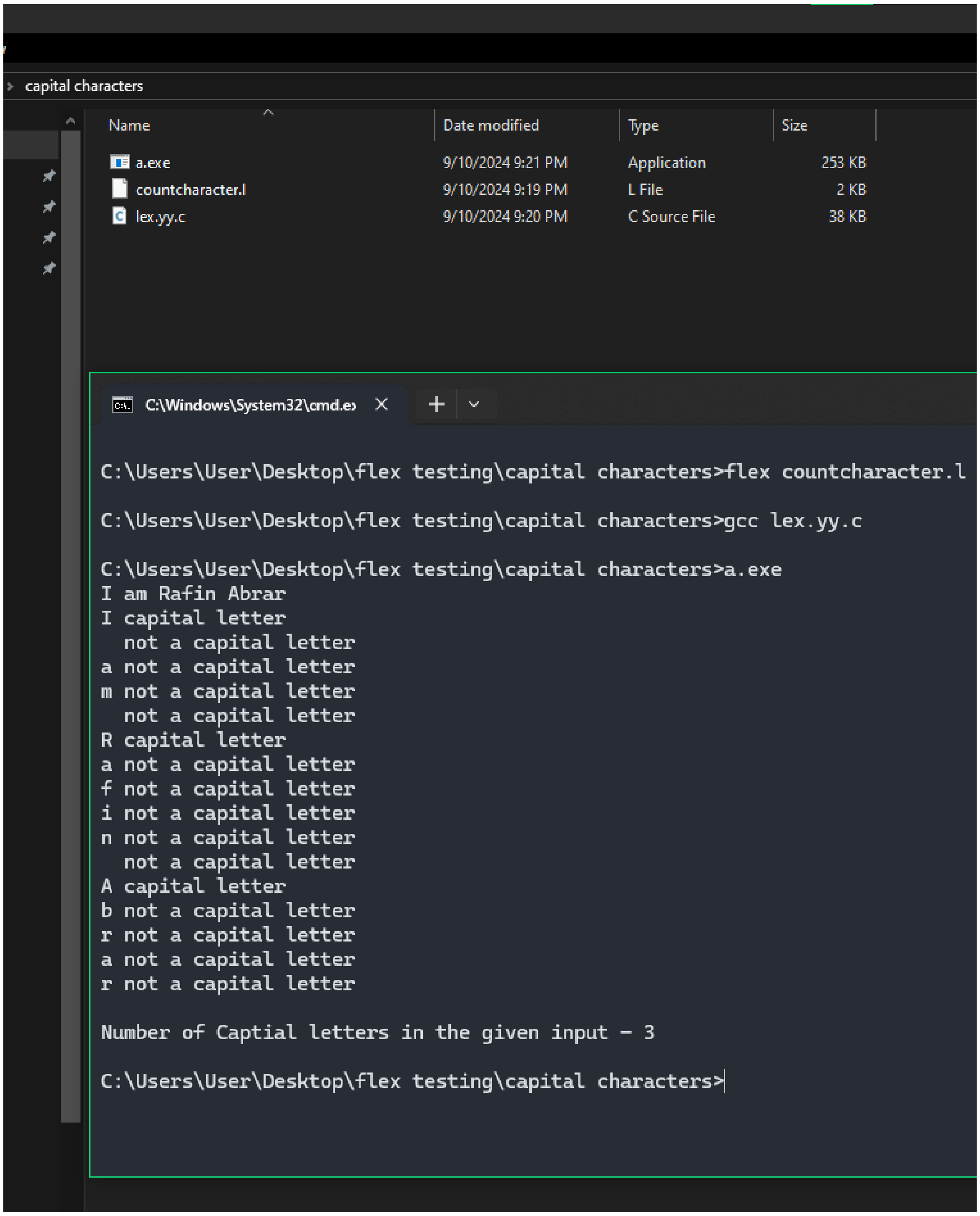
1. Counting the number of capital characters

Code:

```
/*** Definition Section has one variable
which can be accessed inside yylex()
and main() ***/
%{
int count = 0;
%}

/*** Rule Section has three rules, first rule
matches with capital letters, second rule
matches with any character except newline and
third rule does not take input after the enter***/
%%
[A-Z] {printf("%s capital letter\n", yytext);
      count++;}
.    {printf("%s not a capital letter\n", yytext);}
\n    {return 0;}
%%

/*** Code Section prints the number of
capital letter present in the given input***/
int yywrap(){}
int main(){
```

```
// Explanation:

// yywrap() - wraps the above rule section

/* yyin - takes the file pointer

        which contains the input*/

/* yylex() - this is the main flex function

        which runs the Rule Section*/

// yytext is the text in the buffer


// Uncomment the lines below

// to take input from file

// FILE *fp;

// char filename[50];

// printf("Enter the filename: \n");

// scanf("%s",filename);

// fp = fopen(filename,"r");

// yyin = fp;


yylex();

printf("\nNumber of Captial letters "

    "in the given input - %d\n", count);


return 0;

}
```

