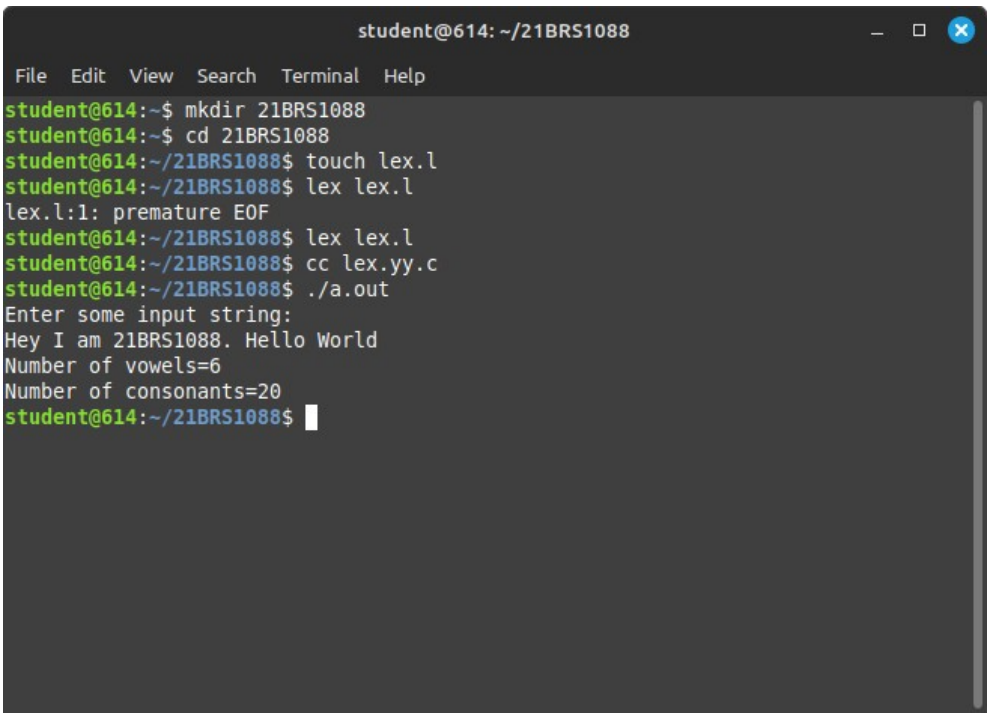
Lab – 4

Name – Akshit Jain
Reg. No – 21BRS1088

1) Write a lex program to input a sentence and count the number of vowels and consonents

Code-

```
%{
        #include<stdio.h>
        int vow=0, con=0;
%}

%%
[ \t\n]+    ;
[aeiouAEIOU]+    {vow++;}
[^aeiouAEIOU]     {con++;}
%%

int main( )
{
        printf("Enter some input string:\n");
        yylex();
        printf("Number of vowels=%d\n",vow);
        printf("Number of consonants=%d\n",con);
}

int yywrap( )
{
        return 1;
}
```



