**Table Driven Scanner**

1. Write the regular expression and draw a DFA for an identifier. Build the corresponding classifier table (character category) and transition table for the same in your observation notebook and then implement it using Table driven scanner. Trace for the sample inputs in your observation notebook and show the program output in the following format. Show the output token as valid or invalid while reading stream of characters continuously till EOF.

**RDP**

Write a set of recursive descent parsing functions that recognize the fictitious grammar given below:

program -> stmt

stmt -> ( for | asst | ϵ ) ';'

for -> FOR ‘(’ expr ‘;’ expr ? ‘;’ expr ‘)’ stmt

asst -> ID ( ‘=’ expr )

Expr -> ID | CONST | ‘(’ expr ‘)’

The functions need only parse and recognize an input, and exit upon encountering the first error.

**Semantic Actions / Parser / Lexer**

Construct a grammar for the language L = {w : n a (w) < n b (w) < n c (w) } and write an attribute grammar to check the input validation.