return (TK_COMMENT, %) ⟵ 35 ⟵ %

return (TK_SQL, [) ⟵ 36 ⟵ [

return (TK_SQR,]) ⟵ 37 ⟵]

return (TK_COMMA, ,) ⟵ 38 ⟵ ,

return (TK_SEM, ;) ⟵ 39 ⟵ ;

return (TK_COLON, :) ⟵ 40 ⟵ :

return (TK_DOT, .) ⟵ 41 ⟵ .

return (TK_AND, &&&) ⟵ 44 ⟵ & ⟵ 43 ⟵ & ⟵ 42 ⟵ &

0 ⟶ start state

return (TK_OR, @@@) ⟵ 47 ⟵ @ ⟵ 46 ⟵ @ ⟵ 45 ⟵ @

return (TK_LT, <) 49 * other
return (TK_ASSIGNOP, <---) ⟵ 54 ⟵ 53 (Minus) ⟵ 52 (Minus) ⟵ 50 (Minus) ⟵ 48 ⟵ <
52 *** other
51 ** other

return (TK_LE, <=) ⟵ 55 ⟵ =

return (TK_GT, >) ⟵ 57 * other ⟵ 56 ⟵ >

return (TK_GE, >=) ⟵ 58 ⟵ =

return (TK_EQ, ==) ⟵ 60 ⟵ 59 ⟵ =

return (TK_OP, () ⟵ 61 ⟵ (

return (TK_CL,)) ⟵ 62 ⟵)

return (TK_PLUS, +) ⟵ 63 ⟵ +

return (TK_MINUS, -) ⟵ 64 ⟵ -

Space ⟶ 1 (Space, other) ⟶ 2 *

[0-9] ⟶ 3 ⟶ [0-9] ⟶ 4 * return (TK_NUM, Lexeme)
3 ⟶ . (decimal) ⟶ 5 ⟶ [0-9] ⟶ 7 ⟶ [0-9] ⟶ 9 ⟶ other ⟶ 10 * return (TK_RNUM, Lexeme)
5 ⟶ other ⟶ 6 **
7 ⟶ other ⟶ 8 ***
9 ⟶ E ⟶ 11 ⟶ other ⟶ 12 ** +, - (Plus, Minus)
11 ⟶ [0-9]
return (TK_RNUM, Lexeme) ⟵ 16 ⟵ [0-9] ⟵ 15 ⟵ [0-9] ⟵ 13 ⟵ other ⟵ 14 ***
12 ⟶ 13

_ (Underscore) ⟶ 17 ⟶ [a-z | A-Z] ⟶ 18 ⟶ [a-z | A-Z] [0-9] ⟶ 19 ⟶ [0-9]
18 ⟶ other ⟶ 20
19 ⟶ other ⟶ 20 * return (Search_token(), Lexeme) for _main
return (TK_FUNID, Lexeme)

⟶ 21 ⟶ [a-z] ⟶ 22 [a-z] other ⟶ 23 * return (TK_RUID, Lexeme)

[b-d] ⟶ 24 ⟶ [2-7] ⟶ 25 [b-d] [2-7] ⟶ 26 [2-7] other ⟶ 27 * return (TK_ID, Lexeme)
24 ⟶ other ⟶ 29
25 ⟶ other ⟶ 29

a, [2-2] ⟶ 28 [a-z] other ⟶ 29 * return (Search_token(), Lexeme) for keywords other than _main
return (TK_FIELDID, Lexeme)

* ⟶ 30 return (TK_MUL, *)
/ ⟶ 31 return (TR_DIV, /)
~ ⟶ 32 return (TK_NOT, ~)
! ⟶ 33 ⟶ = ⟶ 34 return (TK_NE, !=)

Note: "other" refers to those characters for which transition from the state isn't shown

Group - 10

1) Akash S Revankar (2019ATPS 0294P)
2) Harsh Butani (2019ATPS 0022P)
3) Hemant Singh Sisodiya (2019 ATPS 0070 P)
4) Mohit Sharma (2019ATPS0100P)
5) Siddharth Upadhyay (2019 ATPS0033P)

Notes :

1) For state 2, we backtrack by 1 character and do not return any token

2) For state 6, we backtrack by 2 characters and return (TK_NUM, Lexeme)

3) For state 8, we backtrack by 3 characters and return (TK_NUM, Lexeme)

4) For state 12, we backtrack by 2 characters and return (TK_RNUM, Lexeme)

5) For state 14, we backtrack by 3 characters and return (TK_RNUM, Lexeme)

6) For state 20, we first lookup for keyword "_main". If found, we return (TK_MAIN, _main) it else we return (TK_FUNID, Lexeme)

7) For state 29, we first lookup for keywords other than "_main". If found, we return the corresponding token for that keyword else we return (TK_ID, Lexeme)

8) For state 51, we retract by 2 characters and return (TK_LT, <)

9) For state 53, we retract by 3 characters and return (TK_LT, <)

10) If we reach a dead state (not shown in DFA), we report a lexical error.