

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CD Practical 2

1) Write a lex program to count number of words and digit

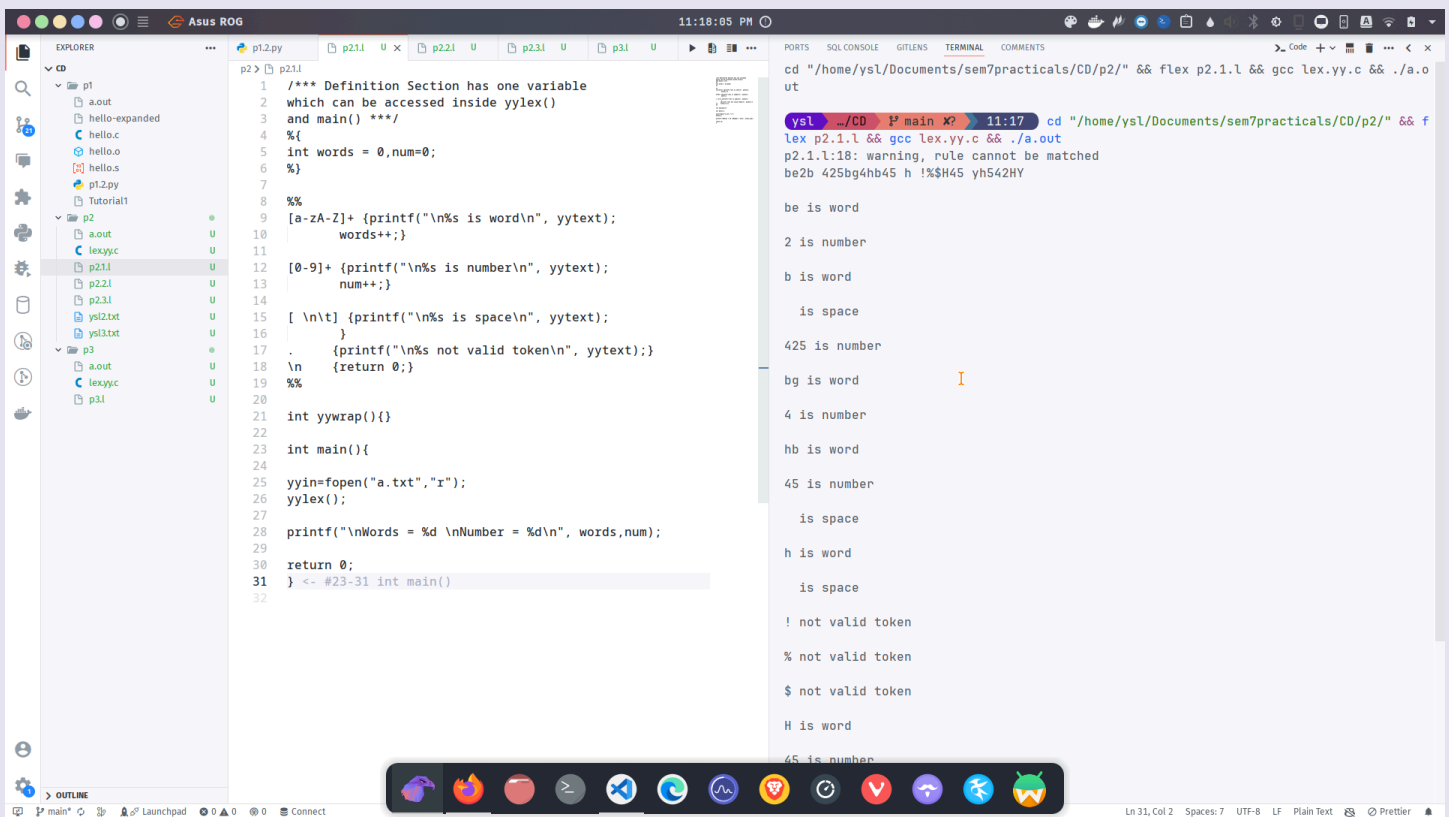
Code :

```
%{  
int words = 0,num=0;  
%}  
  
%%  
[a-zA-Z]+ {printf("\n%s is word\n", yytext);  
           words++;}  
  
[0-9]+ {printf("\n%s is number\n", yytext);  
        num++;}  
  
[ \n\t] {printf("\n%s is space\n", yytext);  
        }  
.  
{printf("\n%s not valid token\n", yytext);};  
\n {return 0;}  
%%  
  
int yywrap(){}  
  
int main(){  
  
yyin=fopen("a.txt","r");  
yylex();  
  
printf("\nWords = %d \nNumber = %d\n", words,num);
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CD Practical 2

```
return 0;  
}
```

Output :



The screenshot shows a VS Code editor with a C program in the main editor and its output in the terminal. The program is a simple lexer that processes input tokens and categorizes them as words, numbers, spaces, or invalid tokens. The terminal output shows the results of running the program on a sample input.