Output Screenshot:



Name	Date modified	Type	Size
a.exe	9/10/2024 9:21 PM	Application	253 KB
countcharacter.l	9/10/2024 9:19 PM	L File	2 KB
lex.yy.c	9/10/2024 9:20 PM	C Source File	38 KB

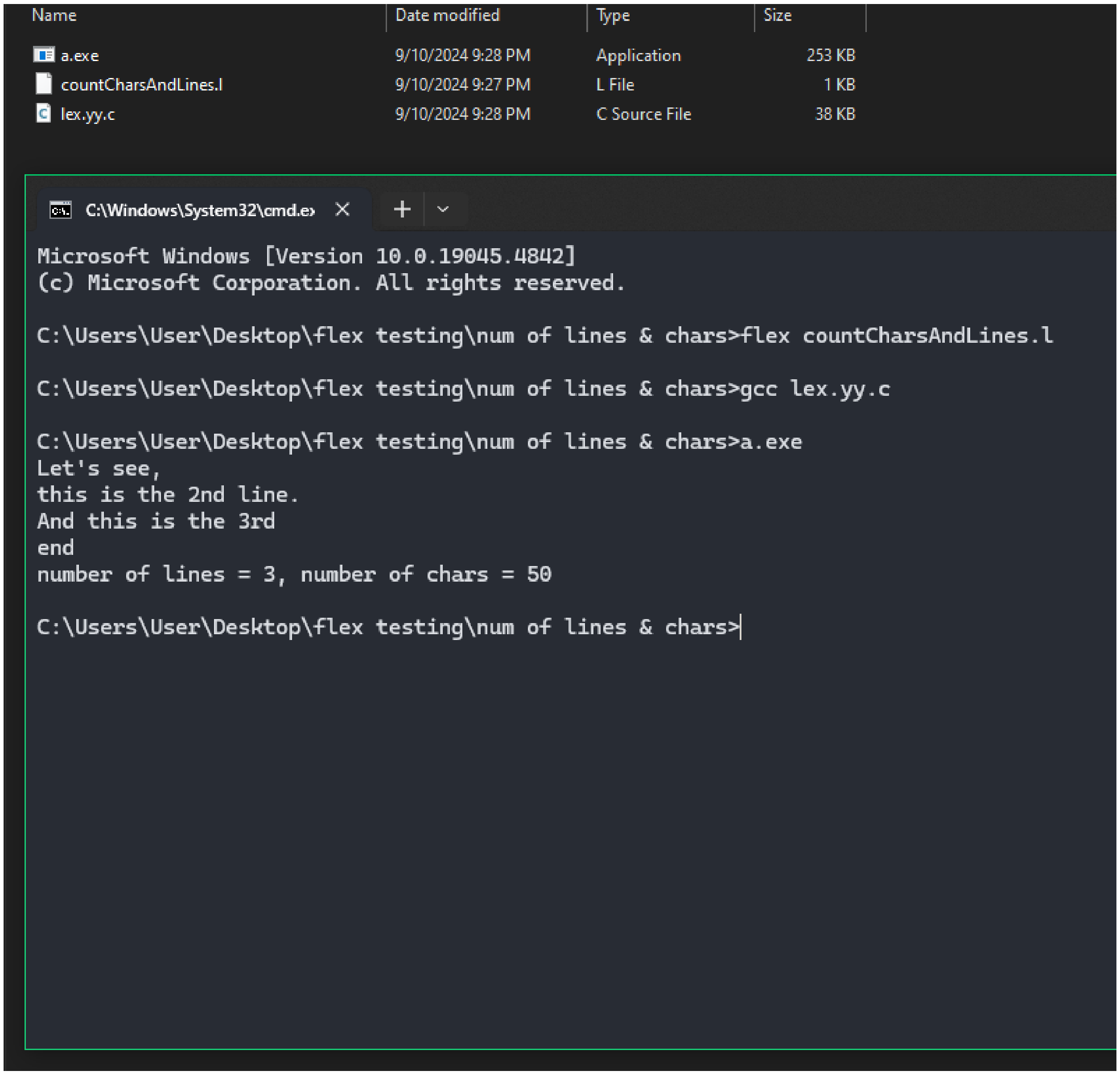
```
C:\Users\User\Desktop\flex testing\capital characters>flex countcharacter.l

C:\Users\User\Desktop\flex testing\capital characters>gcc lex.yy.c

C:\Users\User\Desktop\flex testing\capital characters>a.exe
I am Rafin Abrar
I capital letter
  not a capital letter
a not a capital letter
m not a capital letter
  not a capital letter
R capital letter
a not a capital letter
f not a capital letter
i not a capital letter
n not a capital letter
  not a capital letter
A capital letter
b not a capital letter
r not a capital letter
a not a capital letter
r not a capital letter

Number of Captial letters in the given input - 3

C:\Users\User\Desktop\flex testing\capital characters>
```