2) Write a lex program to input a sentence and count

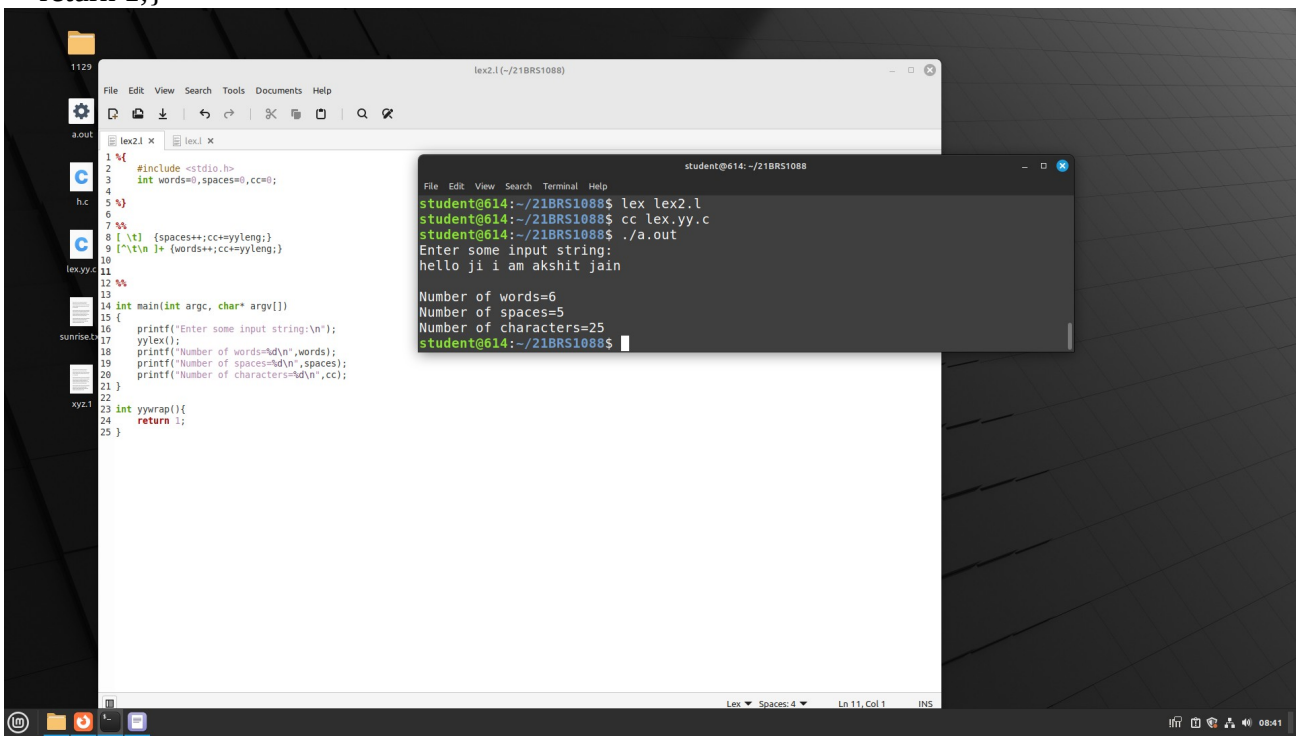
a) No of characters

b) No of words

c) No of spaces

Code

```
%{
   #include <stdio.h>
   int words=0,spaces=0,cc=0;

%}

%%
[ \t]  {spaces++;cc+=yyleng;}
[^\t\n ]+ {words++;cc+=yyleng;}


%%

int main(int argc, char* argv[])
{
        printf("Enter some input string:\n");
        yylex();
        printf("Number of words=%d\n",words);
   printf("Number of spaces=%d\n",spaces);
   printf("Number of characters=%d\n",cc);
}

int yywrap(){
   return 1;}
```



