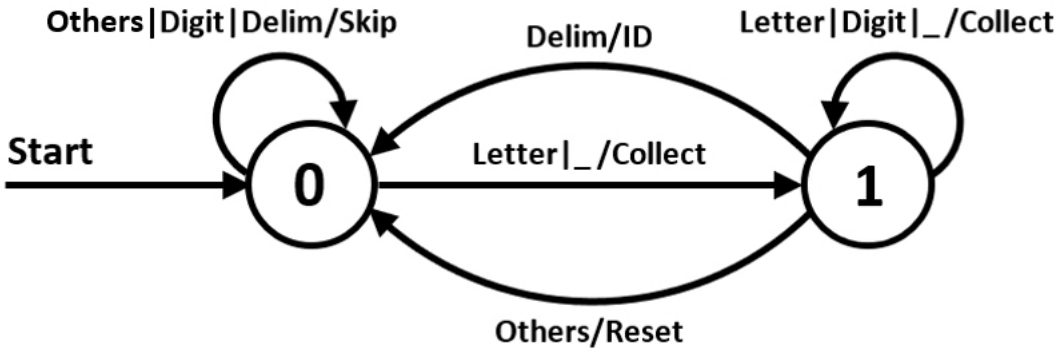
Exp. 4: Symbol table implementation / identifier recognition – Lexical analysis

Step 1: Build a finite automata for identifier recognition



Step 2: Construct the transition table

State	Input Symbols			
	Letter|_	Digit	Delim	Others
0	Collect 1	Skip 0	Skip 0	Skip 0
1	Collect 1	Collect 1	ID 0	Reset 0

Notes:

- **Collect action adds the current input symbol to the token string.**
- **Skip action performs nothing (no action).**
- **ID action inserts a new identifier into the symbol table.**
- **Reset action resets token string (token=null).**
- **Delim refers to a delimiter (separator character).**

```
1  //########## Exp.4: Symbol Table Implementation ##########
2  //#########     Coded By: AHMED ALI ALSAMMAN     #########
3  //#########       Input: Source code file       #########
4  //################ Output: Symbol Table ###################
5  using System;
6  using System.Collections.Generic;
7  using System.Linq;
8  using System.IO;
9
10
11 namespace symbolTable
12 {
13
14
15     public struct Identifier
16     {
17         public string name;
18
19
20         public Identifier(string name)
21         {
22             this.name = name;
23         }
24     }
25
26
27     public class SymbolTable
28     {
29         public List<Identifier> ID = new List<Identifier>();
30
31
32         public int Lookup(string idName)
33         {
34             for (int i = 0; i < ID.Count; i++)
35                 if (ID[i].name == idName)
36                     return i;
37             return -1;
38         }
39
40
41
42         public void Insert(string name)
43         {
44             if (Lookup(name) == -1) // variable not found
45                 ID.Add(new Identifier(name));
46         }
47     }
48
49
50
51
52
```

```
53
54      class Program
55      {
56          static string token = null;
57          static string[] KW = new string[]
              {"int","char","float","string","bool","double","void","main","break",
               "if","else","switch","case","while","for","exit"};
58          static char c;
59          static SymbolTable symbTab= new SymbolTable();
60          public delegate void delgFunc();
61
62
63          static void Collect()
64          {
65              token += c;
66          }
67
68
69          static void ID()
70          {
71              if (!KW.Contains(token)) // check if it's not a keyword
72                  symbTab.Insert(token); // insert a new identifier
73              token = null;
74          }
75
76
77          static void Reset()
78          {
79              token = null;
80          }
81
82          static void Skip() { }
83
84
85          static void Main(string[] args)
86          {
87
88              int[,] transTab = new int[,] {
89                  // letter|_   Digit   Delimiter  Others
90                   {   1,         0,        0,        0   },//state 0
91                   {   1,         1,        0,        0   } //state 1
92
93              };
94
95
96
97              delgFunc[,] action = new delgFunc[,] {
98
99                   {Collect,      Skip,    Skip,    Skip},
100                  {Collect,   Collect,      ID,   Reset}
101             };
102
```

```
103
104             StreamReader sr = new StreamReader(@"d:\source.txt");
105             int state = 0;
106
107             c = (char)sr.Read();
108
109             while (!sr.EndOfStream)
110             {
111                 int colm = getColIndx(c);
112                 action[state, colm]();
113                 state = transTab[state, colm]; // get a new state
114                 c = (char)sr.Read(); // read next character
115                 if (c == '\n')  c = (char)sr.Read(); // skip new line character
116             }
117             sr.Close();
118
119             foreach (Identifier id in symbTab.ID)
120                 Console.WriteLine("Name: {0}    Type:    Object Address:     Line ↵
                    Declared:    Line Referenced:", id.name);
121         }
122
123
124
125         static int getColIndx(char c)
126         {
127             char[] del = { ' ', '=', '+', '-', '*', '/', ',', ';', '\r', '<',     ↵
                '>','|','&' };
128             if (Char.IsLetter(c) || c == '_') return 0;
129             if (Char.IsDigit(c)) return 1;
130             if (del.Contains(c)) return 2;
131             return 3;
132         }
133     }
134 }
```