The objective of this assignment is to gain an understanding of the lexical analysis phase of a compiler and the process of constructing a symbol table.

**Problem:** The first phase of compilation is called scanning or lexical analysis. This phase interprets the input program as a sequence of characters and produces a sequence of tokens, which will be used by the parser.

Write a program (in C, C++, C#, Java, or Python) that implements a simple scanner for a source file given as a command-line argument. The format of the tokens is described below. You may assume that the input is syntactically correct. Your program should build a symbol table which contains an entry for each token that was found in the input. When all the input has been read, your program should produce a summary report that includes a list of all the tokens that appeared in the input, the number of times each token appears in the input and the class of each token. Your program should also list how many times tokens of each class appeared in the input.

The grammar for producing tokens is as follows:

keyword ::= if | then | else | begin | end

identifier -> character | character identifier

integer -> digit | digit integer

real -> integer.integer

special -> ( | ) | [ | ] | + | - | = | , | ;

digit -> 0|1|2|3|4|5|6|7|8|9

character -> a|b|c ... |z|A|B|C ... |Z

More details:



Case is not used to distinguish keywords or identifiers.



The delimiters are space, tab, newline, and the (.



The token classes that should be recognized are keyword, identifier, integer, real and special.