Program:

//Karedla 160114733091
/* Program for Tokenization (counting the no of characters, lines, spaces, words, integer, float, Sum of the given integer & float etc..,).*/

```
#include<stdio.h>
#include<stdlib.h>
#include<stdbool.h>
#include<math.h>
int l,w,i,f;
bool testInt(char *,int *);
bool testFloat(char *,float *);
int main()
{
  FILE *fp;
  //int l,w,i,f;
  l=w=i=f=0;
  char c;
  int sumi=0;
  float sumf=0.0;
  fp=fopen("test.txt","r");char buf[100];int p=0;
  if(fp==NULL)
  exit(1);
  else
  {
    while((c=fgetc(fp))!=EOF)
    {
      if(c!=' '&&c!='\n'&&c!='\r')
      {
        buf[p++]=c;
      }
      else // if(c==' '||c=='\n'||c=='\r')
      {
        buf[p]='\0';
       // printf("hi :%s\n",buf);
        if(c=='\n')//||c=='\r')
        l++;
        int getint;float getfloat;
        if(testInt(buf,&getint))
        {
      //     i++;
          sumi+=getint;
```

```
            }
            else if(testFloat(buf,&getfloat))
            {
        //      f++;
                sumf+=getfloat;
            }
                    else
                w++;
            buf[p]='\0';
            p=0;
        }
        // printf("%d %d %d %d %s",i,l,f,w,buf);

    }
  }
printf("sum if ints %d\n",sumi);
printf("sum of floats %f\n",sumf);
printf("no of words %d\n",w);
printf("no of lines %d\n",l);
printf("no of integers %d\n",i);
printf("no of floats %d\n",f);
}
bool testInt(char *s,int *getint)
{
  //    printf("hi %d\n",*getint);
        *getint=0;
   int hit=1;
   int p;
   if(s[0]=='-')
      hit=-1;
   if(s[0]=='+'||s[0]=='-'||(s[0]>=48&&s[0]<=57))
   {
      if(s[0]>=48&&s[0]<=57)
      *getint=(int)(s[0]-48);
   }
   else
      return false;
   for(p=1;s[p]!='\0';p++)
   {
      if(s[p]>=48&&s[p]<=57)
      {
         *getint=(*getint)*10+(int)(s[p]-48);
      }
      else
         return false;
   }
   *getint=(*getint)*hit;
        i++;
   return true;
}
bool testFloat(char *s,float *getfloat)
{
        //printf("hi 3 : %s",s);
   float hit=1.0;
   *getfloat=0.0;
   int i;int count=0;
   if(s[0]=='-')
      hit=-1.0;
   if(s[0]=='+'||s[0]=='-'||(s[0]>=48&&s[0]<=57))
   {
      if(s[0]>=48&&s[0]<=57)
      *getfloat=(*getfloat)*10.0+(float)(s[0]-48);
   }
   else
      return false;
        int flag=0;int power1=-1;
        //printf("hi 4:%f",*getfloat);
   for(i=1;s[i]!='\0';i++)
   {
      if(s[i]=='.'&&count==0)
      {
         count++;flag=1;
              continue;
      }
      else if(s[i]=='.'&&count>0)
         return false;
      if((s[i]>=48&&s[i]<=57)&&flag==0)
      {
         *getfloat=(*getfloat)*10+(s[i]-48);
      }
      else if((s[i]>=48&&s[i]<=57)&&flag==1)
      {
            *getfloat=(*getfloat)+(s[i]-48)*(pow(10,power1));
            power1--;
      }
      else
         return false;

   }
```

```
    *getfloat=(*getfloat)*hit;
        f++;
    return true;
}
```

Testing:

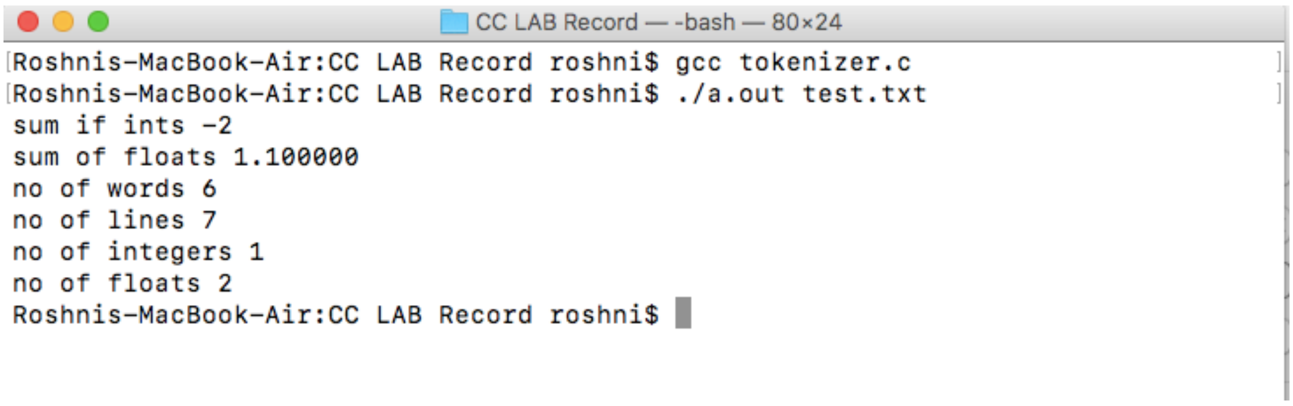
Input:

```
-2 2.2.2.2
+2a3
-a
3.2a
2.2 -1.1
```

Expected Output:

```
sum if ints -2
sum of floats 1.100000
no of words 6
no of lines 7
no of integers 1
no of floats 2
```

Actual Output:-



```
Roshnis-MacBook-Air:CC LAB Record roshni$ gcc tokenizer.c
Roshnis-MacBook-Air:CC LAB Record roshni$ ./a.out test.txt
sum if ints -2
sum of floats 1.100000
no of words 6
no of lines 7
no of integers 1
no of floats 2
Roshnis-MacBook-Air:CC LAB Record roshni$ 
```

Result:-

Successfully executed the program

Program:

```
///Karedla160114733091
// Program to implement Scanner using C.
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
char key[32]
[10]={"auto","break","case","char","const","continue","default","do","double","else",
"enum","extern","float","for","goto","if","int","long","register","return","short","sign
ed","sizeof","static","struct","switch","typedef","union","unsigned","void","volatile",
"while"};
char echar[10]={'b','t','v','r','f','n','\\','?','0','a'},com[100];
int main()
{
 FILE *f;
 char c,x,y,z,temp[20],fr[5];
 int i,j,k,l=1,t=1,f1,flag,m,p;
 f=fopen("file.txt","r");
 if(f)
 {
  printf("Line\tToken no\tToken name\t\tLexeme\n");
  while((c=getc(f))!=EOF)
  {
   j=0;
   if(isalpha(c))
   {
       flag=0;
       temp[j++]=c;c=getc(f);
       while(isalnum(c))
       {
        temp[j++]=c;c=getc(f);
       }
       for(i=0;i<32;i++)
       {
        if(strcmp(temp,key[i])==0)
        {
         flag=1;
         break;
        }
       }
       if(flag==1)
       {
        printf("%d\t%d\tkeyword\t\t\t%s\n",l,t,temp);
        t++;
       }
       else
       {
        if(c=='.')
         {
         temp[j++]=c;
         for(p=0;p<5;p++)
         { fr[p]='\0'; }
         y=getc(f);
         while(y!='>')
         {
          fr[p++]=y;
          temp[j++]=y;
          y=getc(f);
          if(y==';'||y=='"')  break;
         }
         fseek(f,-(j+10),SEEK_CUR);
         y=getc(f);
         fseek(f,(j+9),SEEK_CUR);
         if(y=='#')
         {
          printf("%d\t%d\theader\t\t\t%s\n",l,t,temp);
         }
         else
         { printf("%d\t%d\tidentifier\t\t%s\n",l,t,temp); t++;
         fseek(f,-1,SEEK_CUR);}
        }
        else
        {
         fseek(f,-1,SEEK_CUR);
         printf("%d\t%d\tidentifier\t\t%s\n",l,t,temp);
          t++;
        }
       }
}
else if(c=='+'||c=='='||c=='-'||c=='<'||c=='>'||c=='*'||c=='/')
{
       if(c=='+')
       {
        x=getc(f);
        if(x=='+')
         {
         temp[j++]=c;
         temp[j++]=x;
         printf("%d\t%d\tincrementer\t\t%s\n",l,t,temp);
         t++;
 }
}
else if(c=='-')
{
       x=getc(f);
       if(x=='-')
       {
         temp[j++]=c;
         temp[j++]=x;
         printf("%d\t%d\tdecrementer\t\t%s\n",l,t,temp);
         t++;
 }
}
else if(c=='/')
{
       x=getc(f);
       if(x=='*')
       {
         temp[j++]=c;
         temp[j++]=x;
         printf("%d\t%d\tcomment starts\t\t%s\n",l,t,temp);
         t++;
         x=getc(f);
         k=0;
         while(x!='*')
         {
                if(x=='\n')
                 x=' ';
                com[k++]=x;
                x=getc(f);
         }
         fseek(f,-1,SEEK_CUR);
         printf("%d\t%d\tcomment\t\t\t%s\n",l,t,com);
         t++;
         for(k=0;k<100;k++)
          com[k]='\0';
       }
       if(x=='/')
       {
         temp[j++]=c;
         temp[j++]=x;
         printf("%d\t%d\tcomment starts\t\t%s\n",l,t,temp);
         t++;
         x=getc(f);
         k=0;
         while(x!='\n')
         {
                com[k++]=x;
                x=getc(f);
         }
         fseek(f,-1,SEEK_CUR);
         printf("%d\t%d\tcomment\t\t\t%s\n",l,t,com);
         t++;
         for(k=0;k<100;k++)
          com[k]='\0';
        }
 }
 else if(c=='*')
 {
        x=getc(f);
        if(x=='/')
        {
         temp[j++]=c;
         temp[j++]=x;
         printf("%d\t%d\tcomment ends\t\t%s\n",l,t,temp);
         t++;
        }
 }
 else
 {
        printf("%d\t%d\toperator\t\t%c\n",l,t,c);
        t++;
  }
}
else if(isdigit(c))
{
       temp[j++]=c;
       c=getc(f);
       while(isdigit(c))
       {
        temp[j++]=c;
        c=getc(f);
       }
       fseek(f,-1,SEEK_CUR);
       printf("%d\t%d\tdigit\t\t\t%s\n",l,t,temp);
       t++;
}
else if(c=='{'||c=='}'||c==';'||c==','||c=='('||c==')'||c=='?'||c=='!')
```

```
  {
        printf("%d\t%d\tspecial symbol\t\t%c\n",l,t,c);t++;
  }
  else if(c=='#')
  {
        printf("%d\t%d\tpreprocesssor\t\t%c\n",l,t,c);t++;
  }
  else if(c=='\\')
  {
        y=getc(f);
        for(i=0;i<3;i++)
        {
         if(y==echar[i])
         {
          temp[j++]=c;
          temp[j++]=y;
          printf("%d\t%d\t\tesacpe character\t%s\n",l,t,temp);
          t++;
         }
        }
  }
  else if(c==' '){ }
  else if(c=='\n'){l++;}
  else{ }
  for(i=0;i<20;i++)
         temp[i]='\0';
 }
}
else
{
 printf("\nfp file doesnot exist\n");
 return -1;
}
fclose(f);
return 0;
}
```

