10/9/2020
Thursday

Week-5

1. Consider the input expression and identify the tokens in expression:

   i. c=a++ + ++b

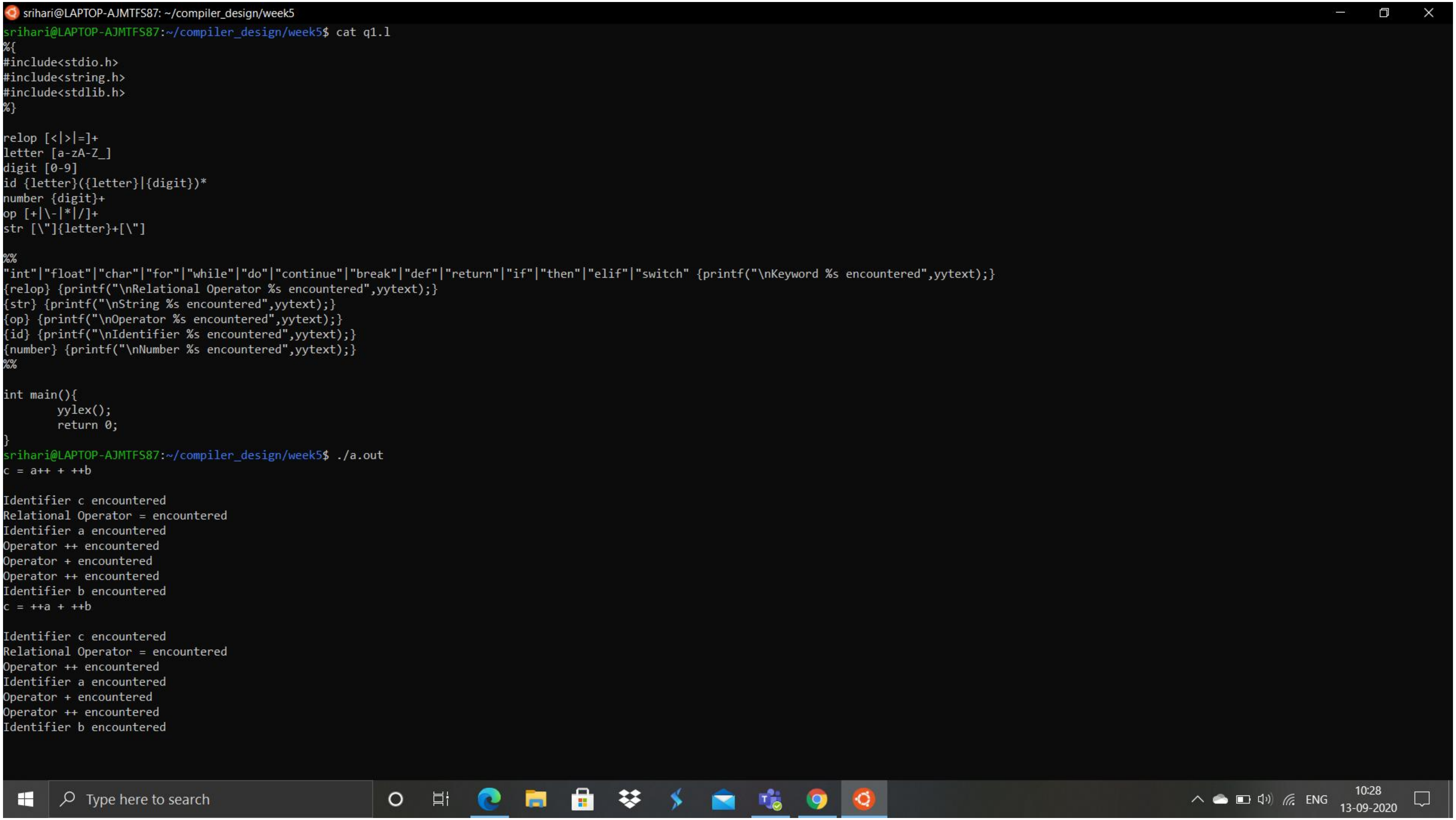
   ii. c=++a + ++b

```
%{
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
%}
relop [<|>|=]+
letter [a-zA-Z_]
digit [0-9]
id {letter}({letter}|{digit})*
number {digit}+
op [+|\-|*|/]+
str [\"]{letter}+[\"]

%%
"int"|"float"|"char"|"for"|"while"|"do"|"continue"|"break"|"def"|
  "return"|"if"|"then"|"elif"|"switch" {printf("\nKeyword
        %s encountered", yytext);}
{relop} {printf("\nRelational Operator %s encountered",yytext);}
{str}{printf("\nString %s encountered,", yytext);}
{op} {printf("\nOperator %s encountered", yytext);}
{id} {printf("\nIdentifier %s encountered", yytext);}
{number} {printf("\nNumber %s encountered", yytext);}
%%
```

