192311044

Girija B

CSA1428 - Compiler Design

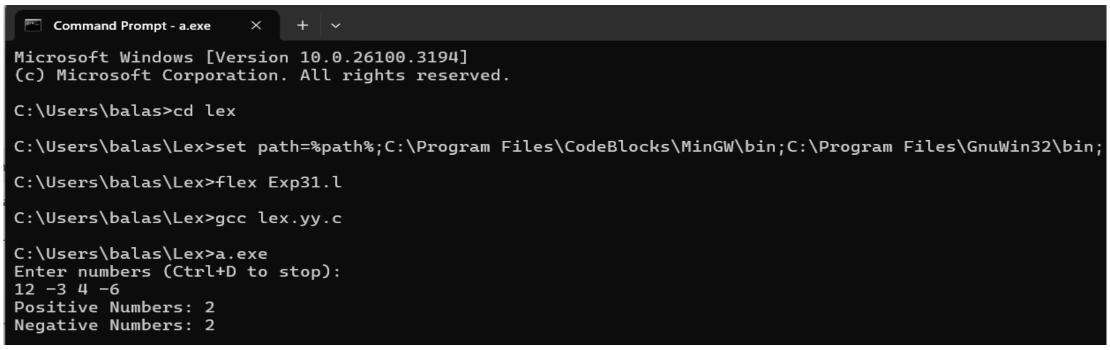
LAB ACTIVITY-6

1. Write a LEX program to identify and count positive and negative numbers.

Code (Lex):

```
%{
#include <stdio.h>
int pos_count = 0, neg_count = 0;
%}
%%
[+-]?[0-9]+(\.[0-9]+)? {
    if(yytext[0] == '-')
        neg_count++;
    else
        pos_count++;
}
\n  { printf("Positive Numbers: %d\nNegative Numbers: %d\n", pos_count, neg_count); }
.   ;
%%
int main() {
    printf("Enter numbers (Ctrl+D to stop):\n");
    yylex();
    return 0;
}
int yywrap() {
    return 1;
}
```

Output:



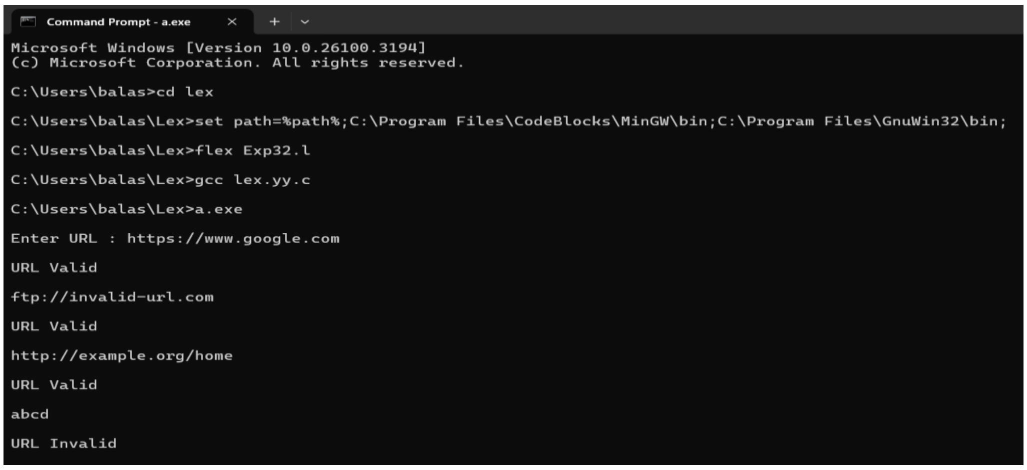
```
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved.

C:\Users\balas>cd lex

C:\Users\balas\Lex>set path=%path%;C:\Program Files\CodeBlocks\MinGW\bin;C:\Program Files\GnuWin32\bin;

C:\Users\balas\Lex>flex Exp31.l

C:\Users\balas\Lex>gcc lex.yy.c

C:\Users\balas\Lex>a.exe
Enter numbers (Ctrl+D to stop):
12 -3 4 -6
Positive Numbers: 2
Negative Numbers: 2
```

