Devise a finite state automaton which will recognize the next valid Pascal Junior token from the input each time it is called. Label transitions with char1,char2,...,charn/action. Use lambda for "other" characters. The scanner should recognize all Pascal Junior tokens, it should handle reading and printing of source lines, and it should recognize compiler options of the form (*$option.....*).

Implement the automaton as a transition matrix with a procedure named GETBSU. Add another procedure, GETTOKEN, which calls GETBSU and then looks up and classifies language keywords (e.g., "and", "program", "if", etc.). Except for alphabetic tokens and string literals, the classification can be done directly by GETBSU, so GETTOKEN does not need to consider them.

It is suggested that GETTOKEN have 3 parameters: an array of characters to hold the token itself, a variable to hold the number of characters in the token, and a variable to hold the classification of the token. GETTOKEN should be called from a driver routine as a test:

Test data will be placed on the web site for CS5304. All files are named la*.tst