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

}
```

OUTPUT

```
C= a++  +  ++b

Identifier  c   encountered
Relational Operator  =  encountered
Identi fier  a   encountered
Operator  ++  encountered
Operator  +   encountered
Operator  ++  encountered
Identifier   b  encantered

 C=  ++a   +  ++b
Identfier  c    encountered
Relational  Operator  = encountered
Operator  ++   encountered
Identifier  a    encountered
Operator  +    encountered
Operator  ++   encountered
Identifier   b   encountered
```



```
srihari@LAPTOP-AJMTFS87:~/compiler_design/week5$ cat q1.l
%{
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
%}

relop [<|>|=]+
letter [a-zA-Z_]
digit [0-9]
id {letter}({letter}|{digit})*
number {digit}+
op [+|\-|*|/]+
str [\"]{letter}+[\"]

%%
"int"|"float"|"char"|"for"|"while"|"do"|"continue"|"break"|"def"|"return"|"if"|"then"|"elif"|"switch" {printf("\nKeyword %s encountered",yytext);}
{relop} {printf("\nRelational Operator %s encountered",yytext);}
{str} {printf("\nString %s encountered",yytext);}
{op} {printf("\nOperator %s encountered",yytext);}
{id} {printf("\nIdentifier %s encountered",yytext);}
{number} {printf("\nNumber %s encountered",yytext);}
%%

int main(){
        yylex();
        return 0;
}
srihari@LAPTOP-AJMTFS87:~/compiler_design/week5$ ./a.out
c = a++ + ++b

Identifier c encountered
Relational Operator = encountered
Identifier a encountered
Operator ++ encountered
Operator + encountered
Operator ++ encountered
Identifier b encountered
c = ++a + ++b

Identifier c encountered
Relational Operator = encountered
Operator ++ encountered
Identifier a encountered
Operator + encountered
Operator ++ encountered
Identifier b encountered
```

2. Identify the tokens in given input.