2. Counting the number of lines and characters

Code:

```
/* Decalring two counters one for number

of lines other for number of characters */

%{

int no_of_lines = 0;

int no_of_chars = 0;

%}

/***rule 1 counts the number of lines,

rule 2 counts the number of characters

and rule 3 specifies when to stop

taking input***/

%%

\n     ++no_of_lines;

.      ++no_of_chars;

end    return 0;

%%

 /*** User code section***/

int yywrap(){}

int main(int argc, char **argv)

{

 yylex();

printf("number of lines = %d, number of chars = %d\n",

      no_of_lines, no_of_chars );
return 0;

}
```

Output Screenshot:



Name	Date modified	Type	Size
a.exe	9/10/2024 9:28 PM	Application	253 KB
countCharsAndLines.l	9/10/2024 9:27 PM	L File	1 KB
lex.yy.c	9/10/2024 9:28 PM	C Source File	38 KB

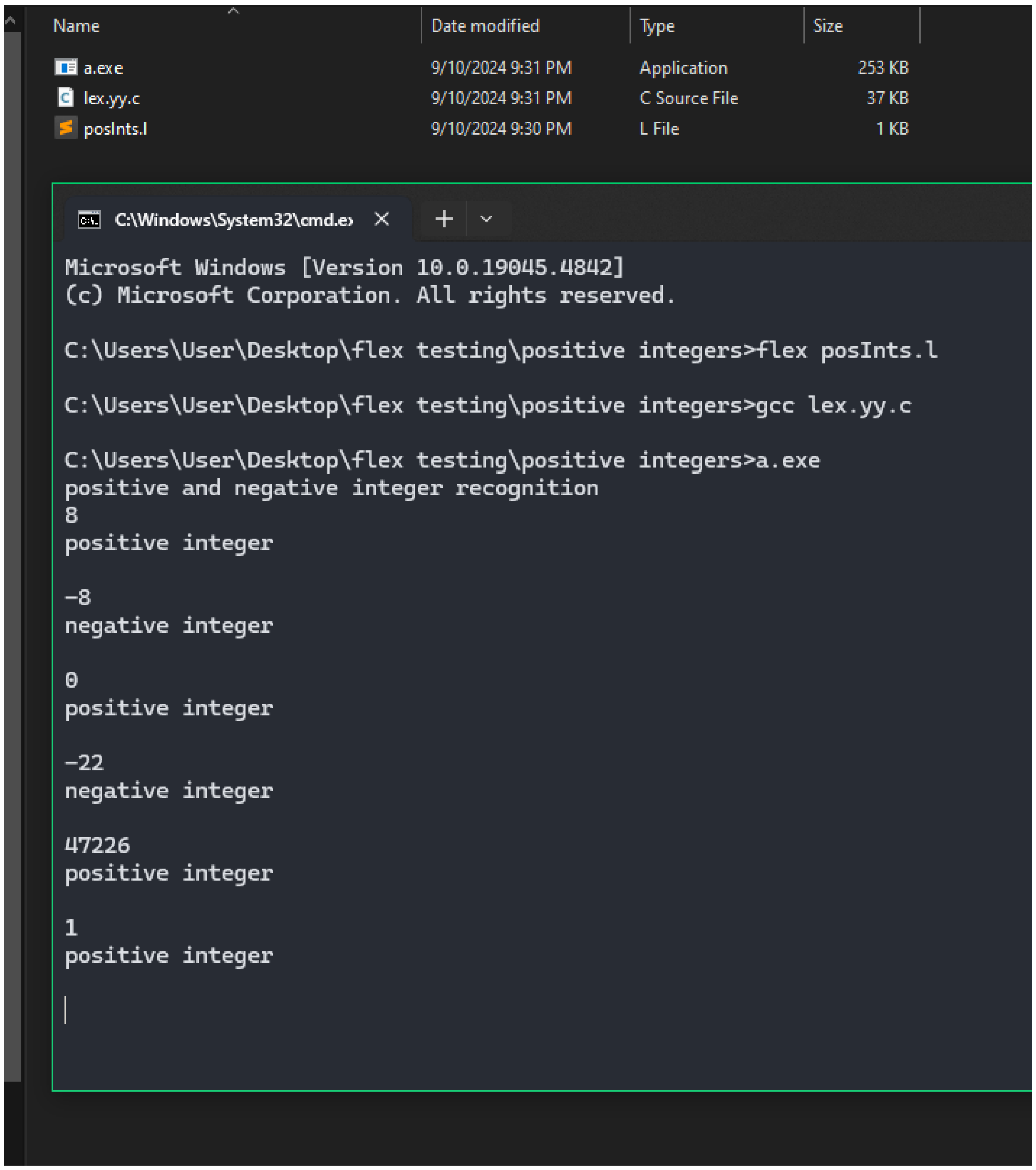
```
Microsoft Windows [Version 10.0.19045.4842]
(c) Microsoft Corporation. All rights reserved.

C:\Users\User\Desktop\flex testing\num of lines & chars>flex countCharsAndLines.l

C:\Users\User\Desktop\flex testing\num of lines & chars>gcc lex.yy.c

C:\Users\User\Desktop\flex testing\num of lines & chars>a.exe
Let's see,
this is the 2nd line.
And this is the 3rd
end
number of lines = 3, number of chars = 50

C:\Users\User\Desktop\flex testing\num of lines & chars>
```

