Name: Brijesh Rameshbhai Rohit
Admission number: U19CS009

SS-ASSIGNMENT-09

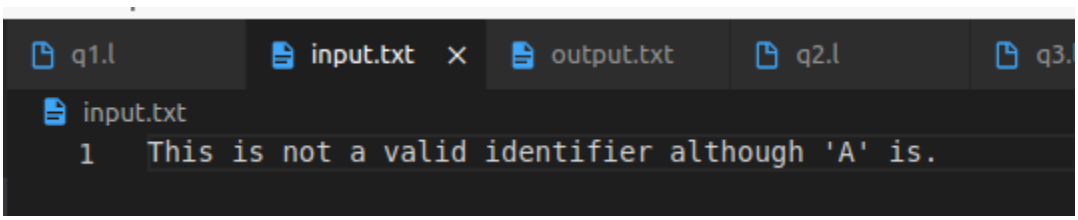
1. Write a lex program to identify identifiers, constants and keywords (int, float) used in c/c++ from a given input file.

Code:

```
%{
   #include<stdio.h>
%}

%%

['](([a-zA-z0-9])['] {fprintf(yyout,"%s\t: Constant\n",yytext);};
([0-9])+ {fprintf(yyout,"%s\t: Constant\n",yytext);};
int {fprintf(yyout,"%s\t: Keyword\n",yytext);};
float {fprintf(yyout,"%s\t: Keyword\n",yytext);};
[a-zA-Z_]([a-zA-Z0-9])* {fprintf(yyout,"%s\t: Identifier\n",yytext);};
([a-zA-Z0-9])* {fprintf(yyout,"%s\t: Invalid\n",yytext);};

%%
int main()
{
   extern FILE * yyin,*yyout;
   yyin=fopen("input.txt","r");
   yyout=fopen("output.txt","w");
   yylex();
   return 0;
}
```

Output:

Input-file :



Output-file :

q1.l | input.txt | output.txt | q2.l | q3.l

output.txt

```
This    : Identifier
 is : Identifier
 not    : Identifier
 a  : Identifier
 valid  : Identifier
 identifier : Identifier
 although   : Identifier
 'A'    : Constant
 is : Identifier
 .
```

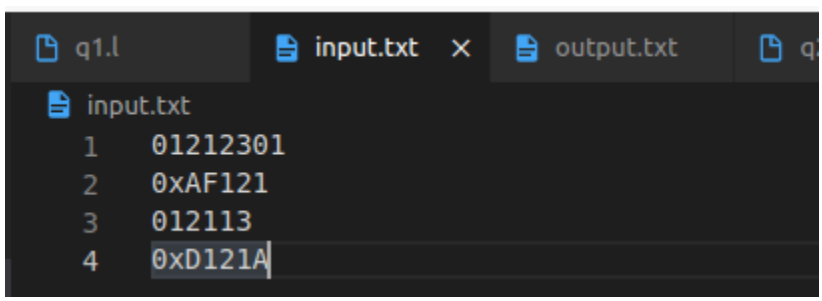
2. Write a lex Program to find octal and hexadecimal numbers.

Code:

```
%{
   #include<stdio.h>
%}

%%

[0]([0-7])+ {fprintf(yyout,"%s : Octal Number\n",yytext);};
[0][x]([0-9a-fA-F])+ {fprintf(yyout,"%s : Hexadecimal Number\n",yytext);};
([0-9])+ {fprintf(yyout,"%s : Decimal Number\n",yytext);};

%%

int main()
{
   extern FILE * yyin,*yyout;
   yyin=fopen("input.txt","r");
   yyout=fopen("output.txt","w");
   yylex();
   return 0;
}
```

Output:

Input-file :



Output-file :

q1.l | input.txt | output.txt | q2.l

output.txt

```
01212301 : Octal Number

0xAF121 : Hexadecimal Number

012113 : Octal Number

0xD121A : Hexadecimal Number
```

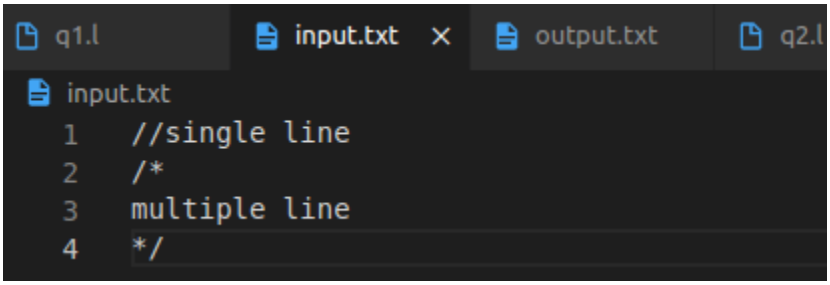
3. Write a lex program to count and display Single line and multiline comments.

