, lexeme and pattern.

structure of symbol table ?