Testing:

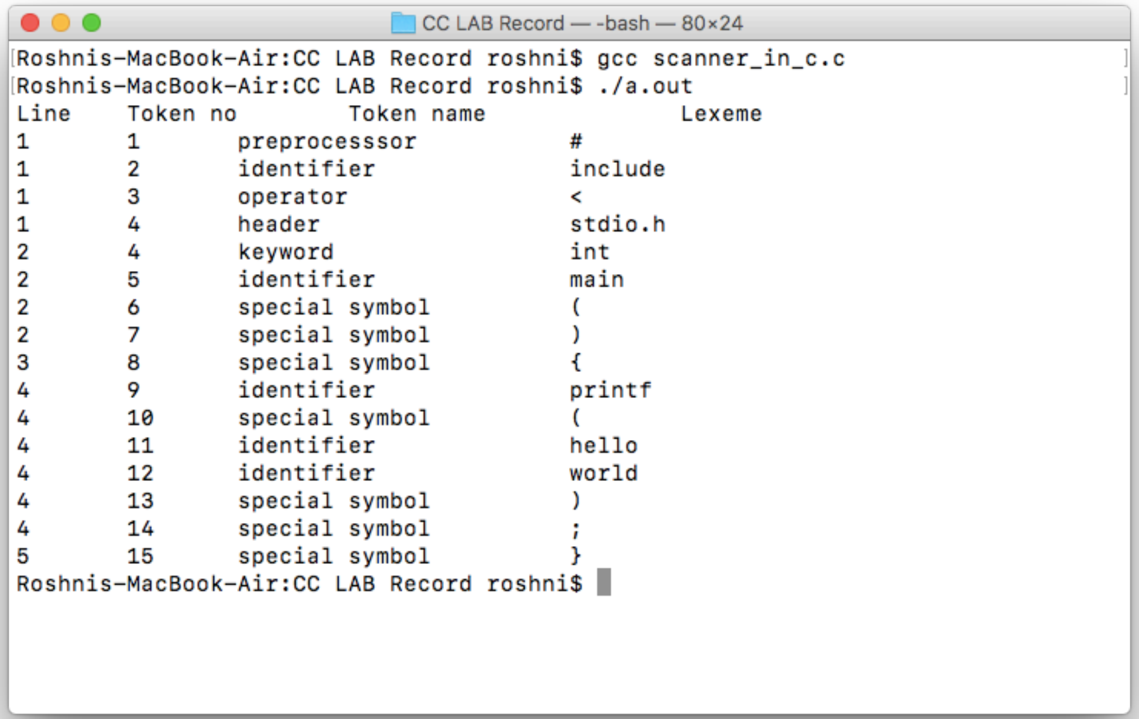
Input:

```
        hello.c
        #include<stdio.h>
int main()
{
printf("hello world");
}
```

Expected Output:

```
Line Token no Token name       Lexeme
1    1        preprocesssor    #
1    2        identifier       include
1    3        operator         <
1    4        header           stdio.h
2    4        keyword          int
2    5        identifier       main
2    6        special symbol   (
2    7        special symbol   )
3    8        special symbol   {
4    9        identifier       printf
4    10       special symbol   (
4    11       identifier       hello
4    12       identifier       world
4    13       special symbol   )
4    14       special symbol   ;
5    15       special symbol   }
```

Actual Output:-



```
Roshnis-MacBook-Air:CC LAB Record roshni$ gcc scanner_in_c.c
Roshnis-MacBook-Air:CC LAB Record roshni$ ./a.out
Line     Token no         Token name              Lexeme
1        1        preprocesssor           #
1        2        identifier              include
1        3        operator                <
1        4        header                  stdio.h
2        4        keyword                 int
2        5        identifier              main
2        6        special symbol          (
2        7        special symbol          )
3        8        special symbol          {
4        9        identifier              printf
4        10       special symbol          (
4        11       identifier              hello
4        12       identifier              world
4        13       special symbol          )
4        14       special symbol          ;
5        15       special symbol          }
Roshnis-MacBook-Air:CC LAB Record roshni$
```

Result:-

Successful executed the program.