```
cd "/home/ysl/Documents/sem7practicals/CD/p2/" && flex p2.1.l && gcc lex.yy.c && ./a.out  
ysl ~/CD P main X? 11:17 cd "/home/ysl/Documents/sem7practicals/CD/p2/" && f  
lex p2.1.l && gcc lex.yy.c && ./a.out  
p2.1.l:18: warning, rule cannot be matched  
be2b 425bg4hb45 h !%$H4$ yh542HY  
  
be is word  
2 is number  
b is word  
is space  
425 is number  
bg is word  
4 is number  
hb is word  
45 is number  
is space  
h is word  
is space  
! not valid token  
% not valid token  
$ not valid token  
H is word  
45 is number
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CD Practical 2

2) Write a lex program to Scan and Count the number of characters, words, digits, vowels, consonants, special characters and lines in a file.

Code :

```
%{
int word = 0, digit = 0, line = 0, vowel = 0, cons = 0, chars =
0, special = 0;
}%
%%
\n { line++; }
[0-9]+ {
    digit += yyleng;
    chars += yyleng;
}
[a-zA-Z]+ {
    word++;
    for (int i = 0; i < yyleng; i++) {
        chars++;
        char ch = yytext[i];
        if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o'
|| ch == 'u' ||
            ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O'
|| ch == 'U') {
            vowel++;
        } else {
            cons++;
        }
    }
}
```

Name - Yash Lakhtariya

Enrollment number - 21162101012

Branch - CBA Batch - 71

CD Practical 2

```
}
[!@#$%^&*.,;?'""(){}[\]] { special++; chars++; }
. { chars++; }
%%

int yywrap() {}

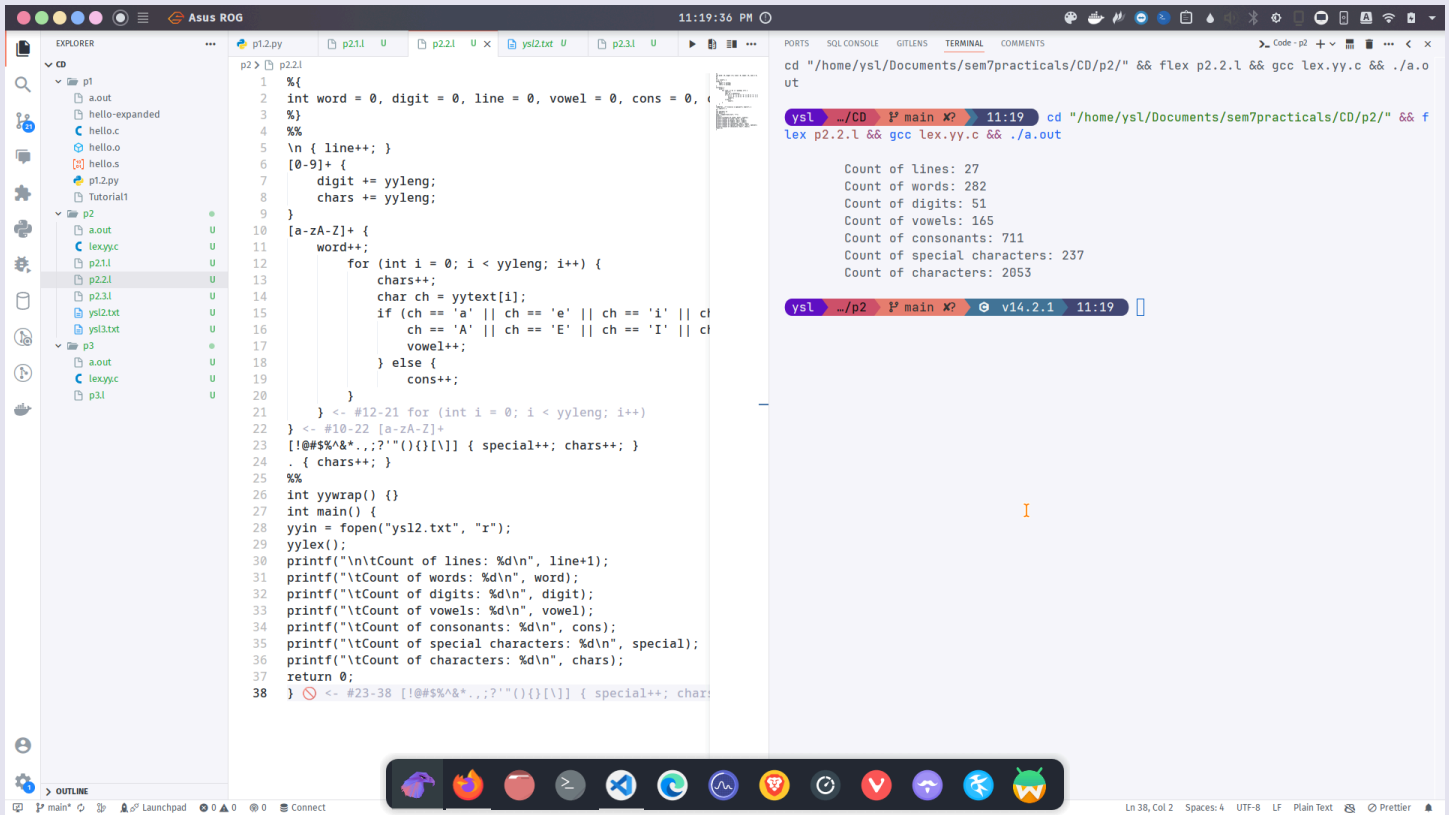
int main() {
yyin = fopen("ysl2.txt", "r");
yylex();

printf("\n\tCount of lines: %d\n", line+1);
printf("\tCount of words: %d\n", word);
printf("\tCount of digits: %d\n", digit);
printf("\tCount of vowels: %d\n", vowel);
printf("\tCount of consonants: %d\n", cons);
printf("\tCount of special characters: %d\n", special);
printf("\tCount of characters: %d\n", chars);
return 0;

}
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CD Practical 2

Output :



The screenshot shows a VS Code editor with a C program in the main editor and its output in the terminal. The program is a lexical analyzer that counts various statistics from an input file.