```
def f(x):
if x>=1:
    return x*x

  else: return x

print 3
```

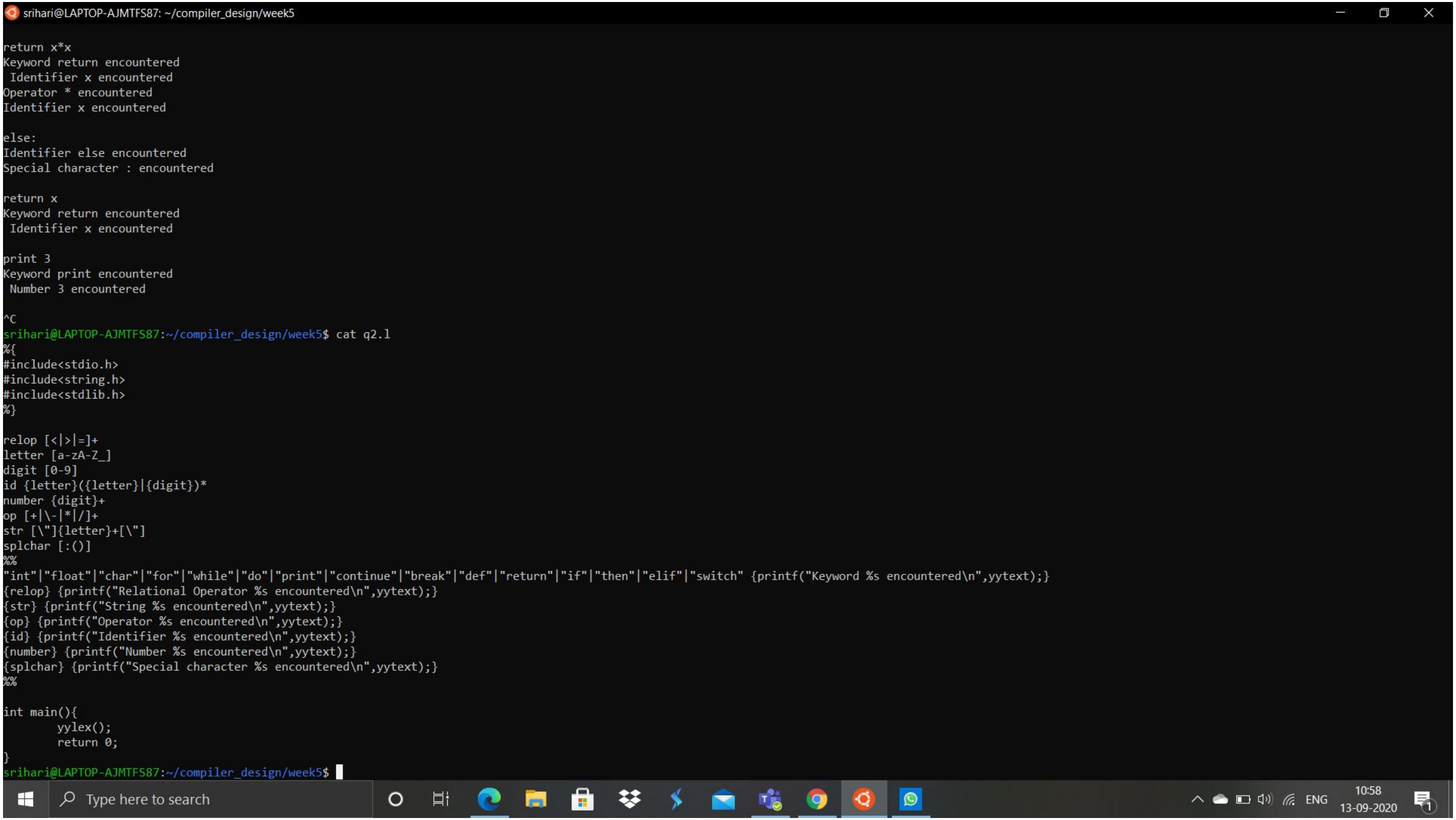
```
%{
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
%}
relop [<|>|=]+
letter [a-zA-Z_]
digit [0-9]
id {letter}({letter}|{digit})*
number {digit}+
op [+|\-|*|/]+
str [\"]{letter}+[\"]
splchar [:()]

%%
"int"|"float"|"char"|"for"|"while"|"do"|"print"|"def"|"return"|
    "if"|"else"|"elif"|"then" {printf("Keyword %s
        encountered \n", yytext);}

{relop} {printf("Relational Operator %s encountered \n",yytext);
{str} { printf ("string %s encountered \n", yytext); }
{op} {printf("Operator %s encountered", yytext); }
```

```
{id} {printf("Identifier %s encountered \n", yytext);}
{number} {printf("Number %s encountered \n", yytext);}
{splchar} { printf("Special Character %s encountered \n",(yytext)