Code:

```
%{
   #include<stdio.h>
   int scount=0 ,mcount=0;
%}

%%

[/][/]([a-zA-Z0-9 \t])*[\n] {fprintf(yyout,"\nSingle line Comment
:\n%s",yytext);scount++;};
[/][*]([a-zA-Z0-9 \t\n])*[*][/] {fprintf(yyout,"\nMultiline Comment
:\n%s",yytext);mcount++;};
end {fprintf(yyout,"\nSingle line comment count : %d\nMultiple Line
comment count : %d",scount,mcount);};

%%

int main()
{
   extern FILE * yyin,*yyout;
   yyin=fopen("input.txt","r");
   yyout=fopen("output.txt","w");
   yylex();
   return 0;
}
```

Output:

Input-file :



Output-file :

q1.l | input.txt | output.txt | q2.l

output.txt

```

Single line Comment :
//single line

Multiline Comment :
/*
multiple line
*/
```

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

Code:

```
%{
   #include<stdio.h>
   int nn=0,pn=0,zn=0;
%}

%%

[-][1-9]([0-9])* {nn++;};
[+][1-9]([0-9])* {pn++;};
[1-9]([0-9])* {pn++;};
([0])+ {zn++;};

%%

int main()
{
   yylex();
   printf("\nNumber of Positive Numbers: %d",pn);
   printf("\nNumber of Negative Numbers: %d",nn);
   printf("\nNumber of Zero Numbers: %d\n",zn);
   return 0;
}
```

Output:

```
brijesh@brijesh-VirtualBox:~/Documents/ss-assign09$ lex q4.l
brijesh@brijesh-VirtualBox:~/Documents/ss-assign09$ gcc lex.yy.c -lfl
brijesh@brijesh-VirtualBox:~/Documents/ss-assign09$ ./a.out
1

0

-1

1

-1

0

2

-4

6

10

-11
Number of Positive Numbers: 5
Number of Negative Numbers: 4
Number of Zero Numbers: 2
```

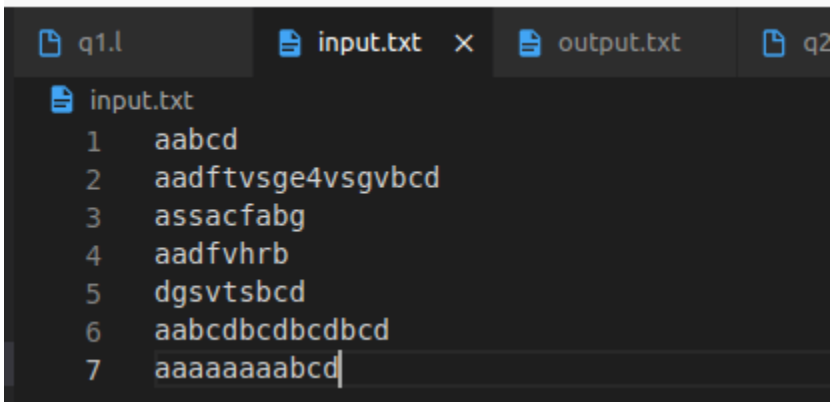
5. Write a Lex program to accept strings that start with aa and end with bcd.

Code:

```
%{
   #include<stdio.h>
%}

%%

aa([a-zA-Z])*bcd {fprintf(yyout,"%s : Accepted",yytext);};
([a-zA-Z])+ {fprintf(yyout,"%s : Declined",yytext);};

%%

int main()
{
   extern FILE * yyin,*yyout;
   yyin=fopen("input.txt","r");
   yyout=fopen("output.txt","w");
   yylex();
   return 0;
}
```

Output:

Input-file :



Output-file :

q1.l | input.txt | output.txt | q2.l

output.txt

```
aabcd : Accepted
aadftvsge4vsgvbcd : Declined
assacfabg : Declined
aadfvhrb : Declined
dgsvtsbcd : Declined
aabcdbcdbcdbcd : Accepted
aaaaaaaabcd : Accepted
```