Q.6>>Tokenization of a c prog.

```
%{

%}

%%

[0-9]+"."[0-9]+ {printf("%s is a floating constant\n",yytext);}

[0-9]+ {printf("%s is a int constant\n",yytext);}

"if"|"else"|"do"|"while" {printf("%s is a keywoard\n",yytext);}

[A-Za-z][A-Za-z0-9_]* {printf("%s is an identifier\n",yytext);}

"++"|">=" {printf("%s is a multioperator\n",yytext);}

[=-] {printf("%s is an operator\n",yytext);}

[(){};] {printf("%s is a seperator\n",yytext);}

. ;

%%

int main()

{

        printf("enter the string: ");

        yylex();

        return 0;

}

int yywrap()

{

return 1;

}
```

```
rajasree@ubuntu-RajasreeVM:~/Desktop/lex$ lex tokenization_cprog.l
rajasree@ubuntu-RajasreeVM:~/Desktop/lex$ gcc lex.yy.c
rajasree@ubuntu-RajasreeVM:~/Desktop/lex$ ./a.out
enter the string: do{
if(p>=0)
  q++;
else
  q=q-1.0;
}while(1);do is a keywoard
{ is a seperator

if is a keywoard
( is a seperator
p is an identifier
>= is a multioperator
0 is a int constant
) is a seperator

q is an identifier
++ is a multioperator
; is a seperator

else is a keywoard

q is an identifier
= is an operator
q is an identifier
- is an operator
1.0 is a floating constant
; is a seperator

} is a seperator
while is a keywoard
( is a seperator
1 is a int constant
) is a seperator
; is a seperator
rajasree@ubuntu-RajasreeVM:~/Desktop/lex$
```