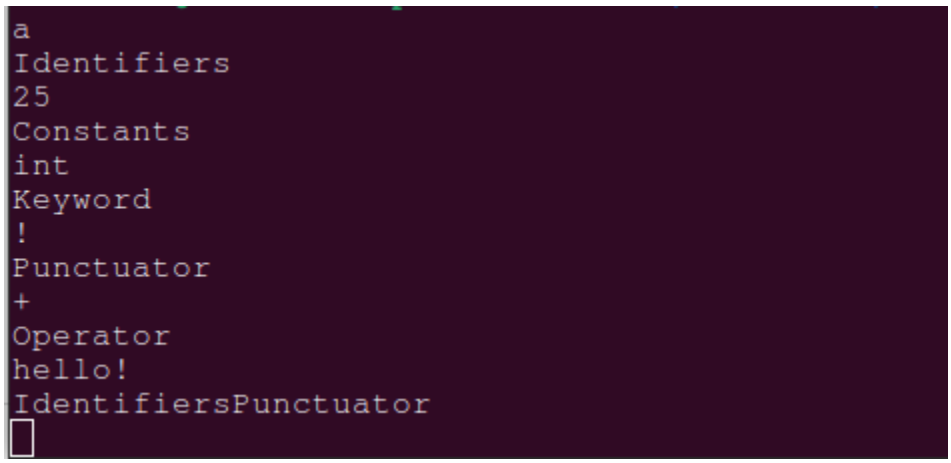
1. Write a program in LEX to recognize different tokes:Keywords, Identifiers, Constants, Operators and Punctuations?

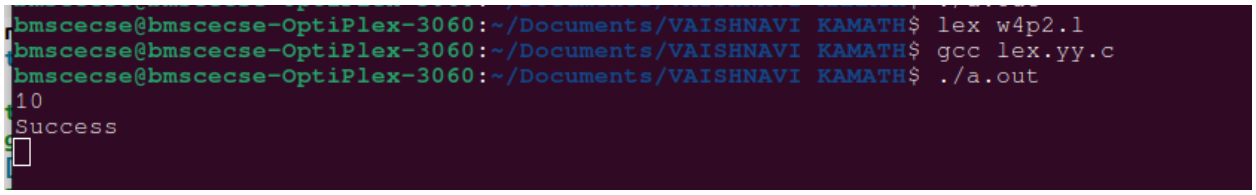
```
%{
#include<stdio.h>
int cnt=0;
%}
letter [a-zA-Z]
digit [0-9]
punc [!|,|.]
oper [+|*|-|/|%]
boole [true|false]
%%
{digit}+|{digit}*.{digit}+ {printf("Constants");}
int|float {printf("Keyword");}
{letter}({digit}|{letter})* {printf("Identifiers");}
{oper} {printf("Operator");}
{punc} {printf("Punctuator");}

%%

int yywrap()
{
}

int main()
{
yylex();
return 0;
}
```



2.The set of all strings beginning with a 1 which, interpreted as the binary representation of an integer, is congruent to zero modulo 5.

```
%{
#include<stdio.h>
int num,i,r,b=0,p=1;
%}
letter [a-zA-Z]
digit [0-9]
A [0-9]
punc [!|,|.]
oper [+|*|-|/|%]
boole [true|false]
%%
{A}+ {num=atoi(yytext);
if(yytext[0]=='1'){
while(num>0){
r=num%2;
b=b+r*p;
p=p*10;
num=num/2;
}
if(b%10==0){
printf("Success");
}
else {
printf("Fail");
printf("%d",b);
}
}
}
```

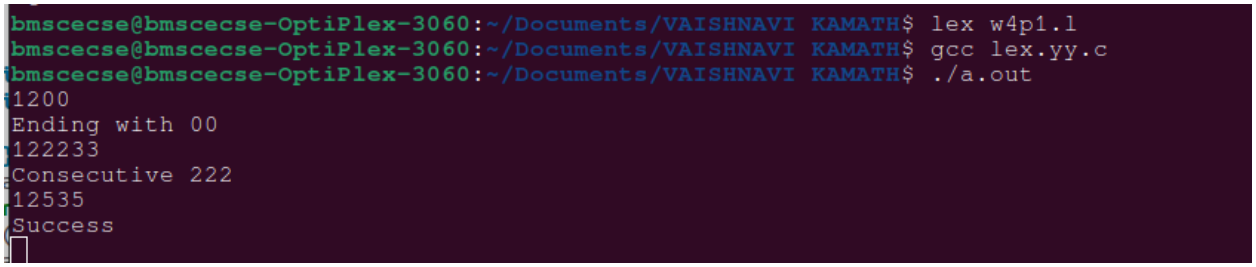
```
%%

int yywrap()
{
}

int main()
{
yylex();
return 0;
}
```



3. Write a LEX program to recognize the following tokens over the alphabets{0,1,..,9}
a) The set of all string ending in 00.
b) The set of all strings with three consecutive 222's.
c) The set of all string such that every block of five consecutive symbols contains at least two 5's.

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

letter [a-zA-Z]
digit [0-9]
A [0-9]
punc [!|,|.]
oper [+|*|-|/|%]
boole [true|false]


%%
{digit}*00 {printf("Ending with 00");}
{digit}*222{digit}* {printf("Consecutive 222");}

{A}{A}{A}{A}{A}  {
flag=0;
for(i=0;i<yyleng;i++){
if(yytext[i]=='5'){
flag=flag+1;
}
}
if(flag>=2){
printf("Success");
}
else{
printf("Failure");
}
```

```
}

%%

int yywrap()
{
}

int main()
{
yylex();
return 0;
}
```



Extra Program

```
d[0-9]
%{
/* d is for recognising digits */
int c1=0,c2=0,c3=0,c4=0,c5=0,c6=0,c7=0;
/* c1 to c7 are counters for rules a1 to a7 */
%}
%%
({d})*00 { c1++; printf("%s rule A\n",yytext);}
({d})*222({d})* { c2++; printf("%s rule B\n",yytext);}
(1(0)*(11|01)(01*01|00*10(0)*(11|1))*0)(1|10(0)*(11|01)(01*01|00*10(0)*(11|1))*10)* {
c4++;
printf("%s rule D \n",yytext);
}
({d})*1{d}{9} {
c5++; printf("%s rule E \n",yytext);
}
({d})* {

int i,c=0;
if(yyleng<5)
{
  printf("%s doesn't match any rule\n",yytext);
}
else
{

for(i=0;i<5;i++) { if(yytext[i]=='5') {
c++; } }
if(c>=2)
{
for(;i<yyleng;i++)
{
if(yytext[i-5]=='5') {
c--; }
if(yytext[i]=='5') { c++;
}
if(c<2) { printf("%s doesn't match any rule\n",yytext);
break; }
}
if(yyleng==i)
{
```

```
printf("%s ruleC\n",yytext); c3++; }
}
else
{
printf("%s doesn't match any rule\n",yytext);
}
}
}
%%
int yywrap()
{
}
int main()
{
printf("Enter text\n");
yylex();
printf("Total number of tokens matching rules are : \n");
printf("Rule A : %d \n",c1);
printf("Rule B : %d \n",c2);
printf("Rule C : %d \n",c3);
printf("Rule D : %d \n",c4);
printf("Rule E : %d \n",c5);

return 0;
}
```