Program:

```
/*Karedla 160114733091*/
/* Program to implement Scanner application using LEX.*/
%{
#include<stdio.h>
#include<string.h>
int t=1;
int l=1,i;
%}

hfile assert.h|complex.h|ctype.h|errno.h|fenv.h|float.h|inttypes.h|iso646.h|limits.h|
locale.h|math.h|setjmp.h|signal.h|stdalign.h|stdarg.h|stdatomic.h|stdbool.h|stddef.h|
stdint.h|stdio.h|stdlib.h|stdnoreturn.h|string.h|tgmath.h|threads.h|time.h|uchar.h|
wchar.h|wctype.h
key auto|break|case|char|const|continue|default|do|double|else|enum|extern|float|for|
goto|if|int|long|register|return|short|signed|sizeof|static|struct|switch|typedef|union|
unsigned|void|volatile|while
file txt|h|c
format c|d|e|E|f|g|G|o|s|u|x|X
func fopen|fclose|getchar|putchar|printf|scanf|strcat|strcmp|strcpy|isdigit|isalpha|
isalnum|islower|isupper|acos|asin|atan|cos|exp|fabs|sqrt|time|difftime|clock|malloc|
rand|srand
type int|char|double|float
id [a-zA-Z][a-zA-Z0-9_]*
size [0-9]+
index [a-zA-Z]
s {size}|{index}|{index}"++"|{index}--|"--"{index}|"++"{index}
log &&|"||"|!
num [0-9]+
str [a-zA-Z-_]
mode r|w|a|r+|w+|a+
ffile "\""{id}"\."{file}"\""

%%

{id}"="([+-]?{num})|("""{str}""")
{printf("\n%d\t%d\t\tdefinition\t%s",l,t,yytext);t++;}
"*"[ ]?{id}             {printf("\n%d\t%d\t\tpointer\t\t%s",l,t,yytext);t++;}
"//"[^\n]+"\n"          {printf("\n%d\t%d\t\tcomment\t\t",l,t);t++;
                         for(i=2;i<yyleng-1;i++)
                     printf("%c",yytext[i]);
                         l++;}
[;]                     {printf("\n%d\t%d\t\tterminator\t%s",l,t,yytext);t++;}
{func}                  {printf("\n%d\t%d\t\tfunction\t%s",l,t,yytext);t++;}
```

```
"%"{format}         {printf("\n%d\t%d\t\tformat\t\t%s",l,t,yytext);t++;}
["$&^{}(),'#]       {printf("\n%d\t%d\t\tspecial char\t%s",l,t,yytext);t++;}
[+-=*/%]            {printf("\n%d\t%d\t\toperator\t%s",l,t,yytext);t++;}
{log}               {printf("\n%d\t%d\t\tlogical op\t%s",l,t,yytext);t++;}
{key}               {printf("\n%d\t%d\t\tkeyword\t\t%s",l,t,yytext);t++;}
{hfile}             {printf("\n%d\t%d\t\theader file\t%s",l,t,yytext);t++;}
{id}                {printf("\n%d\t%d\t\tidentifier\t%s",l,t,yytext);t++;}
[a-z]+"."[a-z]+     {printf("\n%d\t%d\t\tidentifier\t%s",l,t,yytext);t++;}
{id}"++"            {printf("\n%d\t%d\t\tincrementer\t%s",l,t,yytext);t++;}
{id}"--"            {printf("\n%d\t%d\t\tdecrementer\t%s",l,t,yytext);t++;}
"=="                {printf("\n%d\t%d\t\tequality check\t%s",l,t,yytext);t++;}
"\\"[btvrfn?0a]     {printf("\n%d\t%d\t\tescape char\t%s",l,t,yytext);t++;}
[+-]?{num}          {printf("\n%d\t%d\t\tnumber\t\t%s",l,t,yytext);t++;}
{id}"["{s}"]"       {printf("\n%d\t%d\t\t1-D array\t%s",l,t,yytext);t++;}
"\'"[a-zA-Z0-9]"\'" {printf("\n%d\t%d\t\tcharacter\t%s",l,t,yytext);t++;}
"\""[a-zA-Z0-9]+"\""{printf("\n%d\t%d\t\tstring\t\t%s",l,t,yytext);t++;}
{id}"["{s}"]""["{s}"]"
                    {printf("\n%d\t%d\t\t2-D array\t%s",l,t,yytext);t++;}
"+="|"-="|"*="|"/="|"%="
                    {printf("\n%d\t%d\t\tcompound op\t%s",l,t,yytext);t++;}
"<="|">="|"<"|">"|"!="
                    {printf("\n%d\t%d\t\trelational op\t%s",l,t,yytext);t++;}
"("{type}")"        {printf("\n%d\t%d\t\ttype cast to\t",l,t);t++;
                     for(i=1;i<yyleng-1;i++)
                      printf("%c",yytext[i]);
                    }
"/*"[-_a-zA-Z \n]+"*/"
                    {i=0;
                     printf("\n%d\t%d\t\tcomment\t\t",l,t);t++;
                     for(i=2;i<yyleng-2;i++)
                     {
                      if(yytext[i]=='\n')    yytext[i]=' ';
                      printf("%c",yytext[i]);
                     }
                    }
{ffile}             {printf("\n%d\t%d\t\tspecial char\t\"",l,t);t++;
                     printf("\n%d\t%d\t\tfile\t\t",l,t);t++;
                     for(i=1;i<yyleng-1;i++)
                     printf("%c",yytext[i]);
                     printf("\n%d\t%d\t\tspecial char\t\"",l,t);t++;}
"printf("[a-zA-Z -_]+")"
                    {printf("\n%d\t%d\t\tfunction\tprintf",l,t);t++;
                     printf("\n%d\t%d\t\tspecial char\t(",l,t);t++;
                     printf("\n%d\t%d\t\toutput\t\t",l,t);t++;
                     for(i=7;i<yyleng-1;i++)
```

```
                         printf("%c",yytext[i]);
                        printf("\n%d\t%d\t\tspecial char\t)",l,t);t++;
                        }
("#include<"{hfile}">")|("#include\""[a-z]+"\."{file}"\"")
                        {i=0;
                        printf("\n%d\t%d\t\tpreprocessor\t#",l,t);t++;
                        printf("\n%d\t%d\t\tidentifier\tinclude",l,t);t++;
                        printf("\n%d\t%d\t\tspecial char\t%c",l,t,yytext[8]);t++;
                        printf("\n%d\t%d\t\theader file\t",l,t);t++;
                        for(i=9;i<yyleng-1;i++)
                        printf("%c",yytext[i]);
                        printf("\n%d\t%d\t\tspecial char\t%c",l,t,yytext[i]);t++;
                        }
"#define "[a-z]+" "[a-zA-Z0-9]+
                        {i=0;
                        printf("\n%d\t%d\t\tpreprocessor\t#",l,t);t++;
                        printf("\n%d\t%d\t\tidentifier\tdefine",l,t);t++;
                        printf("\n%d\t%d\t\tidentifier\t",l,t);t++;
                        for(i=8;yytext[i]!=' ';i++)
                         printf("%c",yytext[i]);
                        printf("\n%d\t%d\t\tconstant\t",l,t);t++; i++;
                        for(;i<yyleng;i++)
                         printf("%c",yytext[i]);
                        }
[\n]                    {l++;}

%%

int yywrap()
{       return 1;       }
int main()
{
        yyin=fopen("file.txt","r");
        printf("Line\tToken no\tToken name\tLexeme \n");
        yylex();
}
```

Testing:

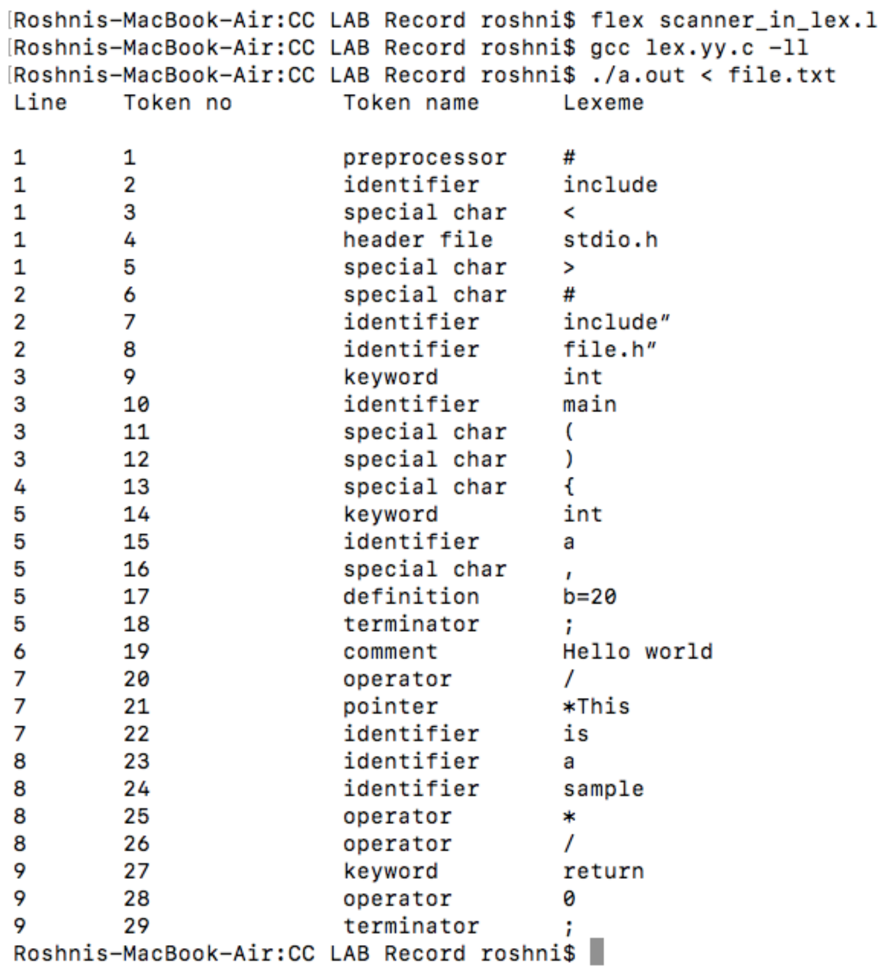
Input:

```
file.txt
#include<stdio.h>
#include"file.h"
int main()
{
        int a,b=20;
        //Hello world
        /*This is
        a sample*/
        return 0;
}
```

Expected Output:-

Line	Token no	Token name	Lexeme
2.	1	preprocessor	#
1.	2	identifier	include
2.	3	operator	<
1.	4	header	stdio.h
1.	5	operator	>
2	6	preprocessor	#
2	7	identifier	include
2	8	special symbol	“
2	9	header	file.h
2	10	special symbol	”
3	11	key word	int
3	12	identifier	main
3	13	special symbol	(
3	14	special symbol	)
4	15	special symbol	{
5	16	keyword	int
5	17	identifier	a
5	18	special symbol	,
5	19	identifier	b
5	20	operator	=
5	21	digit	20
5	22	special symbol	;
6	23	comment starts	//
6	24	comment	Hello world
7	25	comment starts	/*
7	26	comment	This is a sample
7	27	comment ends	*/

Actual Output:-



```
[Roshnis-MacBook-Air:CC LAB Record roshni$ flex scanner_in_lex.l
[Roshnis-MacBook-Air:CC LAB Record roshni$ gcc lex.yy.c -ll
[Roshnis-MacBook-Air:CC LAB Record roshni$ ./a.out < file.txt
Line    Token no        Token name      Lexeme

1       1               preprocessor    #
1       2               identifier      include
1       3               special char    <
1       4               header file     stdio.h
1       5               special char    >
2       6               special char    #
2       7               identifier      include"
2       8               identifier      file.h"
3       9               keyword         int
3       10              identifier      main
3       11              special char    (
3       12              special char    )
4       13              special char    {
5       14              keyword         int
5       15              identifier      a
5       16              special char    ,
5       17              definition      b=20
5       18              terminator      ;
6       19              comment         Hello world
7       20              operator        /
7       21              pointer         *This
7       22              identifier      is
8       23              identifier      a
8       24              identifier      sample
8       25              operator        *
8       26              operator        /
9       27              keyword         return
9       28              operator        0
9       29              terminator      ;
Roshnis-MacBook-Air:CC LAB Record roshni$ 
```

Result:-

Successfully executed the program.