3. Is the given number positive or negative

Code:

```
%%


[+]?[0-9]+      {printf("positive integer\n");}

[-]?[0-9]+      {printf("negative integer\n");}

.

%%


int yywrap()

{

        return 1;

}


int main()

{

printf("positive and negative integer recognition\n");

        yylex();




        return 0;

}
```

Output Screenshot:



```
a.exe        9/10/2024 9:31 PM   Application      253 KB
lex.yy.c     9/10/2024 9:31 PM   C Source File     37 KB
posInts.l    9/10/2024 9:30 PM   L File             1 KB
```

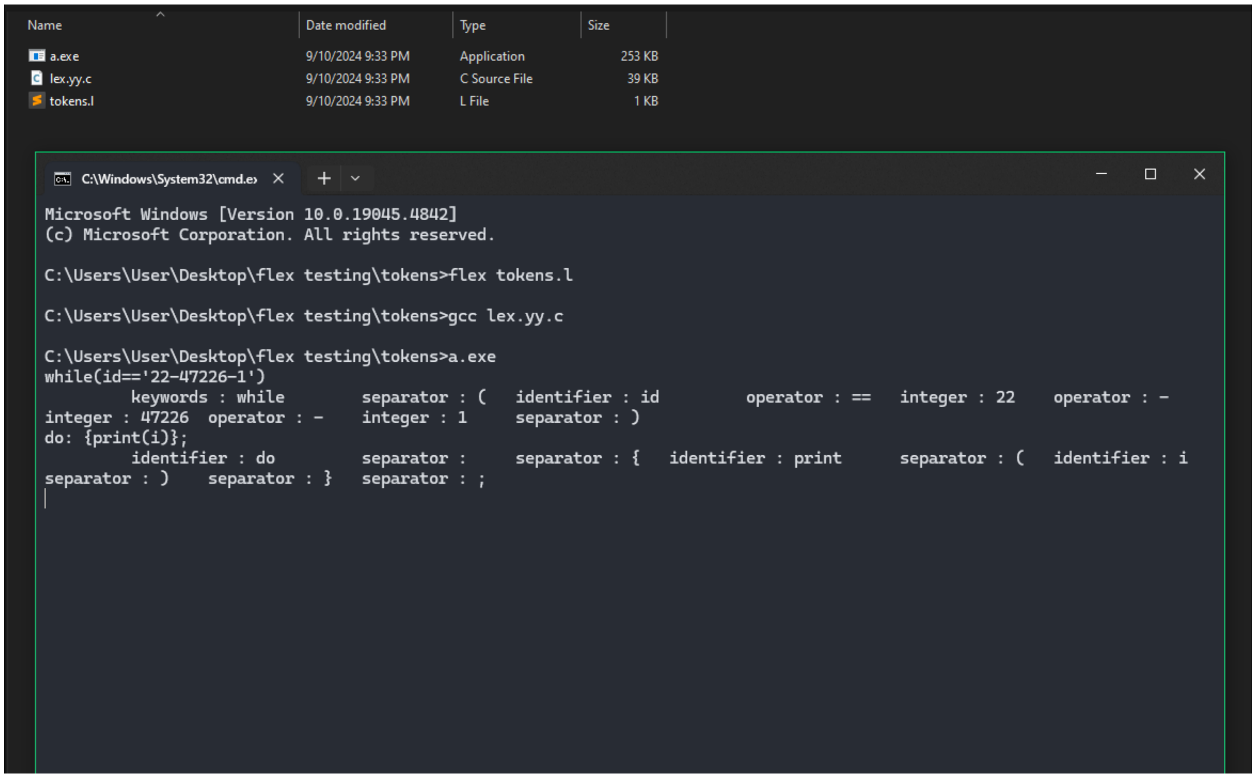
```
Microsoft Windows [Version 10.0.19045.4842]
(c) Microsoft Corporation. All rights reserved.

C:\Users\User\Desktop\flex testing\positive integers>flex posInts.l

C:\Users\User\Desktop\flex testing\positive integers>gcc lex.yy.c

C:\Users\User\Desktop\flex testing\positive integers>a.exe
positive and negative integer recognition
8
positive integer

-8
negative integer

0
positive integer

-22
negative integer

47226
positive integer

1
positive integer

|
```

