Assignment #2

Due Monday April 27th

A). Create a syntatic analyzer (parser) for the *MINI-P* language described below. The parser will utilize the scanner written in Assignment #1.

The parser will have two types of output, diagnostic message or a confirmation message for each line (all to the console, no print). A line should trigger an error (diagnostic) message if it is not recognized as valid syntax. Otherwise, a simple message should be printed to indicate the line is valid syntax.

Your parser must catch syntax errors when a line does not conform to the MINI-P grammar. Be sure your diagnostic message provides enough information to enable the user to locate the offending source code.

Grammar for MINI-P

⟨prog⟩ ::= PROGRAM ⟨prog-name⟩ VAR ⟨dec-list⟩ BEGIN ⟨stmt-list⟩ END.
⟨prog-name⟩ ::= **id**
⟨dec-list⟩ ::= ⟨dec⟩ { ; ⟨dec⟩ }
⟨dec⟩ ::= ⟨id-list⟩ : ⟨type⟩
⟨type⟩ ::= INTEGER
⟨id-list⟩ ::= **id** { , **id** }
⟨stmt-list⟩ ::= ⟨stmt⟩ { ; ⟨stmt⟩ }
⟨stmt⟩ ::= ⟨assign⟩ | ⟨read⟩ | ⟨write⟩ | ⟨for⟩
⟨assign⟩ ::= **id** := ⟨exp⟩
⟨exp⟩ ::= ⟨term⟩ { + ⟨term⟩ | - ⟨term⟩ }
⟨term⟩ ::= ⟨factor⟩ { * ⟨factor⟩ | DIV ⟨factor⟩ }
⟨factor⟩ ::= **id** | **int** | (⟨exp⟩)
⟨read⟩ ::= READ (⟨id-list⟩)
⟨write⟩ ::= WRITE (⟨id-list⟩)
⟨for⟩ ::= FOR ⟨index-exp⟩ DO ⟨body⟩
⟨index-exp⟩ ::= **id** := ⟨exp⟩ TO ⟨exp⟩
⟨body⟩ ::= ⟨stmt⟩ | BEGIN ⟨stmt-list⟩ END