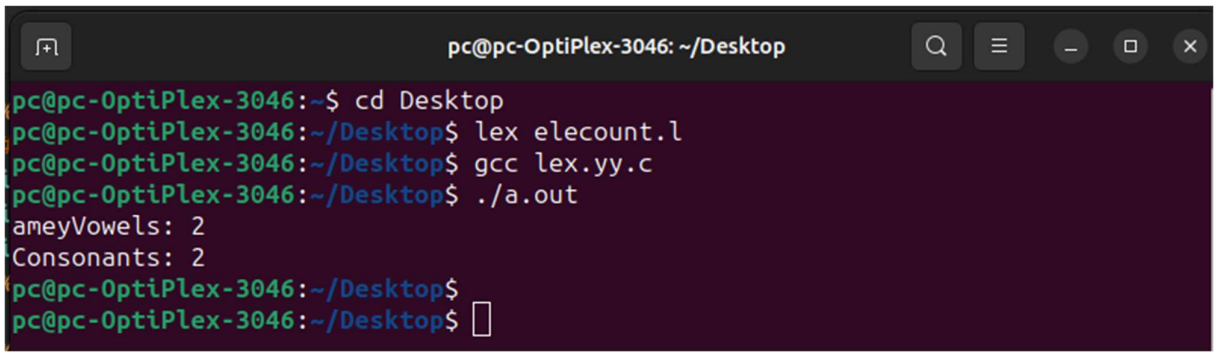
Practical 2

Aim : Write a Lex Specification for each of the following
1. counting number of vowels and consonant in given string.

Code:

```
%{
#undef yywrap
int yywrap(void);
int vowel_count = 0;
int consonant_count = 0;
%}

%%
[aeiouAEIOU]   { vowel_count++; }
[bcdfghjklmnpqrstvwxyzBCDFGHJKLMNPQRSTVWXYZ]  { consonant_count++; }
scanf           { printf("readf"); }
printf          { printf("writef"); }

%%

int main(){
yylex();
printf("Vowels: %d\nConsonants: %d\n", vowel_count, consonant_count);
return 0;

}
int yywrap(void){
return 1;

}
```

Output :



2. For counting number of comments in given c program
Code :

```
%{
#undef yywrap
int yywrap(void);
int comment_lines = 0;
%}

%%

"/*"(.|\n)*?"*/" { comment_lines++; }
"//"(.*)         { comment_lines++; }
.      ;

%%

int main() {
    yylex();
    printf("Comment Lines: %d\n", comment_lines);
    return 0;
}

int yywrap(void){
return 1;

}
```

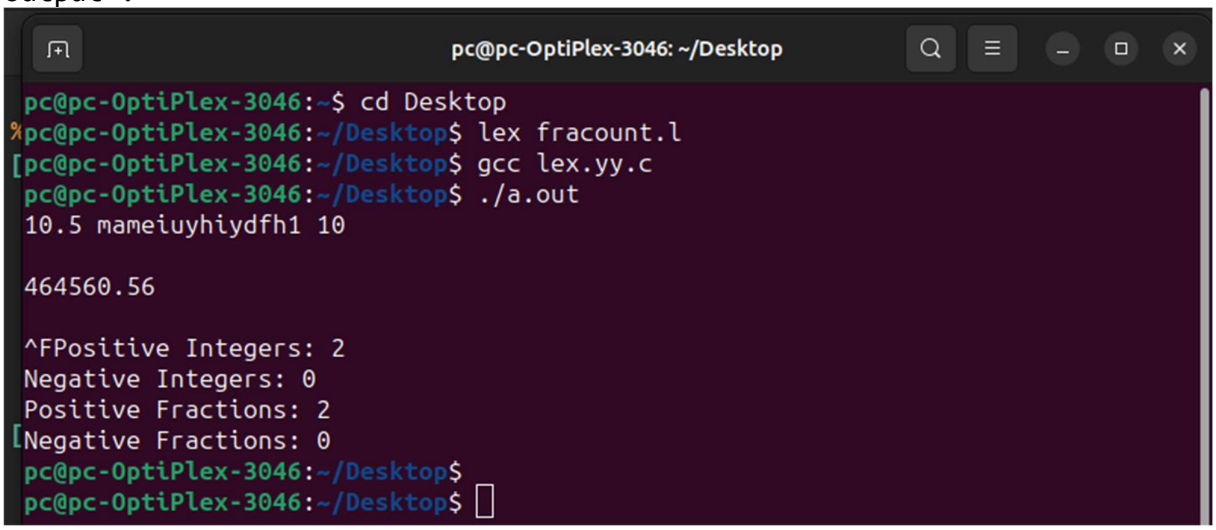
Output:

```
pc@pc-OptiPlex-3046:~/Desktop$ lex comment.l
pc@pc-OptiPlex-3046:~/Desktop$ gcc lex.yy.c
pc@pc-OptiPlex-3046:~/Desktop$ ./a.out
#inclued<stdio.h>

//amey pand

/*ameuyyhiyfh*/

Comment Lines: 2
pc@pc-OptiPlex-3046:~/Desktop$
pc@pc-OptiPlex-3046:~/Desktop$
```