4. Identifying tokens

Code:

```
%{

int n = 0 ;

%}


%%

"while"|"if"|"else" {n++;printf("\t keywords : %s", yytext);}

"int"|"float" {n++;printf("\t keywords : %s", yytext);}

"<="|"=="|"="|"++"|"-"|"*"|"+" {n++;printf("\t operator : %s", yytext);}

[a-zA-Z_][a-zA-Z0-9_]*          {n++;printf("\t identifier : %s", yytext);}

[(){}|, ;] {n++;printf("\t separator : %s", yytext);}

[0-9]*"."[0-9]+            {n++;printf("\t float : %s", yytext);}

[0-9]+  {n++;printf("\t integer : %s", yytext);}

.           ;

%%


int yywrap()

{

            return 1;

}

int main()


{
    yylex();

    printf("\n total no. of token = %d\n", n);



    return 0;



}
```

Output Screenshot:



```
Name                Date modified       Type            Size
a.exe               9/10/2024 9:33 PM   Application     253 KB
lex.yy.c            9/10/2024 9:33 PM   C Source File    39 KB
tokens.l            9/10/2024 9:33 PM   L File            1 KB
```

```
Microsoft Windows [Version 10.0.19045.4842]
(c) Microsoft Corporation. All rights reserved.

C:\Users\User\Desktop\flex testing\tokens>flex tokens.l

C:\Users\User\Desktop\flex testing\tokens>gcc lex.yy.c

C:\Users\User\Desktop\flex testing\tokens>a.exe
while(id=='22-47226-1')
         keywords : while        separator : (   identifier : id         operator : ==   integer : 22    operator : -
integer : 47226  operator : -    integer : 1     separator : )
do: {print(i)};
         identifier : do         separator :     separator : {   identifier : print      separator : (   identifier : i
separator : )    separator : }   separator : ;
```