2. A networking company wants to validate the URL for their clients. Write a LEX program to implement the same.

Code (Lex):

```
%%
((http)|(ftp))s?:\/\/[a-zA-Z0-9](.[a-z])+(.[a-zA-Z0-9+=?]*)* {printf("\nURL Valid\n");}
.+ {printf("\nURL Invalid\n");}
%%
void main()
{
printf("\nEnter URL : ");
yylex();
printf("\n");
}
int yywrap()
{
}
```

Output:



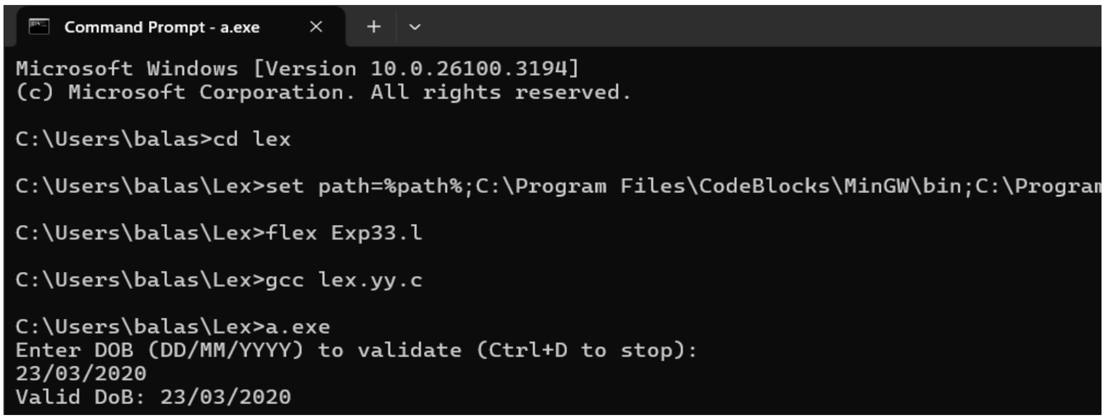
```
Command Prompt - a.exe
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved.

C:\Users\balas>cd lex

C:\Users\balas\Lex>set path=%path%;C:\Program Files\CodeBlocks\MinGW\bin;C:\Program Files\GnuWin32\bin;

C:\Users\balas\Lex>flex Exp32.l

C:\Users\balas\Lex>gcc lex.yy.c

C:\Users\balas\Lex>a.exe

Enter URL : https://www.google.com

URL Valid

ftp://invalid-url.com

URL Valid

http://example.org/home

URL Valid

abcd

URL Invalid
```

3. School management wants to validate DOB of all students. Write a LEX program to implement it.

Code (Lex):

```
%{
#include <stdio.h>
%}
%%
((0[1-9])|([1-2][0-9])|(3[0-1]))\/((0[1-9])|(1[0-2]))\/(19[0-9]{2}|2[0-9]{3}) {
   printf("Valid DoB: %s\n", yytext);
}
.* {
   printf("Invalid DoB: %s\n", yytext);
}
%%
int main() {
   printf("Enter DOB (DD/MM/YYYY) to validate (Ctrl+D to stop):\n");
   yylex();
   return 0;
}
int yywrap() {
   return 1;
}
```

Output:



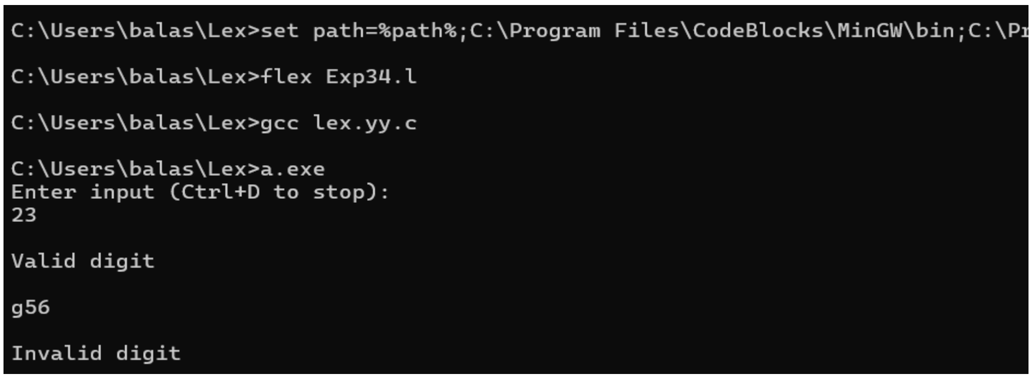
```
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved.

C:\Users\balas>cd lex

C:\Users\balas\Lex>set path=%path%;C:\Program Files\CodeBlocks\MinGW\bin;C:\Progran

C:\Users\balas\Lex>flex Exp33.l

C:\Users\balas\Lex>gcc lex.yy.c

C:\Users\balas\Lex>a.exe
Enter DOB (DD/MM/YYYY) to validate (Ctrl+D to stop):
23/03/2020
Valid DoB: 23/03/2020
```

4. Write a LEX program to check whether the given input is digit or not.

Code (Lex):

```
%{
#include <stdio.h>
%}
%%
[0-9]+ { printf("\nValid digit\n"); }
.* { printf("\nInvalid digit\n"); }
%%
int yywrap() {
   return 1;
}
int main() {
   printf("Enter input (Ctrl+D to stop):\n");
   yylex();
   return 0;
}
```

Output:



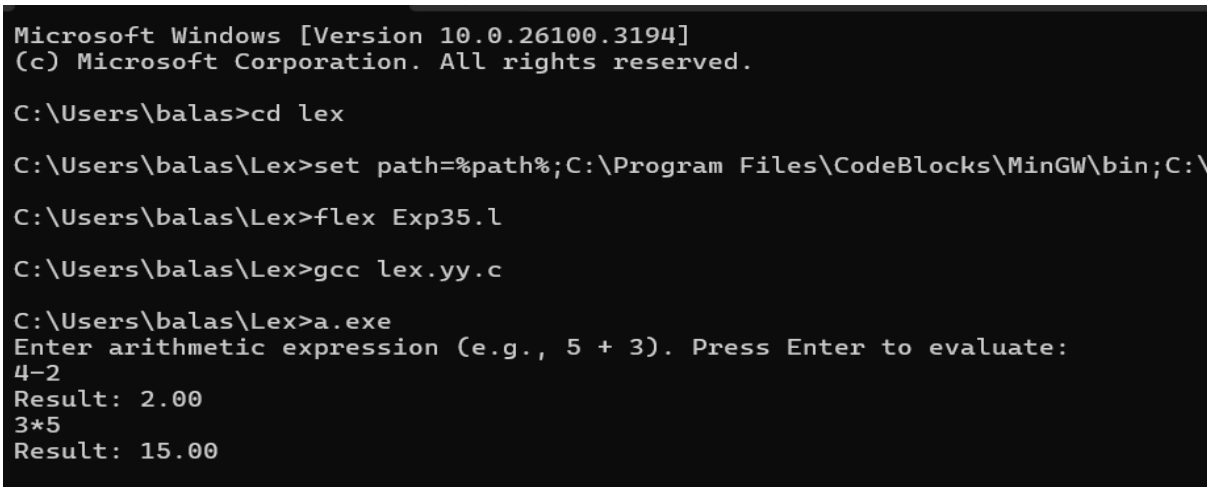
```
C:\Users\balas\Lex>set path=%path%;C:\Program Files\CodeBlocks\MinGW\bin;C:\Pr

C:\Users\balas\Lex>flex Exp34.l

C:\Users\balas\Lex>gcc lex.yy.c

C:\Users\balas\Lex>a.exe
Enter input (Ctrl+D to stop):
23

Valid digit

g56

Invalid digit
```