3. For counting number of
i. Positive and negative integers.
ii. Positive and negative fractions.
Code :

```
%{
#undef yywrap
int yywrap(void);
int positive_integers = 0;
int negative_integers = 0;
int positive_fractions = 0;
int negative_fractions = 0;
%}
%%
[+-]?[0-9]+     {
                    int num = atoi(yytext);
                    if (num > 0) {
                        positive_integers++;
                    } else if (num < 0) {
                        negative_integers++;
                    }
                }
[+-]?[0-9]+[.][0-9]+ {
                        double num = atof(yytext);
                        if (num > 0) {
                            positive_fractions++;
                        } else if (num < 0) {
                            negative_fractions++;
                        }
                    }
.               ;

%%
int main() {
    yylex();
    printf("Positive Integers: %d\nNegative Integers: %d\n",
positive_integers, negative_integers);
    printf("Positive Fractions: %d\nNegative Fractions: %d\n",
positive_fractions, negative_fractions);
    return 0;
}

int yywrap(void){
return 1;

}
```

Output :

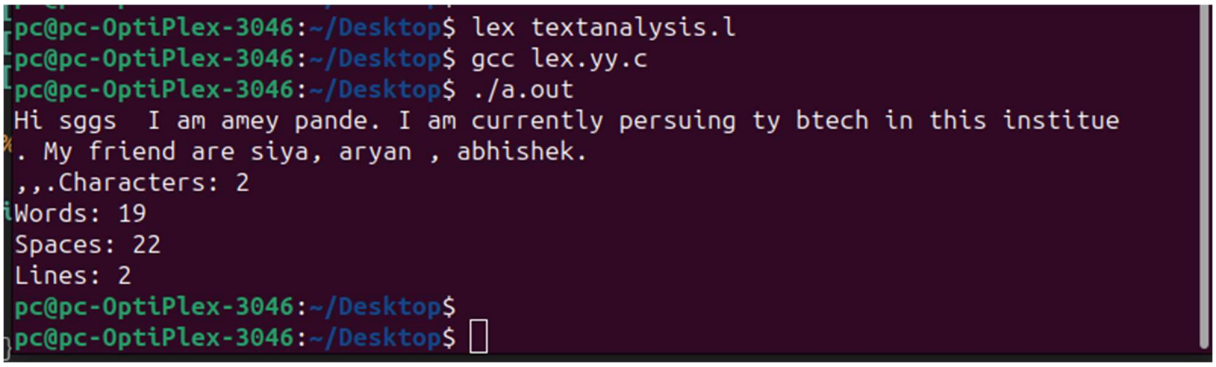


4. for counting Number of characters, words, spaces and line in a given input file

Code :

```
%{
#undef yywrap
int yywrap(void);
int char_count = 0;
int word_count = 0;
int space_count = 0;
int line_count = 0;
%}

%%
\n              { line_count++; }
[ \t]           { space_count++; }
[a-zA-Z]        { char_count++; }
[a-zA-Z]+       { word_count++; }

%%

int main() {
    yylex();
    printf("Characters: %d\nWords: %d\nSpaces: %d\nLines: %d\n",
char_count, word_count, space_count, line_count);
    return 0;
}

int yywrap(void){
return 1;

}
```

Output :



5. Number of ‘scanf’ and ‘printf’ statements in a C program. Replace them with ‘readf’ and ‘writef’ statements respectively.

Code :

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

%%
scanf[^;]*;   { printf("readf"); }
printf[^;]*;  { printf("writef"); }
.             { printf("%s", yytext); }
%%

int main() {
    yylex();
    return 0;
}
/*************************************************************************
**/
%{
#undef yywrap
int yywrap(void);
%}

%%
scanf           { printf("readf"); }
printf          { printf("writef"); }
.               ;

%%

int main() {
    yylex();
    return 0;
}
int yywrap(void){
return 1;

}
```

Output :

```
pc@pc-OptiPlex-3046:~/Desktop$ lex replace2.l
pc@pc-OptiPlex-3046:~/Desktop$ gcc lex.yy.c -o replace2 -ll
pc@pc-OptiPlex-3046:~/Desktop$ ./a.out < input.c




writef
readf

writef




pc@pc-OptiPlex-3046:~/Desktop$
```

```
pc@pc-OptiPlex-3046:~/Desktop$ lex replace.l
pc@pc-OptiPlex-3046:~/Desktop$ gcc lex.yy.c
pc@pc-OptiPlex-3046:~/Desktop$ ./a.out
printf("hello");
writef
scanf("%d", &a);
readf
pc@pc-OptiPlex-3046:~/Desktop$
```