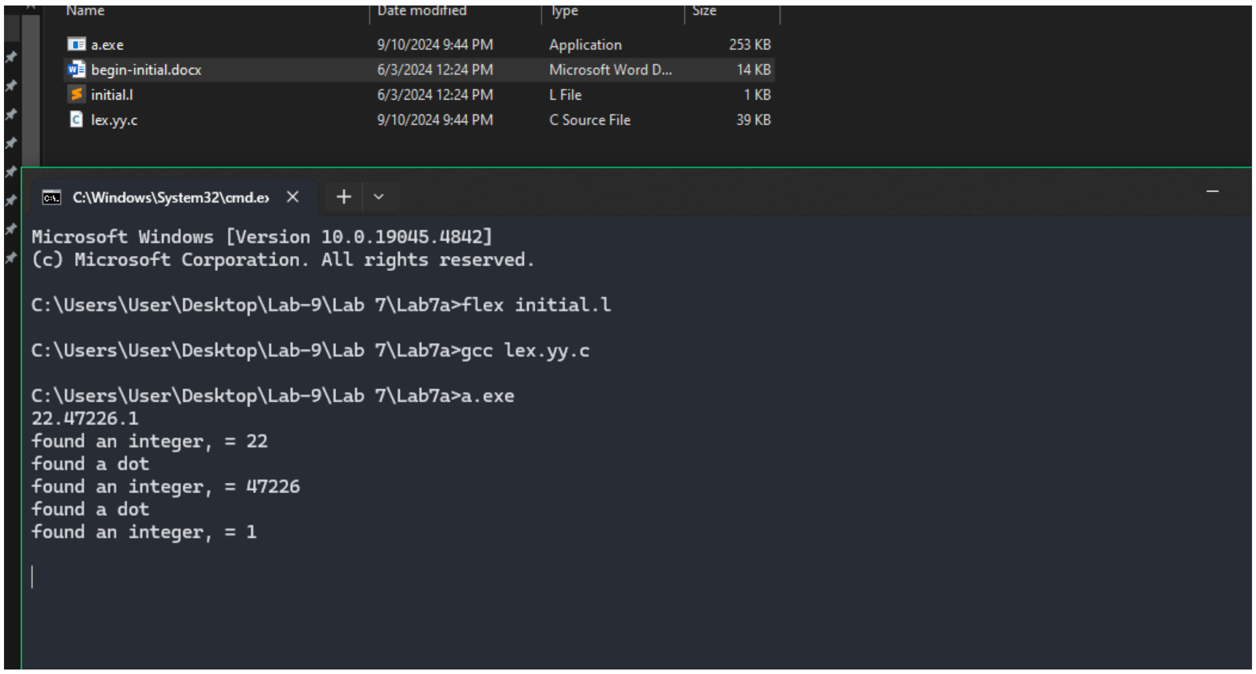
5. Identifying characters

Code:

```
%{
   #include <math.h>
/*Inclusive start condition*/
#undef yywrap
#define yywrap() 1
%}
%s expect
%%
expect-floats BEGIN(expect);
<expect>[0-9]+.[0-9]+     {
            printf( "found a float, = %f\n",
                  atof( yytext ) );
            }
<expect>\n         {
            /* that's the end of the line, so
             * we need another "expect-number"
             * before we'll recognize any more
             * numbers
             */
            BEGIN(INITIAL);
            }
```

```
[0-9]+    {
            printf( "found an integer, = %d\n",
                  atoi( yytext ) );
            }
"."       printf( "found a dot\n" );
%%
int main()
{
 yylex();
}
```

Output Screenshot:



Name	Date modified	Type	Size
a.exe	9/10/2024 9:44 PM	Application	253 KB
begin-initial.docx	6/3/2024 12:24 PM	Microsoft Word D...	14 KB
initial.l	6/3/2024 12:24 PM	L File	1 KB
lex.yy.c	9/10/2024 9:44 PM	C Source File	39 KB

```
Microsoft Windows [Version 10.0.19045.4842]
(c) Microsoft Corporation. All rights reserved.

C:\Users\User\Desktop\Lab-9\Lab 7\Lab7a>flex initial.l

C:\Users\User\Desktop\Lab-9\Lab 7\Lab7a>gcc lex.yy.c

C:\Users\User\Desktop\Lab-9\Lab 7\Lab7a>a.exe
22.47226.1
found an integer, = 22
found a dot
found an integer, = 47226
found a dot
found an integer, = 1
```