5. A School student was asked to do basic mathematical operations. Implement a LEX program to implement the same.

Code (Lex):

```
%{
#include <stdio.h>
#include <stdlib.h>
#undef yywrap
#define yywrap() 1
int f1 = 0, f2 = 0;
char oper;
float op1 = 0, op2 = 0, ans = 0;
void eval();
%}
DIGIT [0-9]
NUM {DIGIT}+(\.{DIGIT}+)?
OP [*/+-]
%%
{NUM} {
   if (f1 == 0) {
      op1 = atof(yytext);
      f1 = 1;
   } else {
      op2 = atof(yytext);
      f2 = 1;
   }
}
{OP} {
    oper = yytext[0];
}
\n {
    if (f1 && f2) {
        eval();
        printf("Result: %.2f\n", ans);
        f1 = f2 = 0; // Reset for next input
    }
}
%%
void eval() {
    switch(oper) {
        case '+': ans = op1 + op2; break;
        case '-': ans = op1 - op2; break;
        case '*': ans = op1 * op2; break;
        case '/':
            if (op2 != 0)
                ans = op1 / op2;
            else
                printf("Error: Division by zero\n");
            break;
        default:
            printf("Invalid operator\n");
    }
}
int main() {
    printf("Enter arithmetic expression (e.g., 5 + 3). Press Enter to evaluate:\n");
    yylex();
    return 0;
}
```

Output:



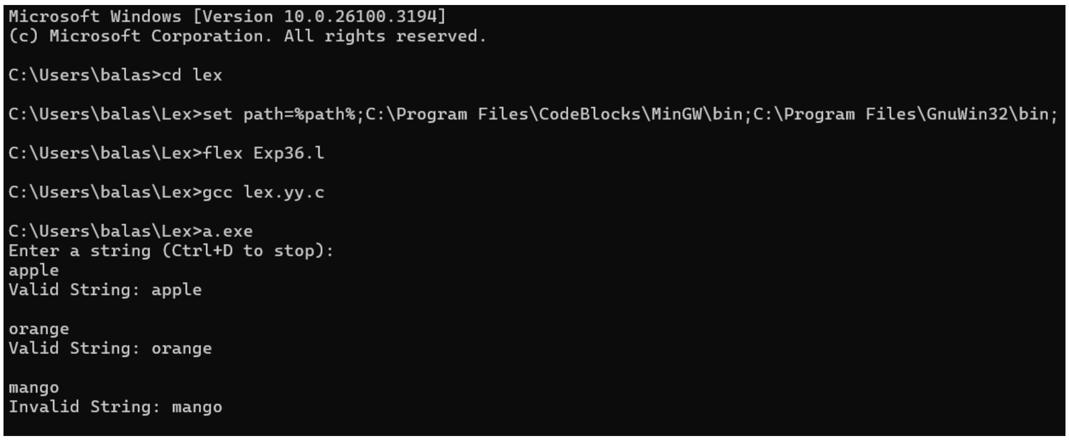
```
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved.

C:\Users\balas>cd lex

C:\Users\balas\Lex>set path=%path%;C:\Program Files\CodeBlocks\MinGW\bin;C:\

C:\Users\balas\Lex>flex Exp35.l

C:\Users\balas\Lex>gcc lex.yy.c

C:\Users\balas\Lex>a.exe
Enter arithmetic expression (e.g., 5 + 3). Press Enter to evaluate:
4-2
Result: 2.00
3*5
Result: 15.00
```

6. Write a LEX program to accept string starting with vowel.

Code (Lex):

```
%{
#include <stdio.h>
%}
%%
^[AEIOUaeiou][a-zA-Z]* { printf("Valid String: %s\n", yytext); }
.* { printf("Invalid String: %s\n", yytext); }
%%
int main() {
    printf("Enter a string (Ctrl+D to stop):\n");
    yylex();
    return 0;
}
int yywrap() {
    return 1;
}
```

Output:



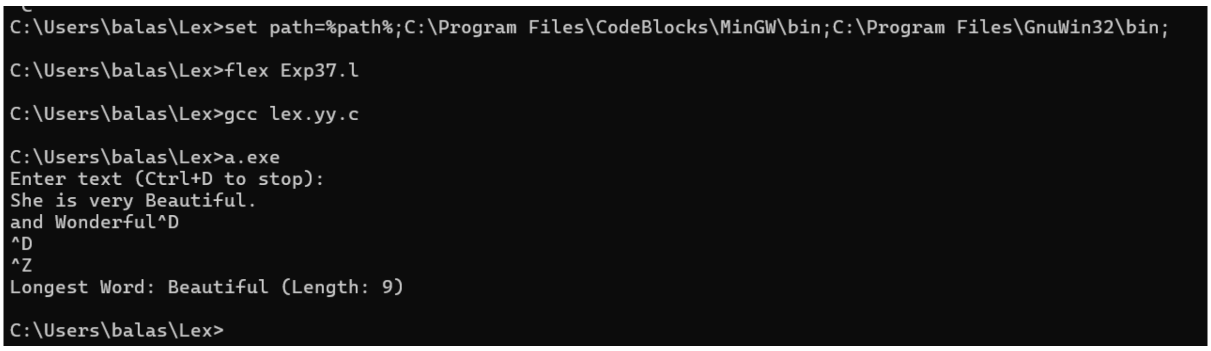
```
Microsoft Windows [Version 10.0.26100.3194]
(c) Microsoft Corporation. All rights reserved.

C:\Users\balas>cd lex

C:\Users\balas\Lex>set path=%path%;C:\Program Files\CodeBlocks\MinGW\bin;C:\Program Files\GnuWin32\bin;

C:\Users\balas\Lex>flex Exp36.l

C:\Users\balas\Lex>gcc lex.yy.c

C:\Users\balas\Lex>a.exe
Enter a string (Ctrl+D to stop):
apple
Valid String: apple

orange
Valid String: orange

mango
Invalid String: mango
```

7. Write a LEX program to find the length of the longest word.

Code (Lex):

```
%{
#include <stdio.h>
#include <string.h>
int max_length = 0;
char longest_word[100]; // Assuming words won't exceed 100 characters
%}
%%
[a-zA-Z]+ {
    int len = strlen(yytext);
    if (len > max_length) {
        max_length = len;
        strcpy(longest_word, yytext);
    }
}
[^a-zA-Z]+ { /* Ignore non-word characters */ }
%%
int main() {
    printf("Enter text (Ctrl+D to stop):\n");
    yylex();
    printf("Longest Word: %s (Length: %d)\n", longest_word, max_length);
    return 0;
}
int yywrap() {
    return 1;
}
```

Output:



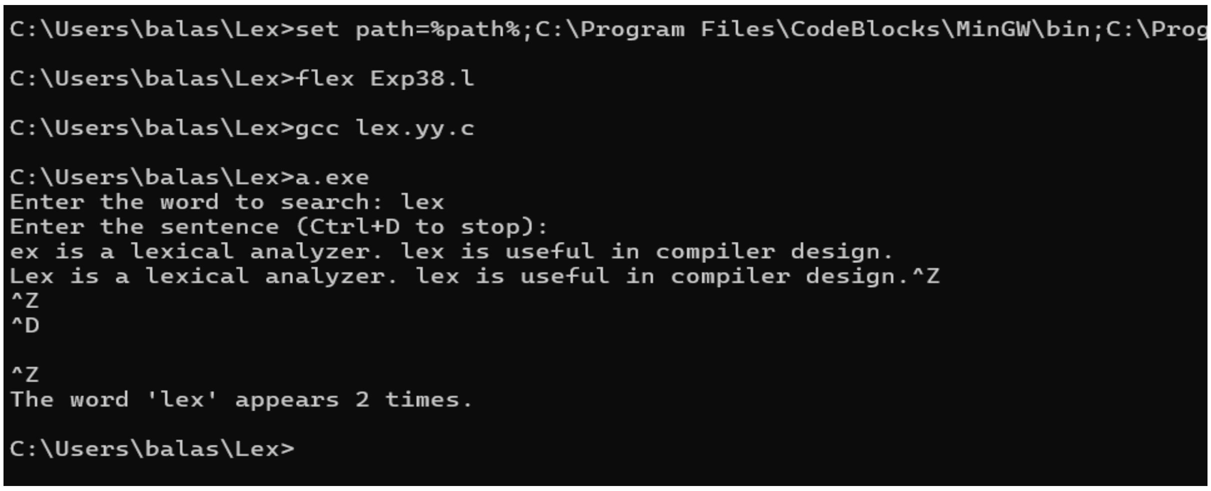
```
C:\Users\balas\Lex>set path=%path%;C:\Program Files\CodeBlocks\MinGW\bin;C:\Program Files\GnuWin32\bin;

C:\Users\balas\Lex>flex Exp37.l

C:\Users\balas\Lex>gcc lex.yy.c

C:\Users\balas\Lex>a.exe
Enter text (Ctrl+D to stop):
She is very Beautiful.
and Wonderful^D
^D
^Z
Longest Word: Beautiful (Length: 9)

C:\Users\balas\Lex>
```