```
1  %{
2  int word = 0, digit = 0, line = 0, vowel = 0, cons = 0,
3  %}
4  %%
5  \n { line++; }
6  [0-9]+ {
7      digit += yyleng;
8      chars += yyleng;
9  }
10 [a-zA-Z]+ {
11     word++;
12     for (int i = 0; i < yyleng; i++) {
13         chars++;
14         char ch = yytext[i];
15         if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' ||
16             ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U') {
17             vowel++;
18         } else {
19             cons++;
20         }
21     }
22 } <- #12-21 for (int i = 0; i < yyleng; i++)
23 } <- #10-22 [a-zA-Z]+
24 [!@#%&*.,;?'{}(){}[\]] { special++; chars++; }
25 . { chars++; }
26 %%
27 int yywrap() {
28     return 0;
29 }
30 int main() {
31     yyin = fopen("ysl2.txt", "r");
32     yylex();
33     printf("\n\tCount of lines: %d\n", line);
34     printf("\n\tCount of words: %d\n", word);
35     printf("\n\tCount of digits: %d\n", digit);
36     printf("\n\tCount of vowels: %d\n", vowel);
37     printf("\n\tCount of consonants: %d\n", cons);
38     printf("\n\tCount of special characters: %d\n", special);
39     printf("\n\tCount of characters: %d\n", chars);
40     return 0;
41 }
```

The terminal output shows the results of the program's execution:

```
Count of lines: 27
Count of words: 282
Count of digits: 51
Count of vowels: 165
Count of consonants: 711
Count of special characters: 237
Count of characters: 2053
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CD Practical 2