Program:-

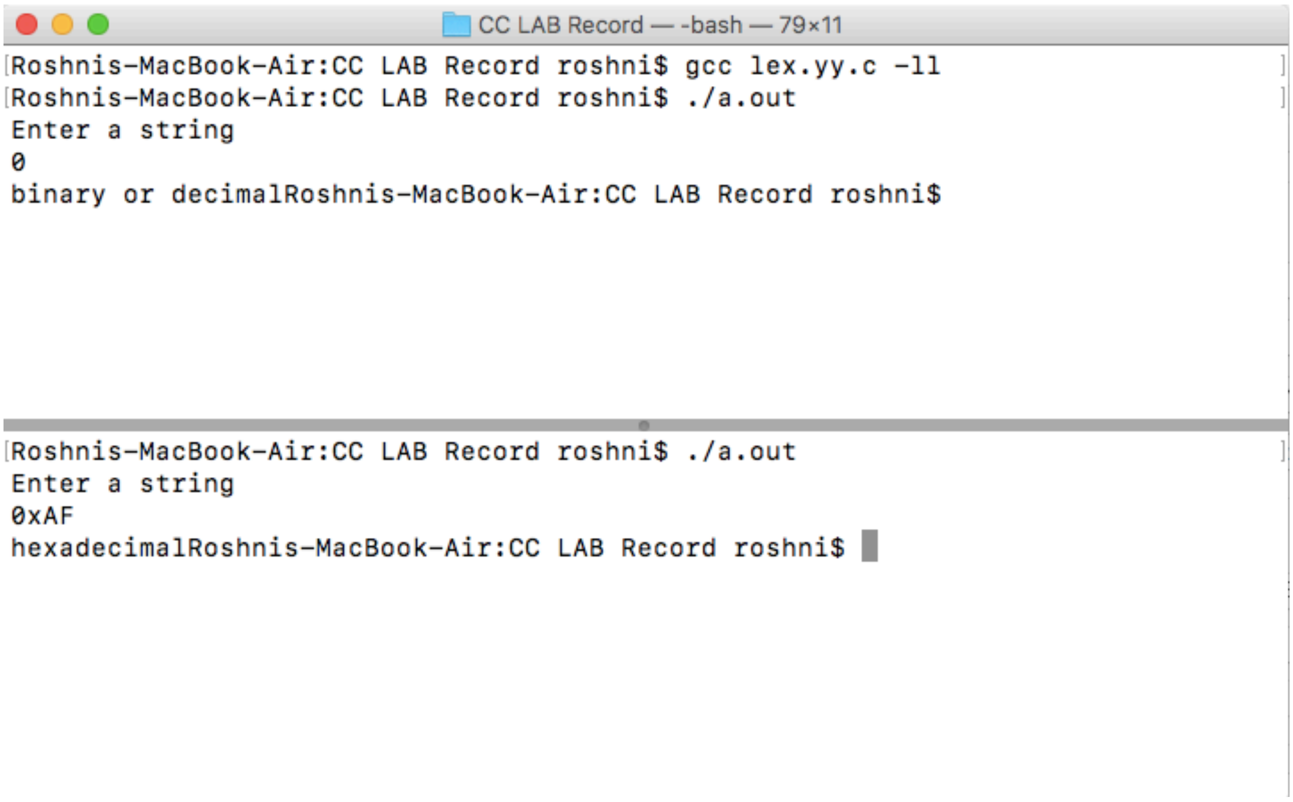
```
//Karedla 160114733091
//Program to identify whether a given a number is Decimal,Octal or Hexa-Decimal

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

%%

[0]                         printf("binary or decimal");
[10]*                       printf("binary");
[1-9][0-9]*                 printf("decimal");
[0][0-7]+                   printf("octal");
[0][xX][0-9a-fA-F]+         printf("hexadecimal");
[\n] return 0;

%%
int yywrap()
{
    return 1;
}

int main()
{
    printf("Enter a string\n");
    yylex();
}
```

Testing:-

Input:-
0
0xAF

Expected Output:-
binary or decimal
hexadecimal

Actual Output:-



```
Roshnis-MacBook-Air:CC LAB Record roshni$ gcc lex.yy.c -ll
Roshnis-MacBook-Air:CC LAB Record roshni$ ./a.out
Enter a string
0
binary or decimalRoshnis-MacBook-Air:CC LAB Record roshni$
```

```
Roshnis-MacBook-Air:CC LAB Record roshni$ ./a.out
Enter a string
0xAF
hexadecimalRoshnis-MacBook-Air:CC LAB Record roshni$
```

Program:-

```
//Karedla 160114733091
//Program to capitalise a given string

%{
#include<stdio.h>
%}
%%
[a-z] {printf("%c",(char)(yytext[0]-32));}
[A-Z] {printf("%c",(char)(yytext[0]+32));}
%%
int main()
{
yylex();
}
```

Testing:-

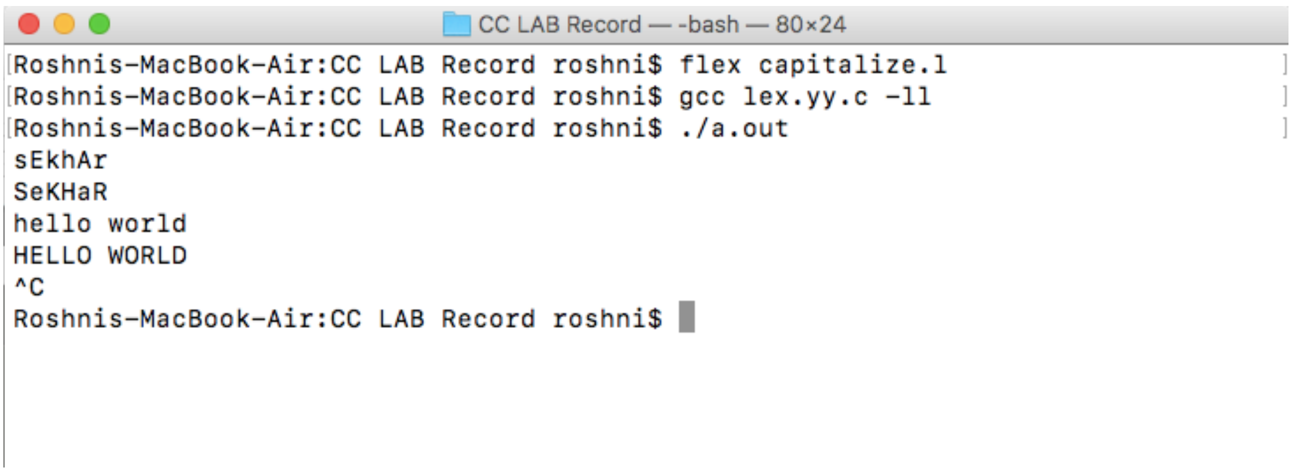
Input:-

```
sEkhAr
SeKHaR
```

Expected Output:-

```
hello world
HELLO WORLD
```

Actual Output:-



```
Roshnis-MacBook-Air:CC LAB Record roshni$ flex capitalize.l
Roshnis-MacBook-Air:CC LAB Record roshni$ gcc lex.yy.c -ll
Roshnis-MacBook-Air:CC LAB Record roshni$ ./a.out
sEkhAr
SeKHaR
hello world
HELLO WORLD
^C
Roshnis-MacBook-Air:CC LAB Record roshni$
```

Result:-
Successfully executed the program.