8. Write a LEX program to count the frequency of the given word in a given sentence.

Code (Lex):

```
%{
#include <stdio.h>
#include <string.h>
int count = 0;
char target[100];  // Word to search for
%}
%%
[a-zA-Z]+ {
    if (strcmp(yytext, target) == 0) {
        count++;
    }
}
.|\n { /* Ignore other characters */ }
%%
int main() {
    printf("Enter the word to search: ");
    scanf("%s", target);
    printf("Enter the sentence (Ctrl+D to stop):\n");
    yylex();
    printf("The word '%s' appears %d times.\n", target, count);
    return 0;
}
int yywrap() {
    return 1;
}
```

Output:



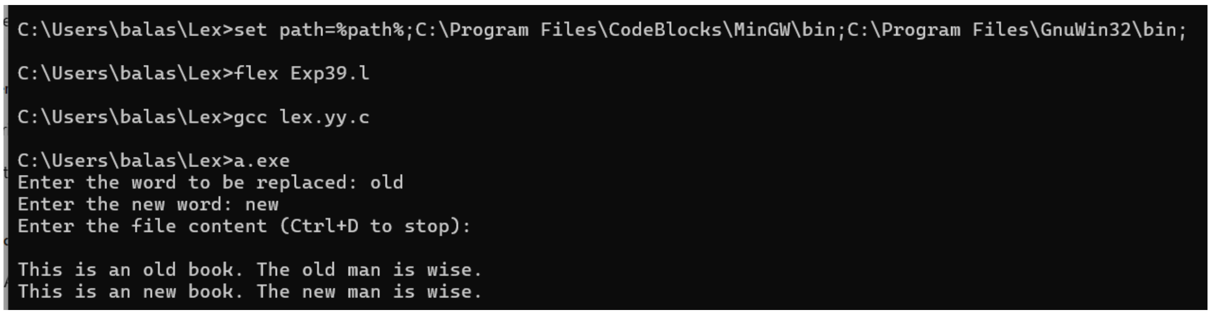
```
C:\Users\balas\Lex>set path=%path%;C:\Program Files\CodeBlocks\MinGW\bin;C:\Prog

C:\Users\balas\Lex>flex Exp38.l

C:\Users\balas\Lex>gcc lex.yy.c

C:\Users\balas\Lex>a.exe
Enter the word to search: lex
Enter the sentence (Ctrl+D to stop):
ex is a lexical analyzer. lex is useful in compiler design.
Lex is a lexical analyzer. lex is useful in compiler design.^Z
^Z
^D

^Z
The word 'lex' appears 2 times.

C:\Users\balas\Lex>
```