LAB ASSIGNMENT

Name – Akshit Jain
Reg. No – 21BRS1088

Question – Create lexical analyzer using lex language

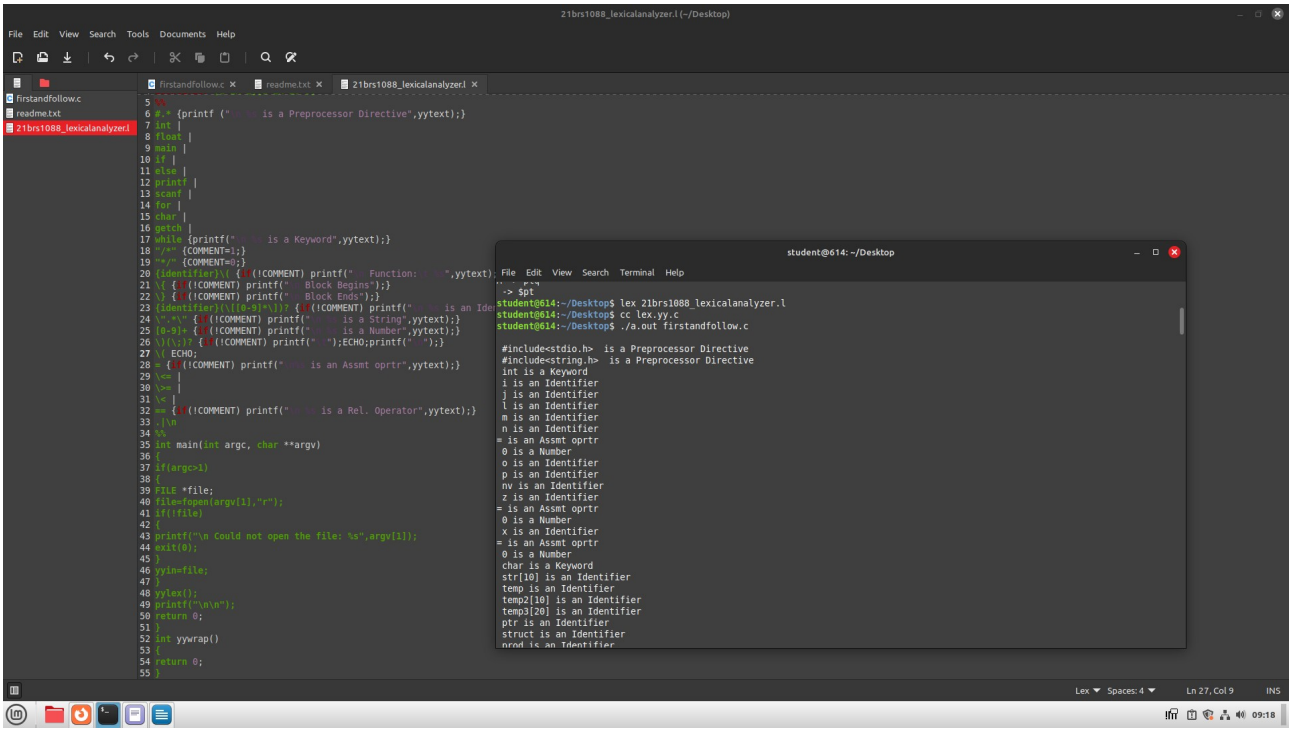
> Reading the first and follow c code to find different tokens

Code

```
%{
int COMMENT=0;
%}
identifier [a-zA-Z][a-zA-Z0-9]*
%%
#.* {printf ("\n %s is a Preprocessor Directive",yytext);}
int |
float |
main |
if |
else |
printf |
scanf |
for |
char |
getch |
while {printf("\n %s is a Keyword",yytext);}
"/*" {COMMENT=1;}
"*/" {COMMENT=0;}
{identifier}\( {if(!COMMENT) printf("\n Function:\t %s",yytext);}
\{ {if(!COMMENT) printf("\n Block Begins");}
\} {if(!COMMENT) printf("\n Block Ends");}
{identifier}(\[[0-9]*\])? {if(!COMMENT) printf("\n %s is an Identifier",yytext);}
\".*\" {if(!COMMENT) printf("\n %s is a String",yytext);}
[0-9]+ {if(!COMMENT) printf("\n %s is a Number",yytext);}
\)(\;)? {if(!COMMENT) printf("\t");ECHO;printf("\n");}
\( ECHO;
= {if(!COMMENT) printf("\n%s is an Assmt oprtr",yytext);}
\<= |
\>= |
\< |
== {if(!COMMENT) printf("\n %s is a Rel. Operator",yytext);}
.|\n
%%
int main(int argc, char **argv)
{
if(argc>1)
{
FILE *file;
file=fopen(argv[1],"r");
```

```
if(!file)
{
printf("\n Could not open the file: %s",argv[1]);
exit(0);
}
yyin=file;
}
yylex();
printf("\n\n");
return 0;
}
int yywrap()
{
return 0;
}
```