Program:-

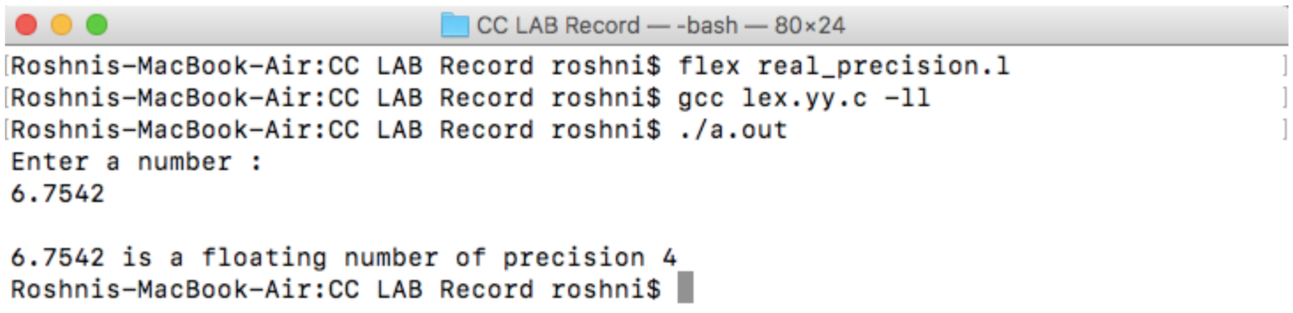
```
//Karedla 160114733091
//Program to find real precision numbers using LEX.

%{
#include<stdio.h>
#include<string.h>
int f,i,j;
%}

%%

[+-]?[0-9]+   {printf("\n%s is an integer!!!",yytext);}
[+-]?[0-9]*[.][0-9]+
        {f=0; for(i=0;i<yyleng;i++)
          if(yytext[i]=='.')
           { j=i+1; break;}
          for(;j<yyleng;j++)
           f++;
          printf("\n%s is a floating number with a precision of %d!!!",yytext,f);}
[0-9a-zA-Z]+[.][0-9+-.a-zA-Z]+          {printf("\ninvalid!!!");}
[\n]              {return 0;}

%%
int main()
{
printf("Enter a number :\n");
yylex();
}
int yywrap()
{
return 1;
}
```

Testing:-

Input:-
6.7542

Expected Output:-
6.7542 is a floating number of precision 4

Actual Output:-



```
Roshnis-MacBook-Air:CC LAB Record roshni$ flex real_precision.l
Roshnis-MacBook-Air:CC LAB Record roshni$ gcc lex.yy.c -ll
Roshnis-MacBook-Air:CC LAB Record roshni$ ./a.out
Enter a number :
6.7542

6.7542 is a floating number of precision 4
Roshnis-MacBook-Air:CC LAB Record roshni$
```

Result:-
Successfully executed the Program.

Program:-

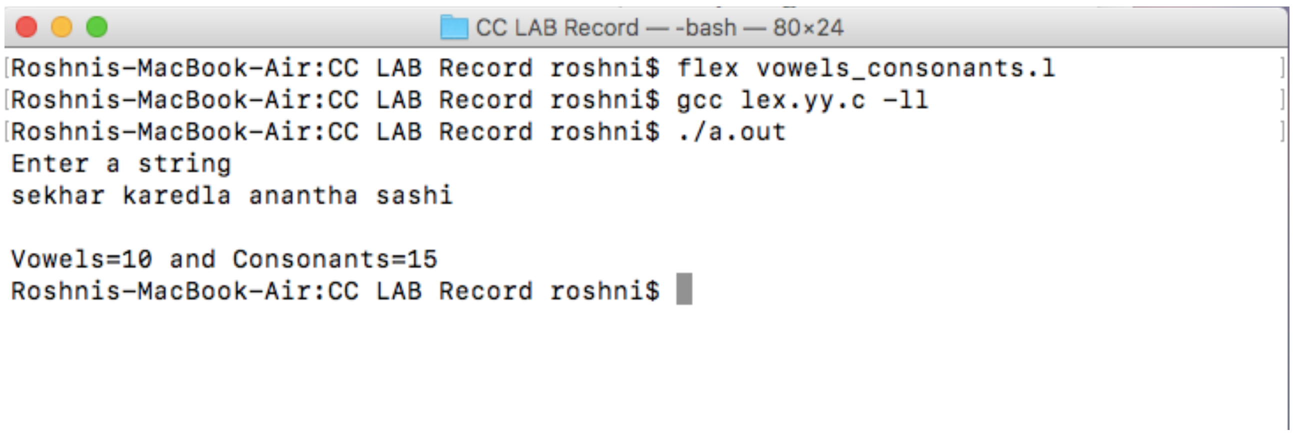
```
//Karedla 160114733091
//Program to find number of consonants and vowels

%{
#include<stdio.h>
int vowel=0;
int cons=0;
%}

%%
[aeiouAEIOU] {vowel++;}
[a-zA-Z] {cons++;}
[\n]  { printf("\nVowels=%d and Consonants=%d\n",vowel,cons); return 0;}
%%

int yywrap()
{
	return 1;
}
int main()
{
	printf("Enter a string\n");
	yylex();
}
```

Testing:-

Input:-
Sekhar Karedla Anantha Sashi

Expected Output:-
Vowels=10 and Consonants=15

Actual Output:-



```
Roshnis-MacBook-Air:CC LAB Record roshni$ flex vowels_consonants.l
Roshnis-MacBook-Air:CC LAB Record roshni$ gcc lex.yy.c -ll
Roshnis-MacBook-Air:CC LAB Record roshni$ ./a.out
Enter a string
sekhar karedla anantha sashi

Vowels=10 and Consonants=15
Roshnis-MacBook-Air:CC LAB Record roshni$
```

Result:-

Successfully executed the program.

Program:-
//Karedla 160114733091
//Program to implement calculator using yacc tool

Simplecalc.l

```
%{
#include<stdio.h>
#include "y.tab.h"
%}

%%
[0-9]+  {yylval.dval = atoi( yytext ); return DIGIT;}
\n|.    {return yytext[0];}
%%
```

Simplecalc.y

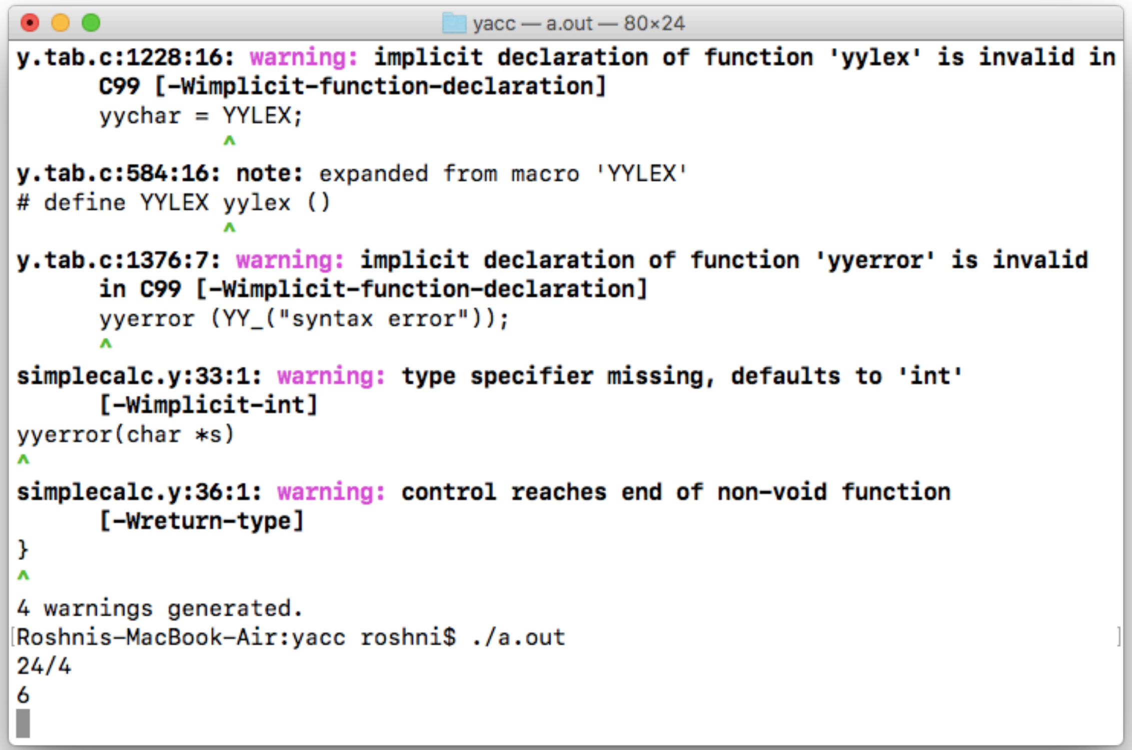
```
%{
#include<stdio.h>
/*E->E+E|E-E|E*E|E/E|(E)|DIGIT comment grammar*/
%}

%union
{
        int dval;
}

%token  <dval> DIGIT
%type   <dval> expr
%type   <dval> expr1

%%
line:expr '\n'{printf("%d\n",$1);}
    ;
expr:expr '+' expr1 {$$=$1+$3;}
    |expr '-' expr1 {$$=$1-$3;}
    |expr '*' expr1 {$$=$1*$3;}
    |expr '/' expr1 {$$=$1/$3;}
    |expr1
    ;
expr1:'('expr')' {$$=$2;}
    |DIGIT
    ;
%%



int main()
{
        yyparse ();
}
yyerror(char *s)
{
        printf("%s",s);
}
```

Testing:-

Input:-
24/4

Expected Output:-
6

Actual Output:-



yacc — a.out — 80×24

```
y.tab.c:1228:16: warning: implicit declaration of function 'yylex' is invalid in
      C99 [-Wimplicit-function-declaration]
      yychar = YYLEX;
               ^
y.tab.c:584:16: note: expanded from macro 'YYLEX'
# define YYLEX yylex ()
               ^
y.tab.c:1376:7: warning: implicit declaration of function 'yyerror' is invalid
      in C99 [-Wimplicit-function-declaration]
      yyerror (YY_("syntax error"));
      ^
simplecalc.y:33:1: warning: type specifier missing, defaults to 'int'
      [-Wimplicit-int]
yyerror(char *s)
^
simplecalc.y:36:1: warning: control reaches end of non-void function
      [-Wreturn-type]
}
^
4 warnings generated.
Roshnis-MacBook-Air:yacc roshni$ ./a.out
24/4
6
```

Result:-
Successfully executed the program.

