

```

1  #include<stdio.h>
2  #include<ctype.h>
3  #include<string.h>
4  char
keyword[24][30]={"int","while","break","for","do","if","float","char","switch","double","shor
rt","long","unsigned","sizeof","else","register","extern","static","auto","case","break","vo
latile","enum","typedef"};

5
6  int check_keyword(char s[]) //linear search
7  {
8      int i;
9      for(i=0;i<24;++i)
10         if(strcmp(s,keyword[i])==0)
11             return 1;
12     return 0;
13 }
14
15 void store_symb_tab(char id[], char symb_tab[][30])
16 {
17     int i;
18     for(i=0; strcmp(symb_tab[i],"")&&i<20;++i)
19         if(!strcmp(symb_tab[i],id))
20             return;
21     if(i==20)
22     { printf("Overflow!"); return;} // create linked list to avoid this
23     strcpy(symb_tab[i],id); //adds id to symb_tab
24 }
25
26 int main()
27 {
28     FILE *fp1,*fp2;
29     char c,id[30], num[10];
30     int state=0,i=0,j=0;
31     fp1=fopen("input.txt","r");//input file containing src prog
32     fp2=fopen("output.txt","w");//output file name
33     //declare symbol table as a doubly dimensional array of characters.
34     char symb_tab[20][30];
35
36     while((c=fgetc(fp1))!=EOF)
37     {
38         switch(state)
39         {
40             case 0:
41
42                 if(isalpha(c)||c=='_')
43                 { state=1; id[i++]=c; }
44                 else if(isdigit(c))
45                 { state=3; num[j++]=c; }
46                 else if(c=='<' || c=='>')
47                     state=5;
48                 else if(c=='=' || c=='!')
49                     state=8;
50                 else if(c=='/')
51                     state=10;
52                 else if(c==' ' || c=='\t' || c=='\n')
53
54                     state=0;
55                 else if(c=='\r'); //checks for newline in file
56                 else
57                     fprintf(fp2,"\n%c",c);
58                 break;
59             case 1:
60                 if(isalnum(c)||c=='_')
61                 { state=1; id[i++]=c; }
62                 else{
63                     id[i]='\0';
64                     if(check_keyword(id))
65                         fprintf(fp2," \n%s : keyword ",id);
66                     else{
67
68                         fprintf(fp2,"\n%s : identifier",id);
69                         // call a function which stores id in symbol table
70                         store_symb_tab(id,symb_tab);
71                     }
72                     state=0;
73                     i=0;
74                     ungetc(c,fp1);
75                 }
76                 break;
77             case 3:
78                 if(isdigit(c))
79                 { num[j++]=c; state=3; }
80                 else{
81                     num[j]='\0';

```

```

82         fprintf(fp2, " \n%s: number", num);
83         state=0;
84         j=0;
85         ungetc(c, fp1);
86     }
87     break;
88 case 5:
89     if(c=='='){
90         //write code to print specific operator like <= or >=
91         fseek(fp1, -2, SEEK_CUR); //go back 2 chars
92         c=fgetc(fp1); // read '<' or '>' again
93         if(c=='<')
94             fprintf(fp2, "\n<=: relational operator LE");
95         else
96             fprintf(fp2, "\n<=: relational operator GE");
97         c=fgetc(fp1); // read '=' again
98         state=0;
99     }
100     else{
101         //write code to print specific operator like <, >, <= or >=
102         fseek(fp1, -2, SEEK_CUR); //go back 2 chars
103         c=fgetc(fp1); // read '<' or '>' again
104         if(c=='<')
105             fprintf(fp2, "\n<: relational operator LT");
106         else
107             fprintf(fp2, "\n>: relational operator GT");
108         //c=fgetc(fp1) // read '=' again
109         state=0;
110         //ungetc(c, fp1);
111     }
112     break;
113 case 8:
114     if(c=='='){
115         //write code to print specific operator like == or !=
116         fseek(fp1, -2, SEEK_CUR); //go back 2 chars
117         c=fgetc(fp1); // read '!' or '=' again
118         if(c=='=')
119             fprintf(fp2, "\n==: relational operator EQ");
120         else
121             fprintf(fp2, "\n!=: relational operator NE");
122         c=fgetc(fp1); // read '=' again
123         state=0;
124     }
125     else{
126         fprintf(fp2, "\n!");
127         ungetc(c, fp1);
128         state=0;
129     }
130     break;
131 case 10:
132     if(c=='*')
133         state=11;
134     else{
135         fprintf(fp2, "\n/%c: invalid lexeme", c);
136         state=0;
137     }
138     break;
139 case 11:
140     if(c=='*')
141         state=12;
142     //else
143     //    state=11;
144     break;
145 case 12:
146     if(c=='*')
147         state=12;
148     else if(c=='/')
149         state=0;
150     else
151         state=11;
152     break;
153
154     } //End of switch
155 } //end of while
156 if(state==11)
157     fprintf(fp2, "comment did not close");
158
159 // To print symbol table remove these comments
160 for(int i=0; strcmp(symb_tab[i], "") && i<20; ++i)
161     fprintf(fp2, "\n identifier %d - %s", i+1, symb_tab[i]);
162 return 0;
163 fclose(fp1);
164 fclose(fp2);
165 }

```