

---

# A2-Implementation of Lexical Analyzer for the patterns using Lex

---

Sneha Sriram Kannan 185001157

19-02-2021

## 1 Code

```
1 /* To implement lexical analyzer using lex */
2 %{
3     #include<stdio.h>
4     #include<string.h>
5
6     void addToSymbolTable(char id[]); //Function called whenever identifier is encountered
7     void PrintSymbolTable(); //Displays the contents of the symbol table
8     void addValueToTable(char value[]); //Value is found for the identifier so the value is
9     added to the symbol table
10    int notInSymbolTable(char id[]); //Checks if the identifier is already present in the
11    symbol table
12    void noValue(); //semicolon encountered so no value available for the identifier
13
14    typedef struct SymbolTable{
15        char id_name[50][20];
16        int addr[50];
17        int size[50];
18        char type[50][10];
19        char value[50][30];
20        int row;
21        int curraddr;
22    }SymbolTable;
23
24    char currType[10];
25    int assignFlag=0; //set when = is encountered
26    int waitingForValue=0; //set when identifier is encountered but value has not yet been
27    defined
28
29    SymbolTable *s;
30
31 %}
32
33 singleline [/]{2}.*
34 multiline "/*"([~*]*)"/"
35 assign "="
36 keyword (if|else|do|while|void|int|float|char)
37 function [ ]*[a-zA-Z]+\((.*)\).*
38 int [-+]?[0-9]+
39 float [-+]?[0-9]*[.][0-9]+
40 charconst \'.\'
41 strconst \".*\n"
42 id [a-zA-z][a-zA-z0-9]*
43 arithop [+*-%/]
44 relop [<=>]{1,2}
45 logicalop &{2}||{1,2}
```

```

42 seperator [();,{}]
43
44 %%
45 {multiline} {printf("MULTILINE ");}
46 {singleline} {printf("SINGLELINE ");}
47 {keyword} {printf("KW ");strcpy(currType,yytext);}
48 {function} {printf("FC ");}
49 {int} {printf("NUMCONST ");addValueToTable(yytext);}
50 {float} {printf("NUMCONST ");addValueToTable(yytext);}
51 {id} {printf("ID ");addToSymbolTable(yytext);}
52 {assign} {printf("ASSIGN ");assignFlag=1;}
53 {charconst} {printf("CHARCONST");addValueToTable(yytext);}
54 {strconst} {printf("STRCONST");addValueToTable(yytext);}
55 {arithop} {printf("ARITHOP ");}
56 {relop} {printf("RELOP ");}
57 {logicalop} {printf("LOGICALOP ");}
58 "," {printf("SP ");assignFlag=0;noValue();}
59 {seperator} {printf("SP ");}
60 "\n" {printf("\n");}
61 %%
62
63 int yywrap(void){}
64
65 int main()
66 {
67     s=malloc(sizeof(SymbolTable));
68     s->row=0;
69     s->curraddr=3000;
70     yyin = fopen("code.txt", "r");
71     printf("\n=====
72     \nOUTPUT OF LEXICAL ANALYZER\n
73     =====\n");
74     yylex();
75     PrintSymbolTable();
76     return 0;
77 }
78
79 void PrintSymbolTable(){
80     printf("\n\n=====
81     \nContents of Symbol Table\n
82     =====\n");
83     int i;
84     printf("\n-----\nName\tType\tValue\tSize\
85     \tAddress\n-----\n");
86     for(i=0;i<s->row;i++){
87         printf("%s\t%s\t\t%s\t%d\t%d\n",s->id_name[i],s->type[i],s->value[i],s->size[i],s->
88         addr[i]);
89     }
90 }
91
92 void addToSymbolTable(char id[]){
93     int index=s->row;
94     if(notInSymbolTable(id)==1)
95     {
96         strcpy(s->id_name[index],id);
97         s->addr[index]=s->curraddr;
98         strcpy(s->type[index],currType);
99         if(strcmp(s->type[index],"int")==0)
100         {
101             s->curraddr+=2;
102             s->size[index]=2;
103         }
104         else if(strcmp(s->type[index],"char")==0)
105         {
106             s->curraddr+=1;
107             s->size[index]=1;
108         }
109         else if(strcmp(s->type[index],"float")==0)
110         {
111             s->curraddr+=4;
112             s->size[index]=4;
113         }
114         waitingForValue=1;
115     }
116 }

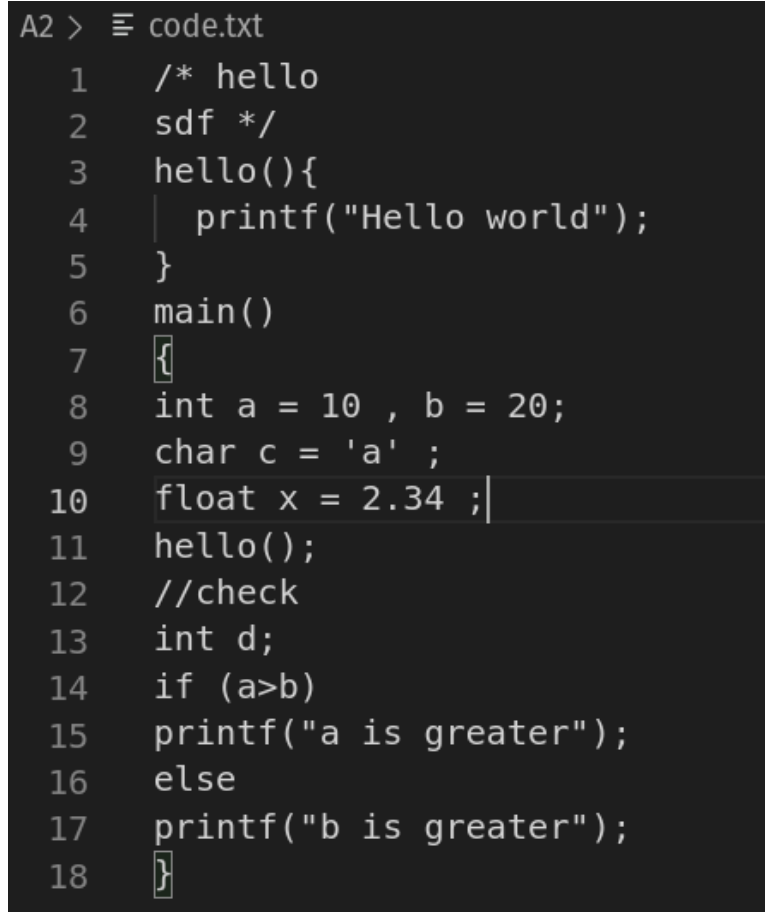
```

```

110 }
111
112 int notInSymbolTable(char id[]){
113     int i=0;
114     //checks if the identifier has already been declared before
115     for(i=0;i<s->row;i++){
116         if(strcmp(id,s->id_name[i])==0){
117             return 0;
118         }
119     }
120     return 1;
121 }
122
123 void addValueToTable(char value[]){
124     int index=s->row;
125     //value for the identifier has been found so it is added to the symbol table
126     if(assignFlag==1 && waitingForValue==1){
127         assignFlag=0;
128         waitingForValue=0;
129         strcpy(s->value[index],value);
130         s->row=s->row+1;
131     }
132 }
133
134 void noValue(){
135     int index=s->row;
136     //semicolon encountered but no value assigned to the identifier
137     if(waitingForValue==1)
138     {
139         strcpy(s->value[index],"NA");
140         s->row=s->row+1;
141     }
142     waitingForValue=0;
143 }