Program:-

//Karedla 160114733091
//Program to recognise string a^nb^n for n>=0

Anbn.l

```
%{
#include<stdio.h>
#include"y.tab.h"
%}
%%
a return A;
b return B;
\n|. return yytext[0];
%%
```

Anbn.y

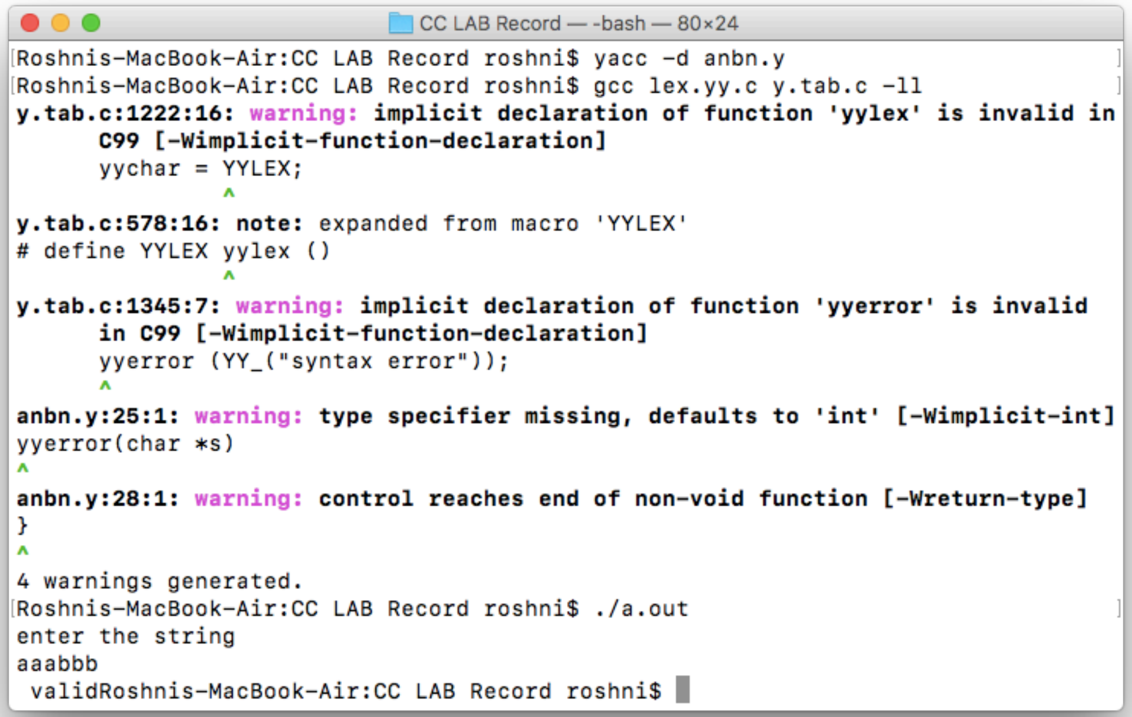
```
%{
#include<stdio.h>
int vd;
%}
%union
{
        char dval;
}
%token <dval> A
%token <dval> B
%%
str:s'\n' { vd=1; return 0;}
s:A s B   ;
 |  ;
%%
int main()
{
        printf("enter the string\n");
        yyparse();
        if(vd==1)
        printf(" valid");
        else
        printf(" not valid");
}
yyerror(char *s)
{
        printf("%s",s);
}
```

Testing:-

Input:-
aaabbb

Expected Output:-
valid

Actual Output:-



```
Roshnis-MacBook-Air:CC LAB Record roshni$ yacc -d anbn.y
Roshnis-MacBook-Air:CC LAB Record roshni$ gcc lex.yy.c y.tab.c -ll
y.tab.c:1222:16: warning: implicit declaration of function 'yylex' is invalid in
      C99 [-Wimplicit-function-declaration]
      yychar = YYLEX;
               ^
y.tab.c:578:16: note: expanded from macro 'YYLEX'
# define YYLEX yylex ()
               ^
y.tab.c:1345:7: warning: implicit declaration of function 'yyerror' is invalid
      in C99 [-Wimplicit-function-declaration]
      yyerror (YY_("syntax error"));
      ^
anbn.y:25:1: warning: type specifier missing, defaults to 'int' [-Wimplicit-int]
yyerror(char *s)
^
anbn.y:28:1: warning: control reaches end of non-void function [-Wreturn-type]
}
^
4 warnings generated.
Roshnis-MacBook-Air:CC LAB Record roshni$ ./a.out
enter the string
aaabbb
 validRoshnis-MacBook-Air:CC LAB Record roshni$ 
```

Result:-
Successfully executed the program.

Program:-

//Karedla 160114733091
//Program to count number of positive and negative numbers using lex

```
%{

    int postiveno=0;
    int negtiveno=0;
    int positivefractions=0;
    int negativefractions=0;

%}

DIGIT [0-9]
%%

\+?{DIGIT}+                              postiveno++;
-{DIGIT}+                                  negtiveno++;

\+?{DIGIT}*\.{DIGIT}+             positivefractions++;
-{DIGIT}*\.{DIGIT}+                 negativefractions++;
. ;
%%

main()
{
    yylex();
    printf("\nNo. of positive numbers: %d",postiveno);
    printf("\nNo. of Negative numbers: %d",negtiveno);
    printf("\nNo. of Positive fractions: %d",positivefractions);
    printf("\nNo. of Negative fractions: %d\n",negativefractions);
}

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

Testing:-

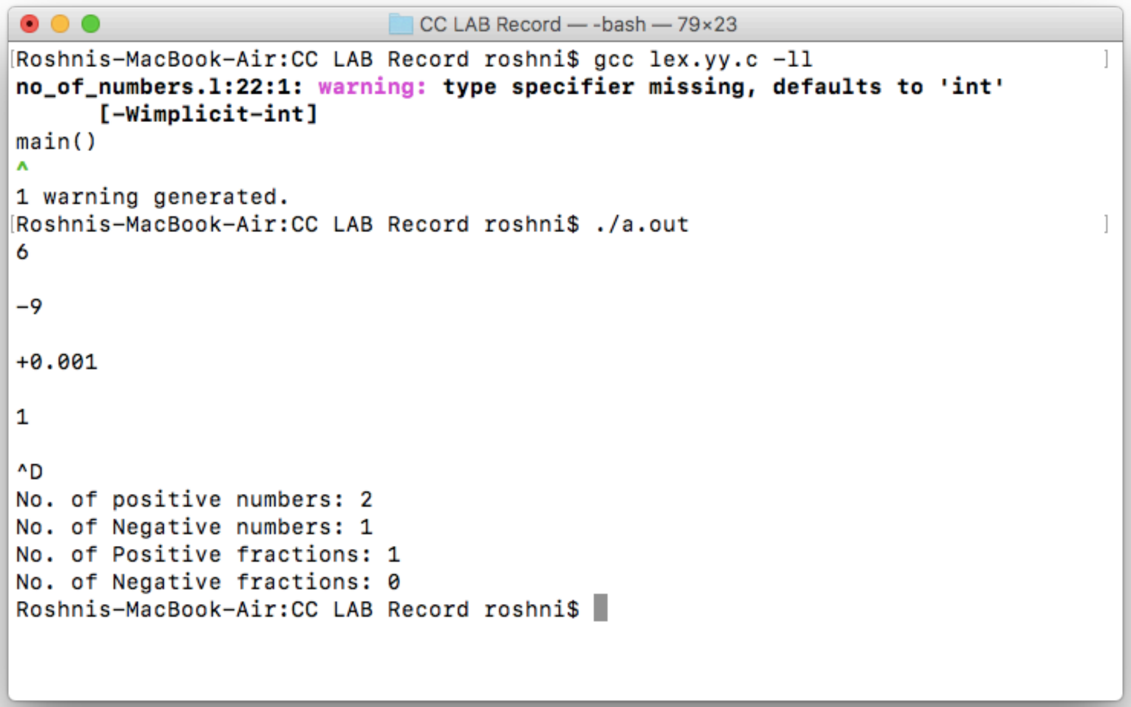
Input:-

```
6

-9

+0.001

1
```

Expected Output:-

```
No. of positive numbers: 2
No. of Negative numbers: 1
No. of Positive fractions: 1
No. of Negative fractions: 0
```

Actual Output:-



```
Roshnis-MacBook-Air:CC LAB Record roshni$ gcc lex.yy.c -ll
no_of_numbers.l:22:1: warning: type specifier missing, defaults to 'int'
      [-Wimplicit-int]
main()
^
1 warning generated.
Roshnis-MacBook-Air:CC LAB Record roshni$ ./a.out
6