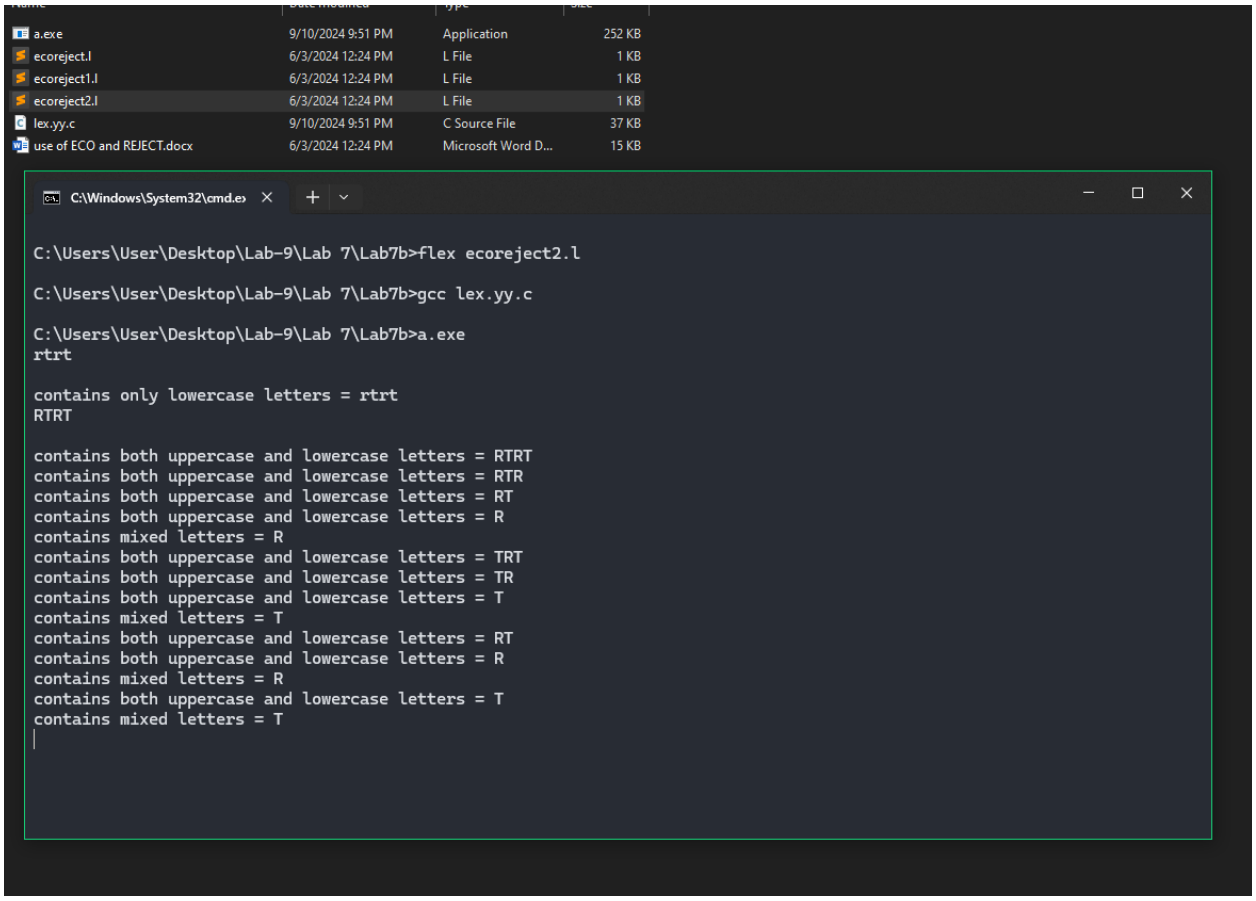
6. Showcasing ECHO & REJECT

Code:

```
%{
/*USE OF REJECT STATEMENT*/
#undef yywrap
#define yywrap() 1
%}
%%
[a-z]+ {
 printf("\ncontains only lowercase letters = ");
 ECHO;
 }
[a-zA-Z]+ {
 printf("\ncontains both uppercase and lowercase letters = ");
 ECHO;
 REJECT;


 }
 {
 printf("\ncontains mixed letters = ");
 ECHO;
 }
 %%
```

```
int main()

{

 yylex();

}
```

Output Screenshot:



```
a.exe                        9/10/2024 9:51 PM   Application         252 KB
ecoreject.l                  6/3/2024 12:24 PM   L File                1 KB
ecoreject1.l                 6/3/2024 12:24 PM   L File                1 KB
ecoreject2.l                 6/3/2024 12:24 PM   L File                1 KB
lex.yy.c                     9/10/2024 9:51 PM   C Source File        37 KB
use of ECO and REJECT.docx   6/3/2024 12:24 PM   Microsoft Word D...  15 KB
```

```
C:\Users\User\Desktop\Lab-9\Lab 7\Lab7b>flex ecoreject2.l

C:\Users\User\Desktop\Lab-9\Lab 7\Lab7b>gcc lex.yy.c

C:\Users\User\Desktop\Lab-9\Lab 7\Lab7b>a.exe
rtrt

contains only lowercase letters = rtrt
RTRT

contains both uppercase and lowercase letters = RTRT
contains both uppercase and lowercase letters = RTR
contains both uppercase and lowercase letters = RT
contains both uppercase and lowercase letters = R
contains mixed letters = R
contains both uppercase and lowercase letters = TRT
contains both uppercase and lowercase letters = TR
contains both uppercase and lowercase letters = T
contains mixed letters = T
contains both uppercase and lowercase letters = RT
contains both uppercase and lowercase letters = R
contains mixed letters = R
contains both uppercase and lowercase letters = T
contains mixed letters = T
```