```

## 2 Output Screenshot



```
A2 > ≡ code.txt
1  /* hello
2  sdf */
3  hello(){
4  |   printf("Hello world");
5  }
6  main()
7  {
8  int a = 10 , b = 20;
9  char c = 'a' ;
10 float x = 2.34 ;|
11 hello();
12 //check
13 int d;
14 if (a>b)
15 printf("a is greater");
16 else
17 printf("b is greater");
18 }
```

Figure 1: Input to Lexical Analyzer

```
=====
OUTPUT OF LEXICAL ANALYZER
=====
MULTILINE
FC
FC
SP
FC
SP
KW  ID  ASSIGN  NUMCONST  SP  ID  ASSIGN  NUMCONST  SP
KW  ID  ASSIGN  CHARCONST  SP
KW  ID  ASSIGN  NUMCONST  SP
FC
SINGLELINE
KW  ID  SP
KW  SP  ID  RELOP  ID  SP
FC
KW
FC
SP

=====
Contents of Symbol Table
=====

-----
Name      Type          Value      Size      Address
-----
a          int             10         2         3000
b          int             20         2         3002
c          char            'a'        1         3004
x          float           2.34       4         3005
d          int             NA         2         3009
snehakannan@pop-os:~/Sneha/Semester 6/Compiler Design/Lab/A2$
```

Figure 2: Lexical Analyzer Output and Symbol Table