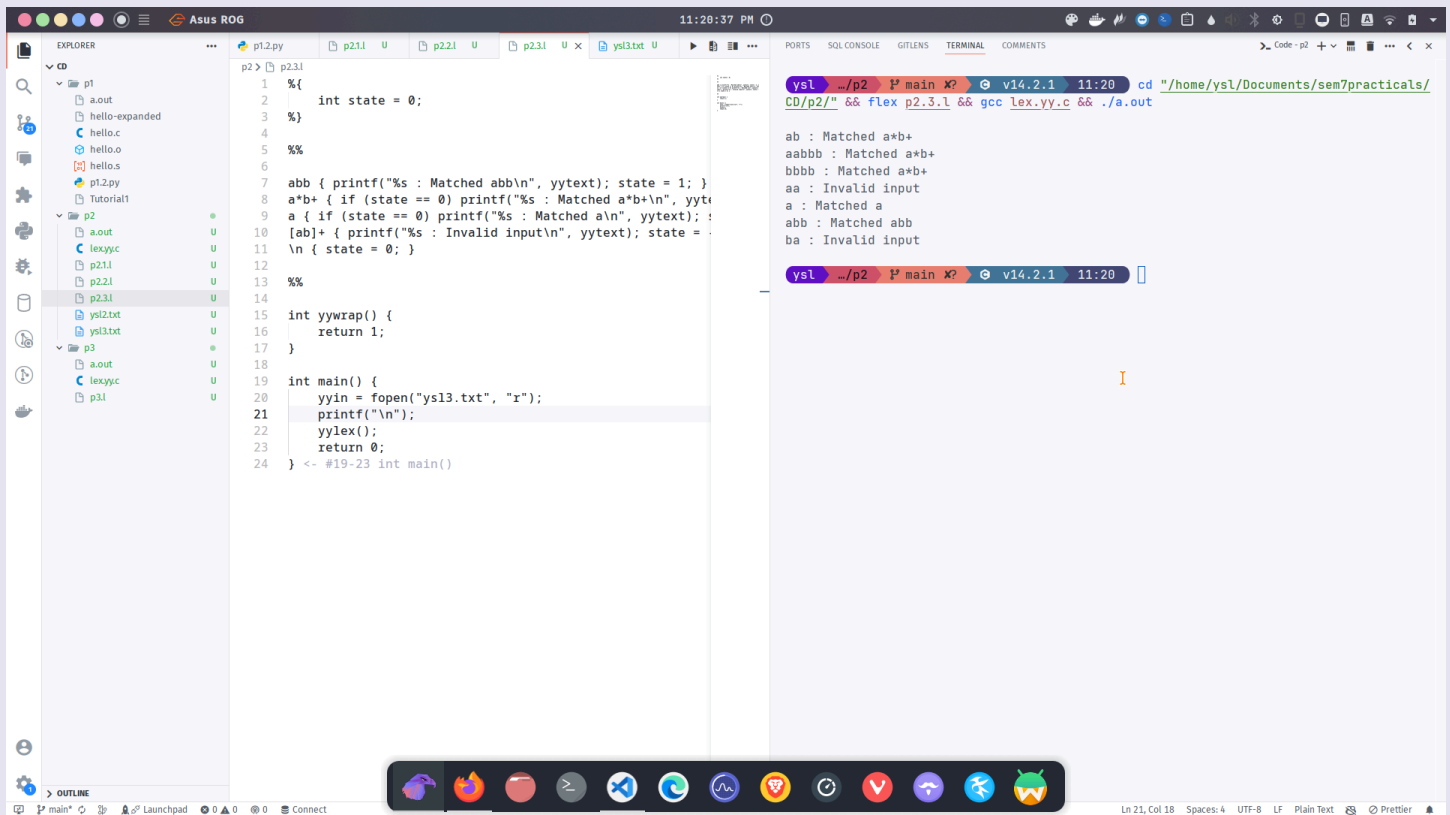
3) Write a lex program to recognize regular expression under 'a', 'a*b+', 'abb', 'b*' over the input set {a,b}.

Code :

```
%{  
    int state = 0;  
}%  
  
%%  
abb { printf("%s : Matched abb\n", yytext); state = 1; }  
a*b+ { if (state == 0) printf("%s : Matched a*b+\n", yytext);  
state = 2; }  
a { if (state == 0) printf("%s : Matched a\n", yytext); state =  
3; }  
[ab]+ { printf("%s : Invalid input\n", yytext); state = -1; }  
\n { state = 0; }  
%%  
  
int yywrap() {  
    return 1;  
}  
  
int main() {  
    yyin = fopen("ysl3.txt", "r");  
    printf("\n");  
    yylex();  
    return 0;  
}
```

Name - Yash Lakhtariya
Enrollment number - 21162101012
Branch - CBA Batch - 71
CD Practical 2

Output :



```
1  %{
2      int state = 0;
3  %}
4
5  %%
6
7  abb { printf("%s : Matched abb\n", yytext); state = 1; }
8  a*b+ { if (state == 0) printf("%s : Matched a*b+\n", yytext);
9  a { if (state == 0) printf("%s : Matched a\n", yytext);
10 [ab]+ { printf("%s : Invalid input\n", yytext); state = 0; }
11 \n { state = 0; }
12
13 %%
14
15 int yywrap() {
16     return 1;
17 }
18
19 int main() {
20     yyin = fopen("ysl3.txt", "r");
21     printf("\n");
22     yylex();
23     return 0;
24 } <- #19-23 int main()
```

```
ysl ~/p2 $ ./main x? v14.2.1 11:20 cd "/home/ysl/Documents/sem7practicals/CD/p2/" && flex p2.3.1 && gcc lex.yy.c && ./a.out
ab : Matched a*b+
aabb : Matched a*b+
bbbb : Matched a*b+
aa : Invalid input
a : Matched a
abb : Matched abb
ba : Invalid input
ysl ~/p2 $ ./main x? v14.2.1 11:20
```