OUTPUT



student@614: ~/Desktop

File Edit View Search Terminal Help

```
 -> $pt
student@614:~/Desktop$ lex 21brs1088_lexicalanalyzer.l
student@614:~/Desktop$ cc lex.yy.c
student@614:~/Desktop$ ./a.out firstandfollow.c

 #include<stdio.h>  is a Preprocessor Directive
 #include<string.h>  is a Preprocessor Directive
 int is a Keyword
 i is an Identifier
 j is an Identifier
 l is an Identifier
 m is an Identifier
 n is an Identifier
= is an Assmt oprtr
 0 is a Number
 o is an Identifier
 p is an Identifier
 nv is an Identifier
 z is an Identifier
= is an Assmt oprtr
 0 is a Number
 x is an Identifier
= is an Assmt oprtr
 0 is a Number
 char is a Keyword
 str[10] is an Identifier
 temp is an Identifier
 temp2[10] is an Identifier
 temp3[20] is an Identifier
 ptr is an Identifier
 struct is an Identifier
 prod is an Identifier
```

LAB ASSIGNMENT

Name – Akshit Jain
Reg. No – 21BRS1088

Question – Calculate first and folllow for the given grammar using C program

> Created a code using file handling method where I am creating a file named “readme.txt”
> Reading the input from the readme file and processing furthur

