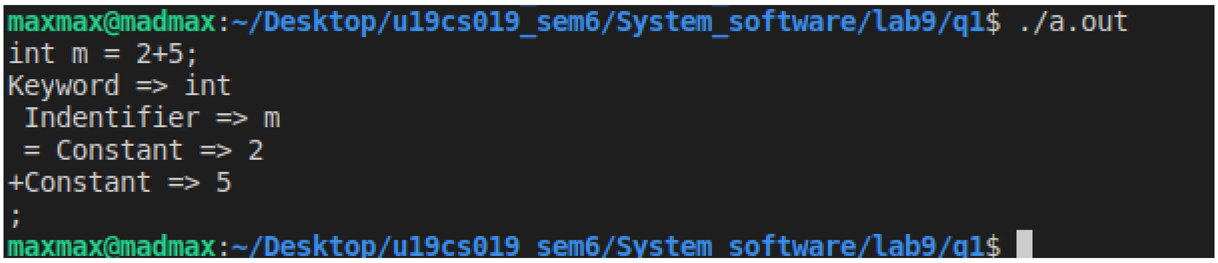
Write a lex program to identify identifiers, constants and keywords (int, float) used in c/c++ from a given
input file.

```
/*lex code to count the number of lines,
        tabs and spaces used in the input*/

%{
#include<stdio.h>
/*Global variables*/
%}

/*Rule Section*/
%%
if|else|while|do|int {printf("Keyword => %s\n", yytext);}
[a-zA-Z_]+[a-zA-Z_0-9]*{30} {printf("Indentifier => %s\n", yytext);}
[0-9]*"."{1}?[0-9]+ {printf("Constant => %s\n", yytext);}
%%
int yywrap(void) {
        return 1;
}

int main(int argc, char* argv[])
{
   yyin = fopen(argv[1], "r");
        // The function that starts the analysis
        yylex();
}
```



```
maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab9/q1$ ./a.out
int m = 2+5;
Keyword => int
 Indentifier => m
 = Constant => 2
+Constant => 5
;
maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab9/q1$
```

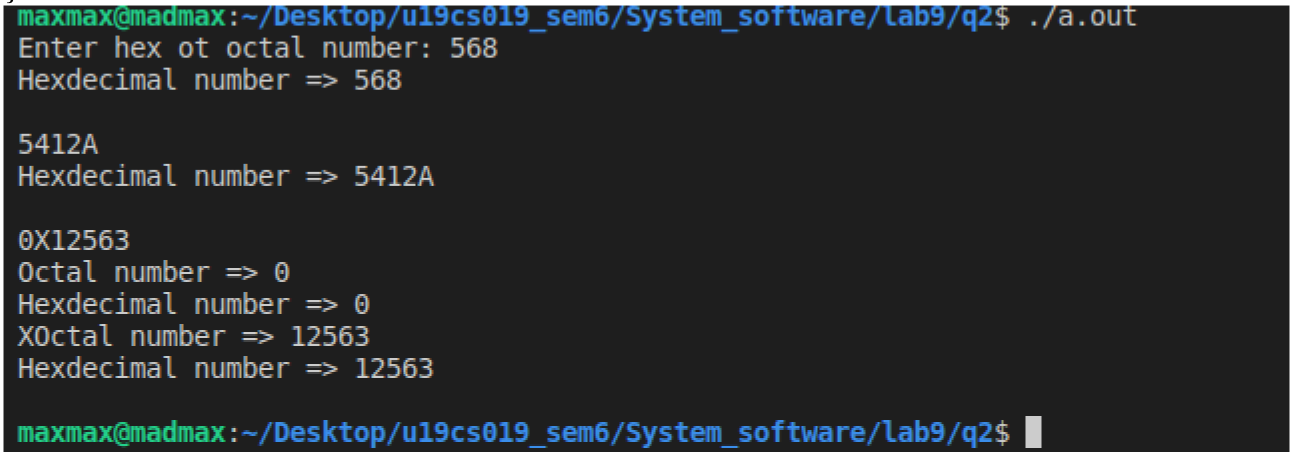
Write a lex Program to find octal and hexadecimal numbers.

```
/*lex code to count the number of lines,
        tabs and spaces used in the input*/

%{
#include<stdio.h>
/*Global variables*/
%}

/*Rule Section*/
%%
"0x"{1}[0-9A-Fa-f]+ {printf("Hexdecimal number => %s\n", yytext);}
[0-7]+ {printf("Octal number => %s\nHexdecimal number => %s\n", yytext, yytext);}
[0-9A-Fa-f]+ {printf("Hexdecimal number => %s\n", yytext);}
%%
int yywrap(void) {
        return 1;
}

int main(int argc, char* argv[])
{
   yyin = fopen(argv[1], "r");
   printf("Enter hex ot octal number: ");
        // The function that starts the analysis
        yylex();
}
```



```
maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab9/q2$ ./a.out
Enter hex ot octal number: 568
Hexdecimal number => 568

5412A
Hexdecimal number => 5412A

0X12563
Octal number => 0
Hexdecimal number => 0
XOctal number => 12563
Hexdecimal number => 12563

maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab9/q2$ 
```

Write a lex program to count and display Single line and multiline comments.

```
/*lex code to count the number of lines,
        tabs and spaces used in the input*/

%{
#include<stdio.h>
/*Global variables*/
%}

/*Rule Section*/
%%
\/\/.{1,} { printf("Single line comment: %s\n", yytext); }
\/\*[.{1,}\n{1,}]*\*\/ { printf("Multi line comment: %s\n", yytext); }
%%
int yywrap(void) {
        return 1;
}

int main(int argc, char* argv[])
{
   yyin = fopen(argv[1], "r");
        yylex();
}
```

Write a lex program to count no of negative, positive and zero numbers.

