NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA,
SURATHKAL

COMPILER LAB PROJECT-1

LEXICAL ANALYZER USING LEX

Submitted By:

Anuj Revankar (15CO109)

Rathan Kumar G (15CO139)

Test Cases

Test case 1

This Test case demonstrates bracket mismatches, nested and unterminated comments.

Input

```
int main()

{

        /* this is a comment;

        /* nested  comment

        */

}
```

Expected output

ERROR: COMMENT DOES NOT END

ERROR: NESTED COMMENT

ERROR: BRACKET MISMATCH (CLOSNG BRACKET IN COMMENT)

Test case 2

This Test case demonstrates printf() errors, string errors and bracket mismatches.

Input

```
int main()

{

	scanf("",);

	printf("",);

	printf("hi



}
```

Expected output

ERROR: SCANFERROR

ERROR: PRINTFERROR

ERROR: INCOMPLETE STRING DELARATION

ERROR: BRACKET MISMATCH

Test case 3

This Test case demonstrates unidentified tokens.

Input

```
int main()

{
```

```
        int $abc;
}
```

Expected output

ERROR: BAD TOKEN

Test case 4

This Test case has no errors. The Expected output shows the contents of the Expected output file. The Expected output file contains the symbol-constant table and information about comment lines.

Input

```
#include<stdio.h>

int main()
{
  int n, i = 3, count, c;

  printf("Enter the number of prime numbers required\n");
  scanf("%d",&n);

  if ( n >= 1 )
```

```
  {
        printf("First %d prime numbers are :\n",n);
        printf("2\n");
  }

  for ( count = 2 ; count <= n ;  )
  {
        for ( c = 2 ; c <= i - 1 ; c++ )
        {
        if ( i%c == 0 )
        break;
        }
        if ( c == i )
        {
        printf("%d\n", i);
        count++;
        }
        i++;
  }

  return 0;
}
```

Expected output

Symbol Table Format is:

Lexeme	Token	Attribute Value
#include<stdio.h>	Preprocessor Statement	0
int	Keyword	1
main	Variable	2
(	Punctuator	3
)	Punctuator	4
{	Punctuator	5
int	Keyword	1
n	Variable	6
,	Punctuator	7
i	Variable	8
=	Punctuator	9
3	Constant	10
,	Punctuator	7
count	Variable	11
,	Punctuator	7
c	Variable	12
;	Punctuator	13

printf	Keyword	14
(	Punctuator	3
"Enter the number of prime numbers required\n"	String Literal	15
)	Punctuator	4
;	Punctuator	13
scanf	Keyword	16
(	Punctuator	3
"%d"	String Literal	17
,	Punctuator	7
&	Operator	18
n	Variable	6
)	Punctuator	4
;	Punctuator	13
if	Keyword	19
(	Punctuator	3
n	Variable	6
>=	Operator	20
1	Constant	21
)	Punctuator	4
{	Punctuator	5
printf	Keyword	14
(	Punctuator	3
"First %d prime numbers are :\n"	String Literal	22

,	Punctuator	7	
n	Variable		6
)	Punctuator	4	
;	Punctuator	13	
printf	Keyword	14	
(	Punctuator	3	
"2\n"	String Literal	23	
)	Punctuator	4	
;	Punctuator	13	
}	Punctuator	24	
for	Keyword	25	
(	Punctuator	3	
count	Variable	11	
=	Punctuator	9	
2	Constant	26	
;	Punctuator	13	
count	Variable	11	
<=	Operator	27	
n	Variable		6
;	Punctuator	13	
)	Punctuator	4	
{	Punctuator	5	
for	Keyword	25	

(	Punctuator	3	
c	Variable	12	
=	Punctuator	9	
2	Constant	26	
;	Punctuator	13	
c	Variable	12	
<=	Operator	27	
i	Variable		8
-	Operator	28	
1	Constant	21	
;	Punctuator	13	
c	Variable	12	
++	Operator	29	
)	Punctuator	4	
{	Punctuator	5	
if	Keyword	19	
(	Punctuator	3	
i	Variable		8
%	Operator	30	
c	Variable	12	
==	Operator	31	
0	Constant	32	
)	Punctuator	4	

break	Keyword	33	
;	Punctuator	13	
}	Punctuator	24	
if	Keyword	19	
(	Punctuator	3	
c	Variable	12	
==	Operator	31	
i	Variable		8
)	Punctuator	4	
{	Punctuator	5	
printf	Keyword	14	
(	Punctuator	3	
"%d\n"	String Literal	34	
,	Punctuator	7	
i	Variable		8
)	Punctuator	4	
;	Punctuator	13	
count	Variable	11	
++	Operator	29	
;	Punctuator	13	
}	Punctuator	24	
i	Variable		8
++	Operator	29	

;	Punctuator	13
}	Punctuator	24
return	Keyword	35
0	Constant	32
;	Punctuator	13
}	Punctuator	24

Comment (0 lines):