PRACTICAL 3
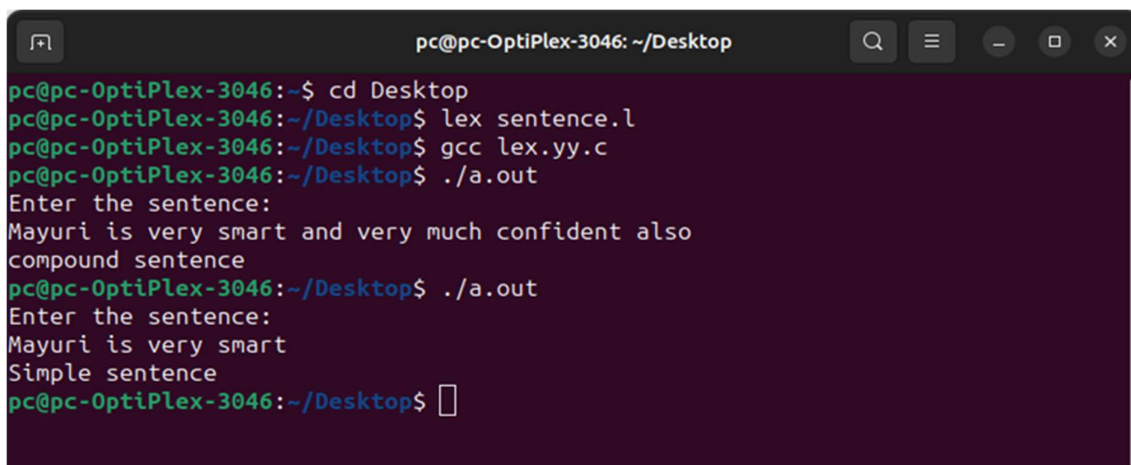
Aim: 1.Write Lex Specification to recognize whether given string is simple or compound.

Code:

```
/*Program to recognize whether a given sentence is simple or compound.*/
%{
    #include<stdio.h>
    int flag=0;
%}
%%
and |
or |
but |
because |
if |
then |
nevertheless  { flag=1; }
.  ;
\n  { return 0; }
%%
int main()
{
    printf("Enter the sentence:\n");
    yylex();
    if(flag==0)
        printf("Simple sentence\n");
    else
        printf("compound sentence\n");
}
int yywrap( )
{
    return 1;
}
```

Output:

2. Write Lex Specification to recognize and count number of identifiers in a given input file

Code:

```
%{
#include <stdio.h>
int count = 0;
int yywrap() {
    return 1; // Indicate that there are no more files to process
}
%}

op [+-*/]
letter [a-zA-Z]
digitt [0-9]
id {letter}*|({letter}{digitt})+
notid {digitt}{letter}+

%%
[\t\n]+

("int")|("float")|("char")|("case")|("default")|("if")|("for")|("printf")
|("scanf") {printf("%s is a keyword\n", yytext);}

{id} {printf("%s is an identifier\n", yytext); count++;}

{notid} {printf("%s is not an identifier\n", yytext);}

%%

int main()
{
    FILE *fp;
    char file[100];
    printf("\nEnter the filename: ");
    scanf("%s", file);
    fp = fopen(file, "r");
    if (fp == NULL) {
        perror("Error opening file");
        return 1;
    }
    yyin = fp;
    yylex();
    printf("Total identifiers are: %d\n", count);
    fclose(fp);
    return 0;
}
```

Input File :
#file name tokens.txt
 int
float
78f
90gh
a
d
are case
default
printf
scanf

Output: