EXPERIMENT-1

1.1 OBJECTIVE:

Design a lexical analyzer for given language and the lexical analyzer should ignore redundant spaces, tabs and new lines. It should also ignore comments. Although the syntax specification states that identifiers can be arbitrarily long, you may restrict the length to some reasonable value. Simulate the same in C language.

1.2 RESOURCE:

Turbo C ++, Codeblock

1.3 PROGRAM LOGIC:

1. Read the input Expression
2. Check whether input is alphabet or digits then store it as identifier
3. If the input is is operator store it as symbol
4. Check the input for keywords

1.4 PROCEDURE:

Go to debug -> run or press CTRL + F9h to run the program

1.5 PROGRAM:

```
#include<string.h>

#include<ctype.h>

#include<stdio.h>

void keyword(char str[10])

{

if(strcmp("for",str)==0||strcmp("while",str)==0||strcmp("do",str)==0||
strcmp("int",str)==0||str
cmp("float",str)==0||strcmp("char",str)==0||strcmp("double",str)==0||strcmp("static",str)=
=0||strcmp("switch",str
)==0||strcmp("case",str)==0)

printf("\n%s is a keyword",str);

else

printf("\n%s is an identifier",str);

}
```

```
main()
{
FILE *f1,*f2,*f3;
char c,str[10],st1[10];
int num[100],lineno=0,tokenvalue=0,i=0,j=0,k=0;
printf("\nEnter the c program");/*gets(st1);*/
f1=fopen("input","w");
while((c=getchar())!=EOF)
putc(c,f1);
fclose(f1);
f1=fopen("input","r");
f2=fopen("identifier","w");
f3=fopen("specialchar","w");
while((c=getc(f1))!=EOF){
if(isdigit(c))
{
tokenvalue=c-'0';

c=getc(f1);
while(isdigit(c)){
tokenvalue*=10+c-'0';
c=getc(f1);
}
num[i++]=tokenvalue;
ungetc(c,f1);
}
```

```
else if(isalpha(c))
{
putc(c,f2);
c=getc(f1);
while(isdigit(c)||isalpha(c)||c=='_'||c=='$')
{
putc(c,f2);
c=getc(f1);
}
putc(' ',f2);
ungetc(c,f1);
}
else if(c==' '||c=='\t')
printf(" ");
else
if(c=='\n')
lineno++;
else
putc(c,f3);
}
fclose(f2);
fclose(f3);
fclose(f1);
printf("\nThe no's in the program are");
```

```
for(j=0;j<i;j++)

printf("%d",num[j]);

printf("\n");

f2=fopen("identifier","r");

k=0;
printf("The keywords and identifiersare:");
 while((c=getc(f2))!=EOF){

if(c!=' ')

str[k++]=c;

else

{

str[k]='\0';

keyword(str);

k=0;
}
}

fclose(f2);

f3=fopen("specialchar","r");

printf("\nSpecial characters are");

while((c=getc(f3))!=EOF)

printf("%c",c);

printf("\n");

fclose(f3);

printf("Total no. of lines are:%d",lineno);

}
```

1.6 INPUT & OUTPUT:

Input:

Enter Program $ for termination:

```
{
int a[3],t1,t2;
t1=2;  a[0]=1; a[1]=2; a[t1]=3;
t2=-(a[2]+t1*6)/(a[2]-t1);
if t2>5 then

print(t2);
else {
int t3;
t3=99;

t2=-25;
print(-t1+t2*t3);                    /* this is a comment        on 2 lines */

}  endif
}
$
```

Output:

Variables : a[3] t1 t2 t3

Operator : - + * / >
Constants : 2 1 3 6 5 99 -25
Keywords : int if then else endif
Special Symbols : , ; ( ) { }
Comments : this is a comment on 2 lines

1.7 LAB ASSIGNMENT

1. Write a program to recognize identifiers.
2. Write a program to recognize constants.
3. Write a program to recognize keywords and identifiers.
4. Write a program to ignore the comments in the given input source program.

EXPERIMENT-2

2.1 OBJECTIVE:

Write a C program to identify whether a given line is a comment or not.

2.2 RESOURCE: Turbo C++, Codeblock

2.3 PROGRAM LOGIC:

Read the input string. Check whether the string is starting with '/' and check next character is '/' or'*'. If condition satisfies print comment. Else not a comment.

2.4 PROCEDURE:

Go to debug -> run or press CTRL + F9 to run the program.

2.5 PROGRAM:

```
#include<stdio.h>

#include<conio.h>

void main()        {

char com[30];

int i=2,a=0;

clrscr();

printf("\n Enter comment:");

gets(com);

if(com[0]=='/')  {

if(com[1]=='/')

printf("\n It is a comment");

else if(com[1]=='*')        {

for(i=2;i<=30;i++)

{

if(com[i]=='*'&&com[i+1]=='/')

{
```

```
printf("\n It is a comment");

a=1;

break;      }

else

continue;  }

if(a==0)

printf("\n It is not a comment");

}

else

printf("\n It is not a comment");

}

else

printf("\n It is not a comment");

getch();  }
```

2.6 INPUT & OUTPUT:

Input: Enter comment: //hello
Output: It is a comment
Input: Enter comment: hello
Output: It is not a comment

2.7 LAB ASSIGNMENT

1. Write a program to recognize comments and how many letters are in this comments.
2. Write a program to recognize the types of comments.
3. Write a program to find the number of line where the comments are written.