Code

```
#include<stdio.h>
#include<string.h>

int i,j,l,m,n=0,o,p,nv,z=0,x=0;
char str[10],temp,temp2[10],temp3[20],*ptr;

struct prod
{
   char lhs[10],rhs[10][10],ft[10],fol[10];
   int n;
}pro[10];

void findter()
{
   int k,t;
   for(k=0;k<n;k++)
   {
      if(temp==pro[k].lhs[0])
      {
         for(t=0;t<pro[k].n;t++)
         {
            if( pro[k].rhs[t][0]<65 || pro[k].rhs[t][0]>90 )
               pro[i].ft[strlen(pro[i].ft)]=pro[k].rhs[t][0];
            else if( pro[k].rhs[t][0]>=65 && pro[k].rhs[t][0]<=90 )
            {
               temp=pro[k].rhs[t][0];
               if(temp=='S')
                  pro[i].ft[strlen(pro[i].ft)]='#';
               findter();
            }
         }
         break;
      }
   }
}

void findfol()
{
   int k,t,p1,o1,chk;
   char *ptr1;
```

```
    for(k=0;k<n;k++)
    {
        chk=0;
        for(t=0;t<pro[k].n;t++)
        {
            ptr1=strchr(pro[k].rhs[t],temp);
            if( ptr1 )
            {
                p1=ptr1-pro[k].rhs[t];
                if(pro[k].rhs[t][p1+1]>=65 && pro[k].rhs[t][p1+1]<=90)
                {
                    for(o1=0;o1<n;o1++)
                        if(pro[o1].lhs[0]==pro[k].rhs[t][p1+1])
                        {
                            strcat(pro[i].fol,pro[o1].ft);
                            chk++;
                        }
                }
                else if(pro[k].rhs[t][p1+1]=='\0')
                {
                    temp=pro[k].lhs[0];
                    if(pro[l].rhs[j][p]==temp)
                        continue;
                    if(temp=='S')
                        strcat(pro[i].fol,"$");
                    findfol();
                    chk++;
                }
                else
                {
                    pro[i].fol[strlen(pro[i].fol)]=pro[k].rhs[t][p1+1];
                    chk++;
                }
            }
        }
        if(chk>0)
            break;
    }
}

int main()
{
    FILE *f;
    //clrscr();

    for(i=0;i<10;i++)
        pro[i].n=0;

    f=fopen("readme.txt","r");
    while(!feof(f))
    {
        fscanf(f,"%s",pro[n].lhs);
```

```
    if(n>0)
    {
        if( strcmp(pro[n].lhs,pro[n-1].lhs) == 0 )
        {
            pro[n].lhs[0]='\0';
            fscanf(f,"%s",pro[n-1].rhs[pro[n-1].n]);
            pro[n-1].n++;
            continue;
        }
    }
    fscanf(f,"%s",pro[n].rhs[pro[n].n]);
    pro[n].n++;
    n++;
}

printf("\n\nGiven Grammar \n\n");
for(i=0;i<n;i++)
    for(j=0;j<pro[i].n;j++)
        printf("%s -> %s\n",pro[i].lhs,pro[i].rhs[j]);

pro[0].ft[0]='#';
for(i=0;i<n;i++)
{
    for(j=0;j<pro[i].n;j++)
    {
        if( pro[i].rhs[j][0]<65 || pro[i].rhs[j][0]>90 )
        {
            pro[i].ft[strlen(pro[i].ft)]=pro[i].rhs[j][0];
        }
        else if( pro[i].rhs[j][0]>=65 && pro[i].rhs[j][0]<=90 )
        {
            temp=pro[i].rhs[j][0];
            if(temp=='S')
                pro[i].ft[strlen(pro[i].ft)]='#';
            findter();
        }
    }
}

printf("\n--FIRST--\n");
for(i=0;i<n;i++)
{
    printf("\n%s -> ",pro[i].lhs);
    for(j=0;j<strlen(pro[i].ft);j++)
    {
        for(l=j-1;l>=0;l--)
            if(pro[i].ft[l]==pro[i].ft[j])
                break;
        if(l==-1)
            printf("%c",pro[i].ft[j]);
    }
}
```

```
    for(i=0;i<n;i++)
        temp2[i]=pro[i].lhs[0];
    pro[0].fol[0]='$';
    for(i=0;i<n;i++)
    {
        for(l=0;l<n;l++)
        {
            for(j=0;j<pro[i].n;j++)
            {
                ptr=strchr(pro[l].rhs[j],temp2[i]);
                if( ptr )
                {
                    p=ptr-pro[l].rhs[j];
                    if(pro[l].rhs[j][p+1]>=65 && pro[l].rhs[j][p+1]<=90)
                    {
                        for(o=0;o<n;o++)
                            if(pro[o].lhs[0]==pro[l].rhs[j][p+1])
                                    strcat(pro[i].fol,pro[o].ft);
                    }
                    else if(pro[l].rhs[j][p+1]=='\0')
                    {
                        temp=pro[l].lhs[0];
                        if(pro[l].rhs[j][p]==temp)
                            continue;
                        if(temp=='S')
                            strcat(pro[i].fol,"$");
                        findfol();
                    }
                    else
                        pro[i].fol[strlen(pro[i].fol)]=pro[l].rhs[j][p+1];
                }
            }
        }
    }

    printf("\n--FOLLOW--\n");
    for(i=0;i<n;i++)
    {
        printf("\n%s -> ",pro[i].lhs);
        for(j=0;j<strlen(pro[i].fol);j++)
        {
            for(l=j-1;l>=0;l--)
                if(pro[i].fol[l]==pro[i].fol[j])
                    break;
            if(l==-1)
                printf("%c",pro[i].fol[j]);
        }
    }
    printf("\n");
    //getch();
}
```

OUTPUT

```
student@614: ~/Desktop
File  Edit  View  Search  Terminal  Help
student@614:~/Desktop$ gcc firstandfollow.c
student@614:~/Desktop$ ./a.out


Given Grammar

S -> ABE
S -> a
A -> p
A -> t
B -> Aq
S -> f
A -> w
 ->

--FIRST--

S -> #pta
A -> pt
B -> pt
S -> f
A -> w
 ->
--FOLLOW--

S -> $
A -> ptq
B ->
S ->
A -> ptq
 -> $pt
student@614:~/Desktop$ //21BRS1088
```