Hints on the lexical analyzer: A tokenizer

1. Read each line of input into a buffer. Assume 80 characters per line. This is to setup a string for each line. March through the string for lexical analysis.

2. Print each string when it is read to a listing file (text file). The syntax analyzer is going to call the lexical analyzer for the next token. Error handling is part of this scheme. The errors can be written for the line printed out with the string. (has all error messages).

3. Symbol table - Preload symbol table with all reserved words and symbols. In our example, the first 36 entries will be reserved words and symbols. Starting at the 37th element, all of the identifiers and constants and non-reserved words. The easiest data structure should be chosen for the symbol table. Reserve the hard work for the compiler itself. There should be a few items in this record. Also fix the identifier type to token type 37 and constants to token type 38. The row index and token type should be returned with the record.

    - Name String
    - location
    - Token Type
    - Index of token
    - constants
    - integer

4. Lexical analyzer returns token type row in symbol table.

5. Constants and Id 's you first search (these constants must be fixed types). The purpose is to see if the string already exists in the symbol table.

6. A match of a symbol, simply returns its location in the symbol table and record in the symbol table.

7. Operators, and special symbols are delimiters. How the delimiter is identified is immaterial. What matters is that when one of these delimiters is encountered, the token acquisition stops. Special cases exist for multiple character special symbols, such as :=. Stopping on space implies that a delimiter on beginning the reading of next token - throw out the spaces. End of line is a delimiter.

8. Token type and symbol table location should be made integer types, and global variables, contrary to SE's idea. The symbol table may be useful as a global variable.

9. One can include the length of identifiers. Another can be identifier types should be alphabetical characters only.