9. Write a LEX code to replace a word with another word in a file.

Code (Lex):

```
%{
#include <stdio.h>
#include <string.h>
char old_word[100], new_word[100]; // Words for replacement
%}
%%
[a-zA-Z]+ {
    if (strcmp(yytext, old_word) == 0) {
        printf("%s", new_word);  // Replace old word with new word
    } else {
        printf("%s", yytext); // Print the word as it is
    }
}
. { printf("%c", yytext[0]); } // Print other characters (punctuation, spaces, etc.)
%%
int main() {
    printf("Enter the word to be replaced: ");
    scanf("%s", old_word);
    printf("Enter the new word: ");
    scanf("%s", new_word);
    printf("Enter the file content (Ctrl+D to stop):\n");
    yylex(); // Process the file
    return 0;
}
int yywrap() {
    return 1;
}
```

Output:



```
C:\Users\balas\Lex>set path=%path%;C:\Program Files\CodeBlocks\MinGW\bin;C:\Program Files\GnuWin32\bin;

C:\Users\balas\Lex>flex Exp39.l

C:\Users\balas\Lex>gcc lex.yy.c

C:\Users\balas\Lex>a.exe
Enter the word to be replaced: old
Enter the new word: new
Enter the file content (Ctrl+D to stop):

This is an old book. The old man is wise.
This is an new book. The new man is wise.
```

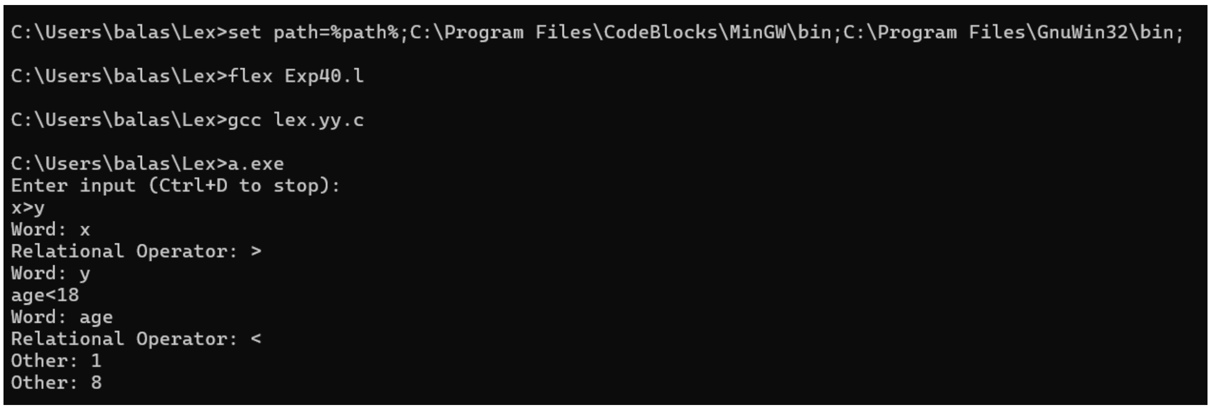
10. Write a LEX program to recognize a word and relational operator.

Code (Lex):

```
%{
#include <stdio.h>
%}
%%
[a-zA-Z_][a-zA-Z0-9_]* { printf("Word: %s\n", yytext); }
(<=|>=|==|!=|<|>) { printf("Relational Operator: %s\n", yytext); }
[ \t\n] { /* Ignore whitespace */ }
```

```
. { printf("Other: %s\n", yytext); } // Print other symbols if needed
%%
int main() {
    printf("Enter input (Ctrl+D to stop):\n");
    yylex();
    return 0;
}
int yywrap() {
    return 1;
}
```

Output:



```
C:\Users\balas\Lex>set path=%path%;C:\Program Files\CodeBlocks\MinGW\bin;C:\Program Files\GnuWin32\bin;

C:\Users\balas\Lex>flex Exp40.l

C:\Users\balas\Lex>gcc lex.yy.c

C:\Users\balas\Lex>a.exe
Enter input (Ctrl+D to stop):
x>y
Word: x
Relational Operator: >
Word: y
age<18
Word: age
Relational Operator: <
Other: 1
Other: 8
```