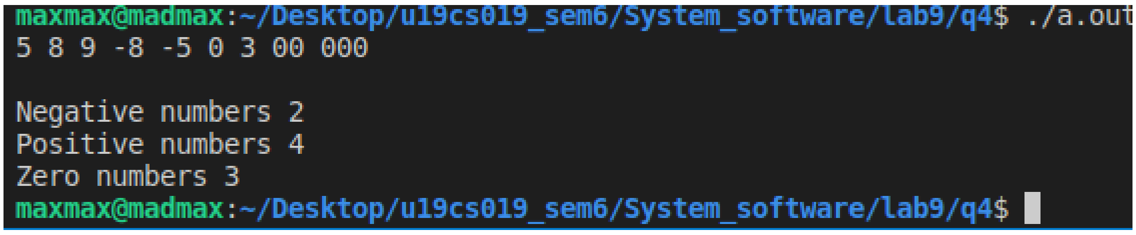
```
/*lex code to count the number of lines,
        tabs and spaces used in the input*/

%{
#include<stdio.h>
int n = 0, p = 0, z = 0;
/*Global variables*/
%}

/*Rule Section*/
%%
-[0-9]+ {n++;}
[1-9]+[0-9]* {p++;}
0+ {z++;}
%%
int yywrap(void) {
        return 1;
}

int main(int argc, char* argv[])
{
        // The function that starts the analysis
        yylex();

   printf("Negative numbers %d\n", n);
   printf("Positive numbers %d\n", p);
   printf("Zero numbers %d\n", z);
}
```



```
maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab9/q4$ ./a.out
5 8 9 -8 -5 0 3 00 000

Negative numbers 2
Positive numbers 4
Zero numbers 3
maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab9/q4$
```

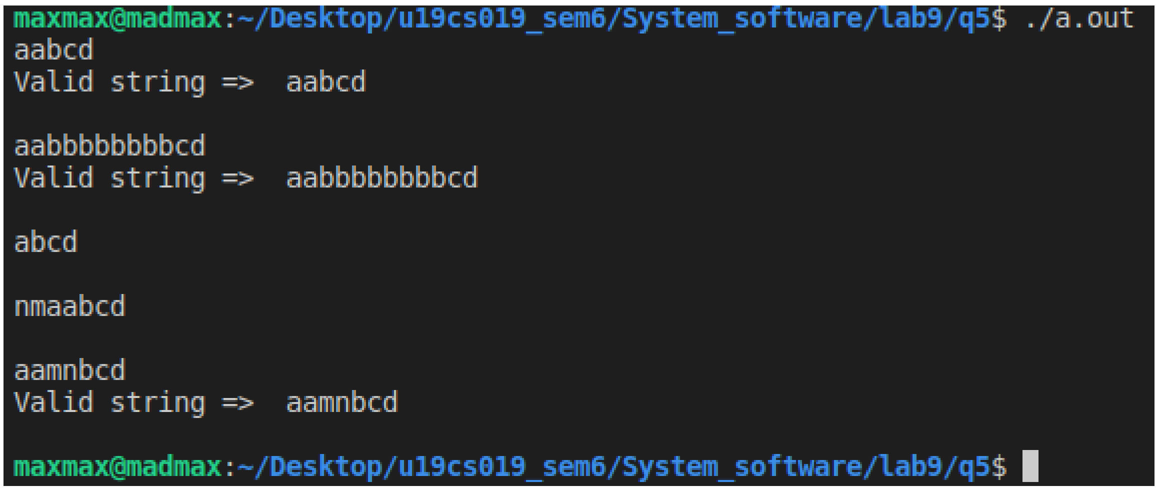
Write a Lex program to accept strings that start with aa and end with bcd .

```
/*lex code to count the number of lines,
        tabs and spaces used in the input*/

%{
#include<stdio.h>
/*Global variables*/
%}

/*Rule Section*/
%%
^aa[a-zA-Z]*bcd$ {printf("Valid string =>\t %s\n", yytext);}
. {}
%%
int yywrap(void) {
        return 1;
}

int main(int argc, char* argv[])
{
        // The function that starts the analysis
        yylex();
}
```



```
maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab9/q5$ ./a.out
aabcd
Valid string =>  aabcd

aabbbbbbbbcd
Valid string =>  aabbbbbbbbcd

abcd

nmaabcd

aamnbcd
Valid string =>  aamnbcd

maxmax@madmax:~/Desktop/u19cs019_sem6/System_software/lab9/q5$
```