-9

+0.001

1

^D
No. of positive numbers: 2
No. of Negative numbers: 1
No. of Positive fractions: 1
No. of Negative fractions: 0
Roshnis-MacBook-Air:CC LAB Record roshni$ 
```

Result:-
Successfully executed the program.

Program:-

//Karedla 160114733091

//Program to count number of printf and scanf and replace them with readf and writef

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

%%

"printf" {fprintf(yyout,"writef"); pfc++;}
"scanf" {fprintf(yyout,"readf"); sfc++;}
%%

main(int argc, char *argv[])
{
if(argc!=3)
{
printf("Usage: ./a.out in.txt out.txt\n");
exit(0);
}
yyin=fopen(argv[1],"r");
yyout=fopen(argv[2],"w");
yylex();
printf("\n the number of printf lines = %d\n",pfc);
printf("\n the number of scanf lines = %d\n",sfc);
}

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

Testing:-

Input:-

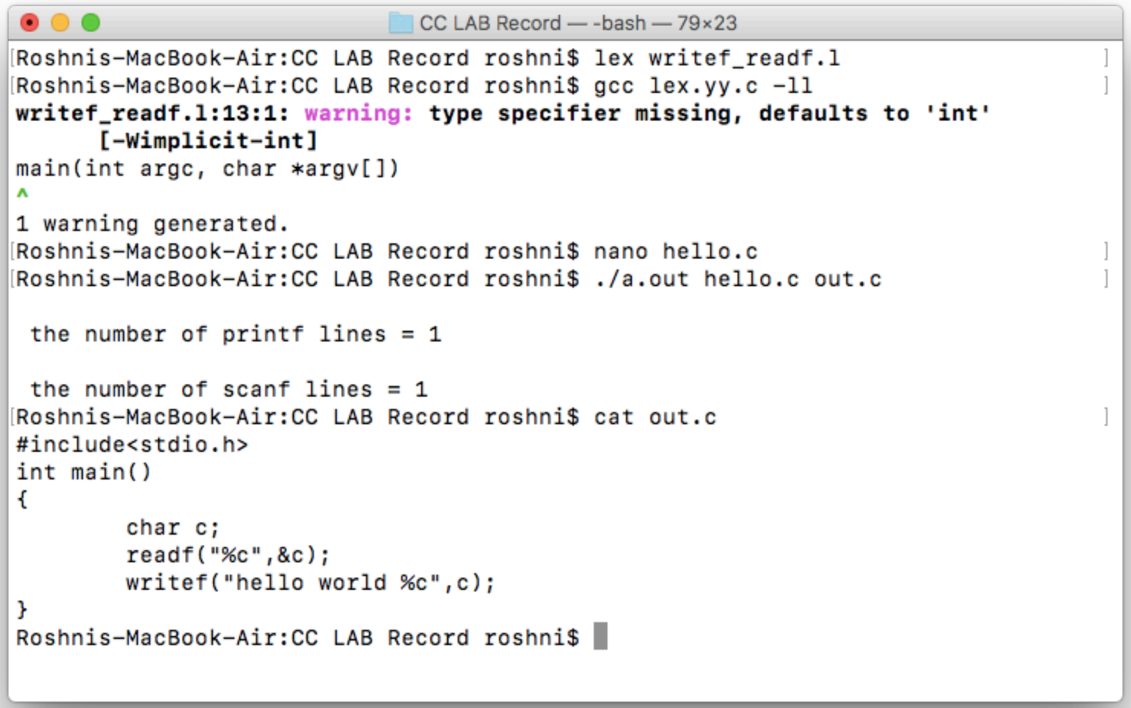
```
#include<stdio.h>
int main()
{
    char c;
    scanf("%c",&c);
    printf("hello world %c",c);
}
```

Expected Output:-

```
#include<stdio.h>
int main()
{
    char c;
```

```
        readf("%c",&c);
        writef("hello world %c",c);
    }
```

Actual Output:-



CC LAB Record — -bash — 79×23

```
Roshnis-MacBook-Air:CC LAB Record roshni$ lex writef_readf.l
Roshnis-MacBook-Air:CC LAB Record roshni$ gcc lex.yy.c -ll
writef_readf.l:13:1: warning: type specifier missing, defaults to 'int'
      [-Wimplicit-int]
main(int argc, char *argv[])
^
1 warning generated.
Roshnis-MacBook-Air:CC LAB Record roshni$ nano hello.c
Roshnis-MacBook-Air:CC LAB Record roshni$ ./a.out hello.c out.c

 the number of printf lines = 1

 the number of scanf lines = 1
Roshnis-MacBook-Air:CC LAB Record roshni$ cat out.c
#include<stdio.h>
int main()
{
        char c;
        readf("%c",&c);
        writef("hello world %c",c);
}
Roshnis-MacBook-Air:CC LAB Record roshni$ 
```

Program:-

```
//Karedla 160114733091
//Program to find the First of a Grammar

n=input("enter number of productions :")
head=[]
body=[]
for k in range(0,n):
        prod1=raw_input("enter the productions:")
        prod1=prod1.split('->')
        head.append(prod1[0])
        body.append(prod1[1])
#print head
#print body
first={}
i=n-1

def isSmall(k):
        if ord(k)>=97 and ord(k)<=122:
                return True
        else:
                return False

for o in range(0,n):
        k=head[i]
        l=body[i].split('|')
        list=[]
#       print k
#       print l
        for m in l:
#               print m
                if isSmall(m[0]):
                        list.append(m[0])
                else:
                        list.append(''.join(first[m[0]]))

        first[k]=list
        i-=1
#print first
for k in first.keys():
        print 'first of '+k+' is: '+''.join(first[k])
```

Testing:-

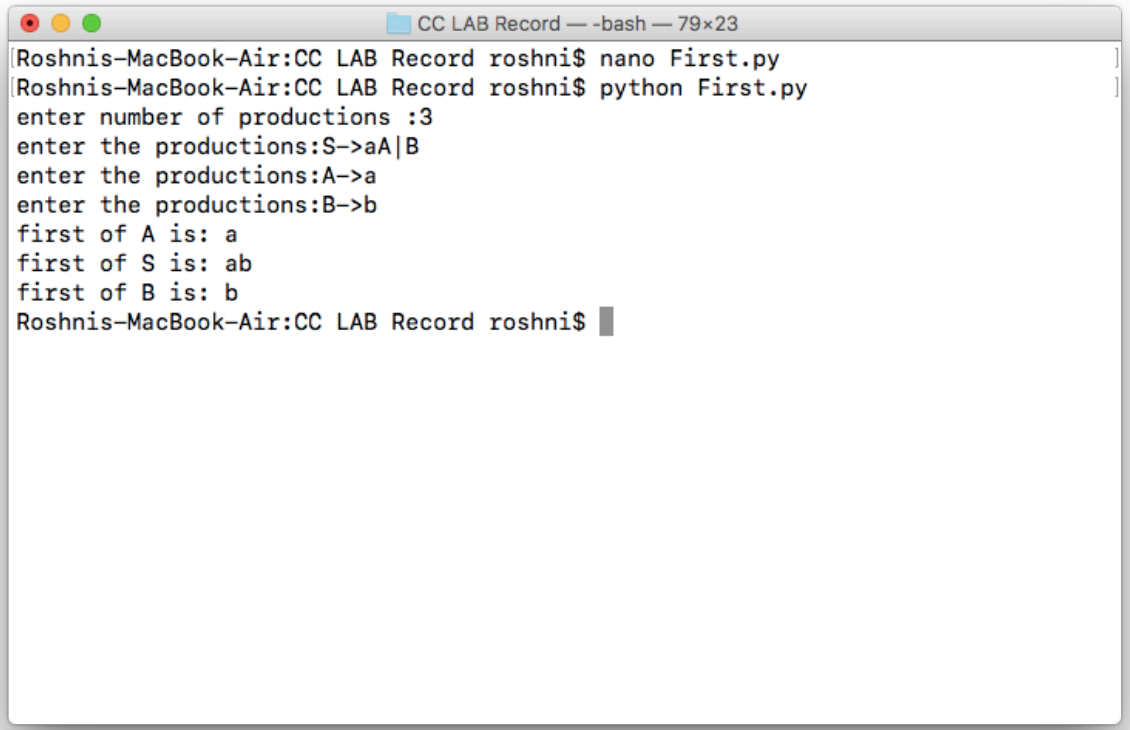
Input:-

```
enter number of productions :3
enter the productions:S->aA|B
enter the productions:A->a
enter the productions:B->b
```

Expected Output:-

```
first of A is: a
first of S is: ab
first of B is: b
```

Actual Output:-



```
Roshnis-MacBook-Air:CC LAB Record roshni$ nano First.py
Roshnis-MacBook-Air:CC LAB Record roshni$ python First.py
enter number of productions :3
enter the productions:S->aA|B
enter the productions:A->a
enter the productions:B->b
first of A is: a
first of S is: ab
first of B is: b
Roshnis-MacBook-Air:CC LAB Record roshni$
```

Program:-

//Karedla 160114733091

//Program to find the number of comments and to remove comments and add to a file