7. Inclusive

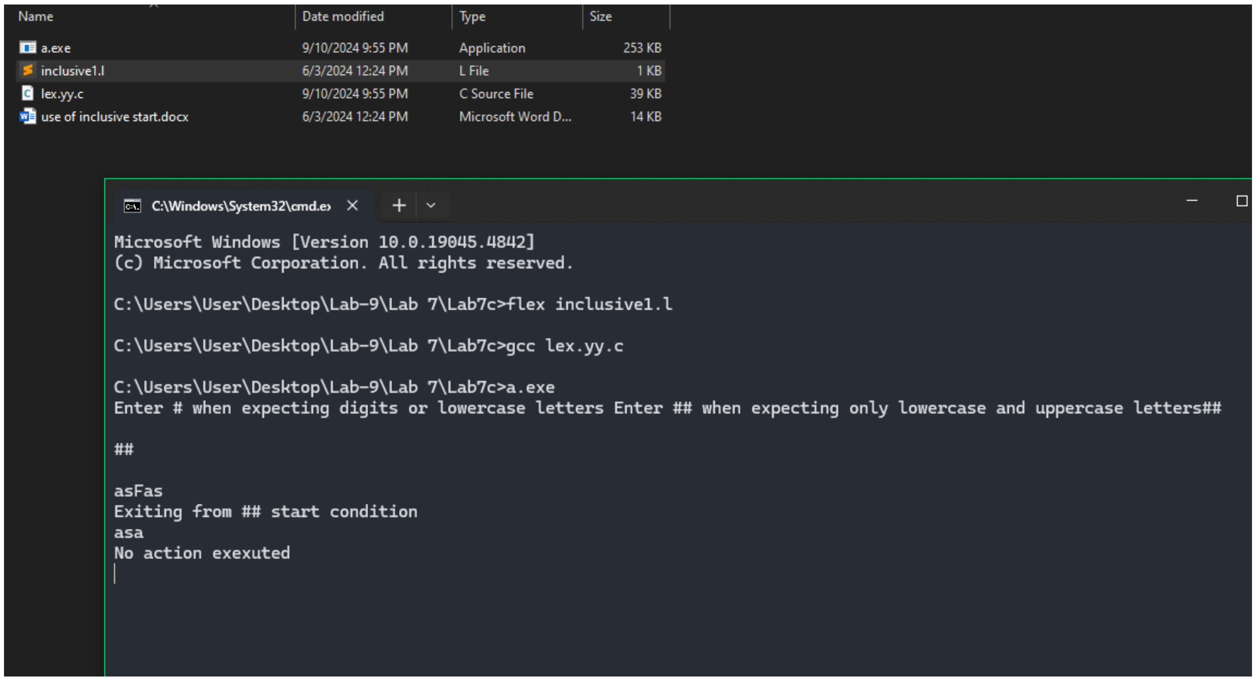
Code:

```
%{

/*Inclusive start condition*/
#undef yywrap
#define yywrap() 1

%}


%s SM SMBG


%%


# BEGIN(SM);
## BEGIN(SMBG);

[0-9]+ {
 printf("Contains only digits");
}
```

```
<SMBG>[A-Z]+ {
 printf("Contains uppercase letters");
 }

<SM>. {
 printf("Exiting from # start condition");
 BEGIN(INITIAL);
}

<SM,SMBG>[a-z]+ {
 printf("Contains lowercase letters");


}

<SMBG>.+ {
 printf("Exiting from ## start condition");
 BEGIN(INITIAL);
}

.+ {
 printf("No action exexuted");


}
```

```
%%



int main()

{

 printf("Enter # when expecting digits or lowercase letters");

 printf(" Enter ## when expecting only lowercase and uppercase letters");

 yylex();

}
```

Output Screenshot:



```
Name                      Date modified        Type                 Size
a.exe                     9/10/2024 9:55 PM    Application          253 KB
inclusive1.l              6/3/2024 12:24 PM    L File                 1 KB
lex.yy.c                  9/10/2024 9:55 PM    C Source File         39 KB
use of inclusive start.docx 6/3/2024 12:24 PM  Microsoft Word D...   14 KB
```

```
Microsoft Windows [Version 10.0.19045.4842]
(c) Microsoft Corporation. All rights reserved.

C:\Users\User\Desktop\Lab-9\Lab 7\Lab7c>flex inclusive1.l

C:\Users\User\Desktop\Lab-9\Lab 7\Lab7c>gcc lex.yy.c

C:\Users\User\Desktop\Lab-9\Lab 7\Lab7c>a.exe
Enter # when expecting digits or lowercase letters Enter ## when expecting only lowercase and uppercase letters##

##

asFas
Exiting from ## start condition
asa
No action exexuted
```