```
%{
        #include<stdio.h>
        int c=0,m=0;
%}
%%
[/][/]([a-zA-Z0-9]*|[\t]?)+  {c++;}
[/][*]([a-zA-Z0-9 ]*|[\n]?|[\t]?)+[*][/] {m++;}
%%
int main(int argc,char *argv[])
{
        yyin = fopen(argv[1],"r");
        yyout = fopen(argv[2],"w");
        yylex();
        printf("Number of single line comments %d\n",c);
        printf("Number of multiline coments %d\n",m);
}
int yywrap()
{
return 1;
}
```

Testing:-

Input:-

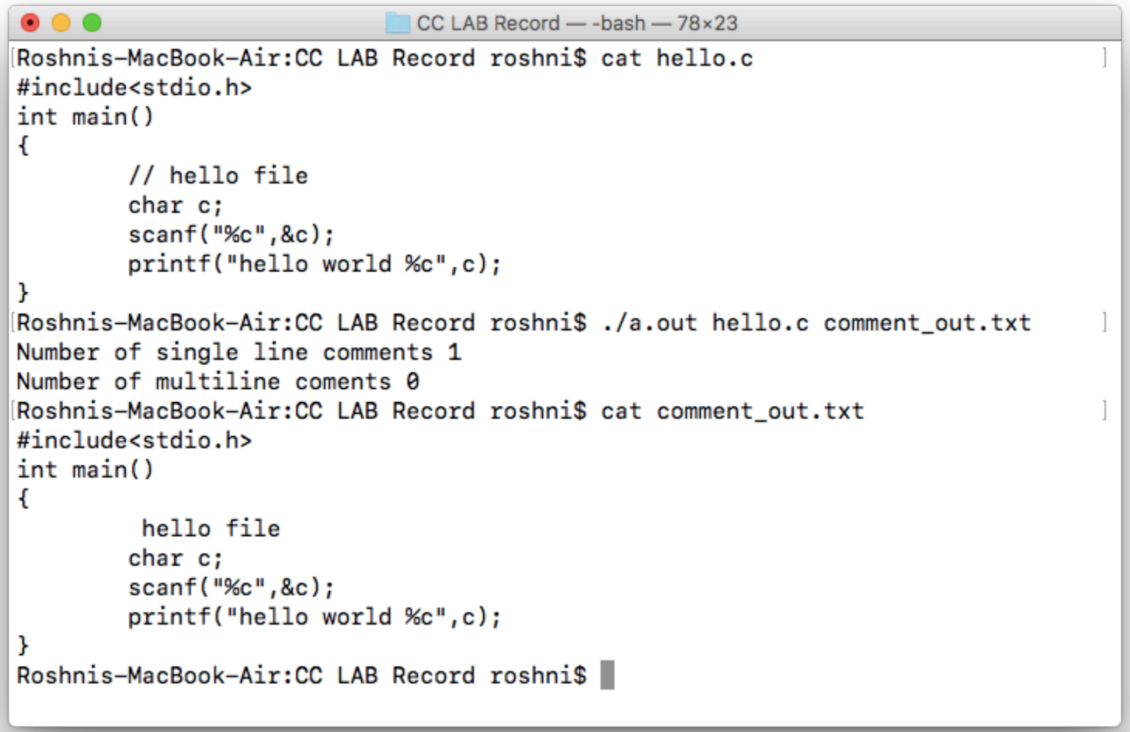
```
#include<stdio.h>
int main()
{
   // hello file
   char c;
   scanf("%c",&c);
   printf("hello world %c",c);
}
```

Expected Output:-

```
Number of single line comments 1
Number of multiline coments 0
```

```
#include<stdio.h>
int main()
{
     hello file
    char c;
    scanf("%c",&c);
    printf("hello world %c",c);
}
```

Actual Output:-



```
Roshnis-MacBook-Air:CC LAB Record roshni$ cat hello.c
#include<stdio.h>
int main()
{
        // hello file
        char c;
        scanf("%c",&c);
        printf("hello world %c",c);
}
Roshnis-MacBook-Air:CC LAB Record roshni$ ./a.out hello.c comment_out.txt
Number of single line comments 1
Number of multiline coments 0
Roshnis-MacBook-Air:CC LAB Record roshni$ cat comment_out.txt
#include<stdio.h>
int main()
{
         hello file
        char c;
        scanf("%c",&c);
        printf("hello world %c",c);
}
Roshnis-MacBook-Air:CC LAB Record roshni$
```

Result:-
Executed Program successfully

Program:-

```
//Karedla 160114733091
//Program to find Follows of a Grammar

n=input("enter number of productions :")
head=[]
body=[]
for k in range(0,n):
	prod1=raw_input("enter the productions:")
	prod1=prod1.split('->')
	head.append(prod1[0])
	body.append(prod1[1])
#print head
#print body
first={}
i=n-1

def isSmall(k):
	if ord(k)>=97 and ord(k)<=122:
		return True
	else:
		return False

for o in range(0,n):
	k=head[i]
	l=body[i].split('|')
	list=[]
#	print k
#	print l
	for m in l:
#		print m
		if isSmall(m[0]):
			list.append(m[0])
		else:
			list.append(''.join(first[m[0]]))

	first[k]=list
	i-=1
#print first
for k in first.keys():
	print 'first of '+k+' is: '+''.join(first[k])

#follows from here
follows={}
```

```
follows['S']=['^']
for o in range(0,n):
        k=head[o]
        count=0
        for l in body:
                count1=0
                for m in l:
                        if k==m:
                                if count1==len(l)-1:
                                        follows[k]=follows[head[count]]
                                else:
                                        if l[count1+1]=='|':
                                                follows[k]=follows[head[count]]
                                        elif isSmall((l[count1+1])):
                                                follows[k]=l[count1+1]
                                        else:
                                                follows[k]=first[l[count1+1]]

                        count1+=1
                count+=1

#print follows
for k in follows.keys():
        print 'follows of '+str(k)+' is: ‘+''.join(follows[k])
```

Testing:-

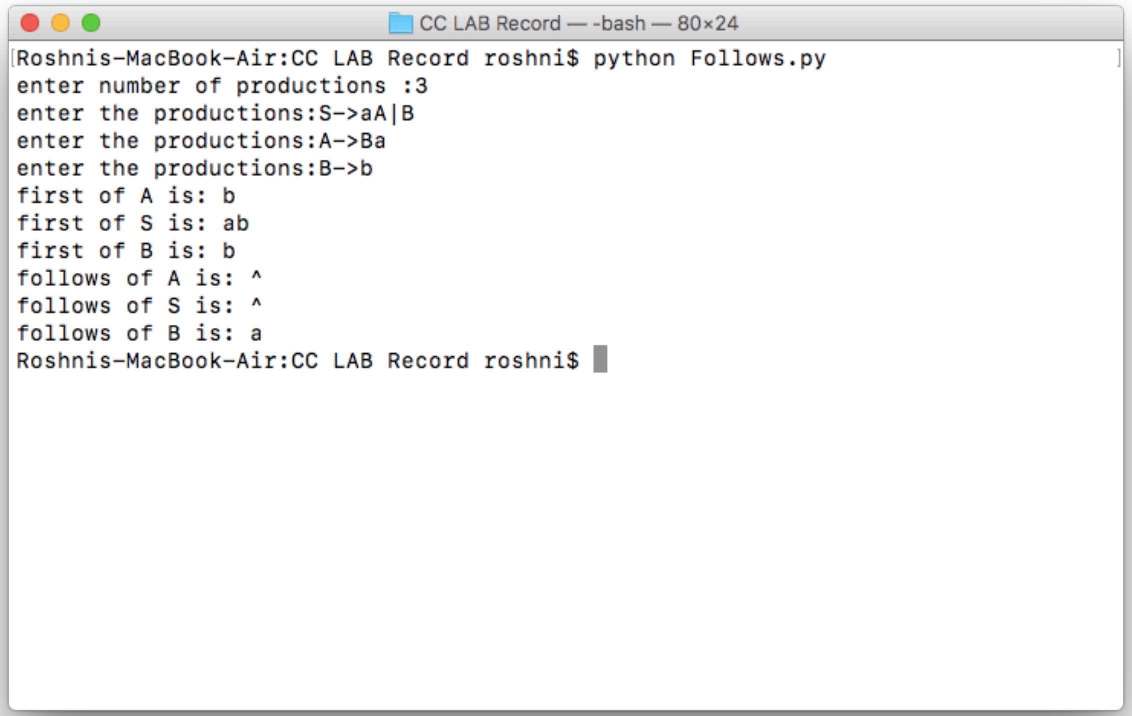
Input:-

```
enter number of productions :3
enter the productions:S->aA|B
enter the productions:A->Ba
enter the productions:B->b
```

Expected Output:-

```
first of A is: b
first of S is: ab
first of B is: b
follows of A is: ^
follows of S is: ^
follows of B is: a
```

Actual Output:-



```
Roshnis-MacBook-Air:CC LAB Record roshni$ python Follows.py
enter number of productions :3
enter the productions:S->aA|B
enter the productions:A->Ba
enter the productions:B->b
first of A is: b
first of S is: ab
first of B is: b
follows of A is: ^
follows of S is: ^
follows of B is: a
Roshnis-MacBook-Air:CC LAB Record roshni$
```