%%

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

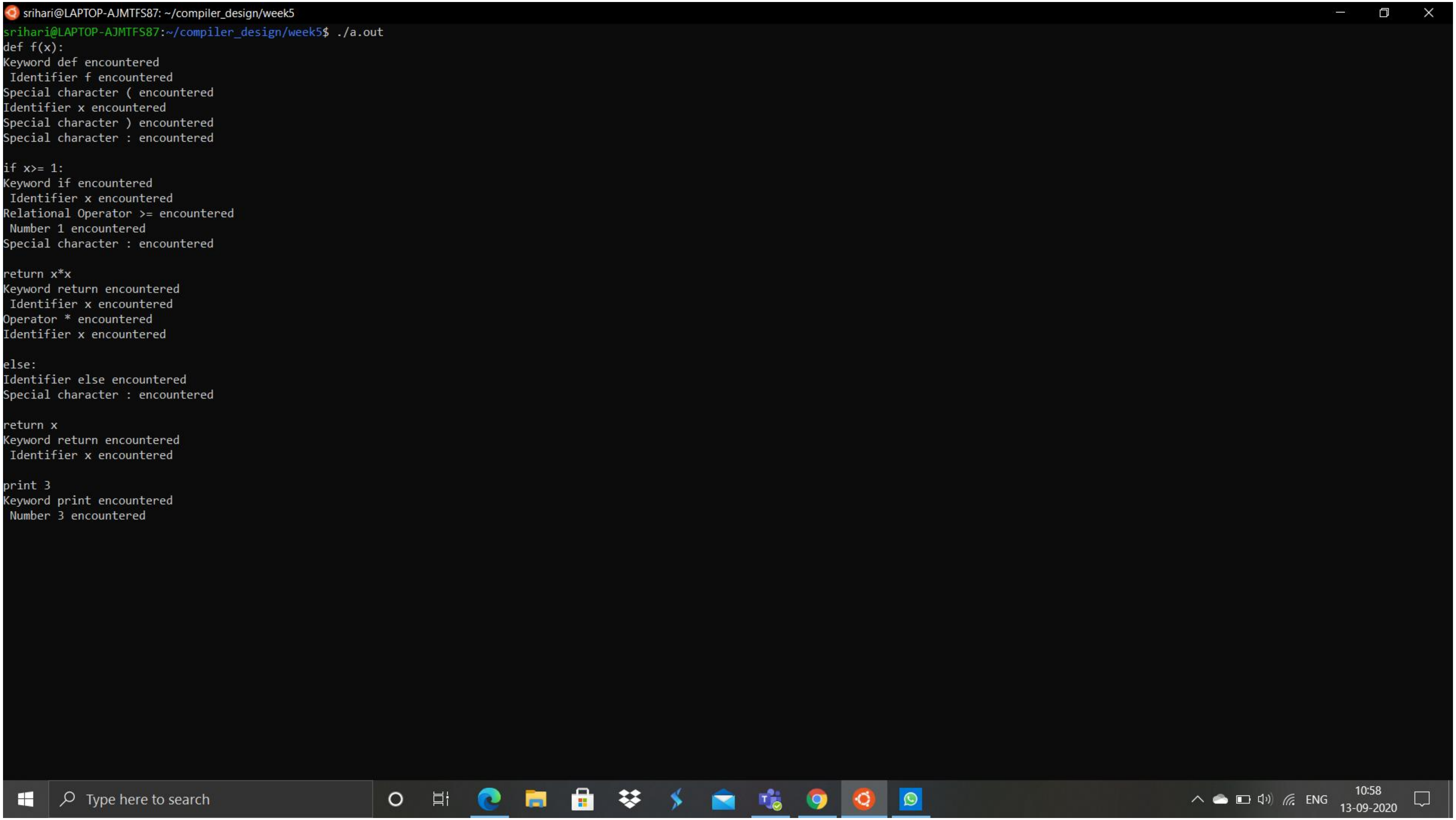
OUTPUT

```
def f(x):
Keyword def encountered
Identifier f encountered
Special character ( encountered
Identifier x encountered
Special character ) encountered
Special Character : encountered

 if x>=1:
Keyword if encountered
Identifier x encountered
Relational Operator >= encountered
Number 1 encountered
Special character : encountered

return x*x.
Keyword return encountered
Identifier x encountered
Operator * encountered
Identifier x encountered
```

```
else:
Identifier else encountered
Special character : encountered

return x
Keyword return encountered
Identifier x encountered

print 3
Keyword print encountered
Number 3 encountered.
```



```
return x*x
Keyword return encountered
 Identifier x encountered
Operator * encountered
Identifier x encountered

else:
Identifier else encountered
Special character : encountered

return x
Keyword return encountered
 Identifier x encountered

print 3
Keyword print encountered
 Number 3 encountered

^C
srihari@LAPTOP-AJMTFS87:~/compiler_design/week5$ cat q2.l
%{
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
%}

relop [<|>|=]+
letter [a-zA-Z_]
digit [0-9]
id {letter}({letter}|{digit})*
number {digit}+
op [+|\-|*|/]+
str [\"]{letter}+[\"]
splchar [:()]
%%
"int"|"float"|"char"|"for"|"while"|"do"|"print"|"continue"|"break"|"def"|"return"|"if"|"then"|"elif"|"switch" {printf("Keyword %s encountered\n",yytext);}
{relop} {printf("Relational Operator %s encountered\n",yytext);}
{str} {printf("String %s encountered\n",yytext);}
{op} {printf("Operator %s encountered\n",yytext);}
{id} {printf("Identifier %s encountered\n",yytext);}
{number} {printf("Number %s encountered\n",yytext);}
{splchar} {printf("Special character %s encountered\n",yytext);}
%%

int main(){
        yylex();
        return 0;
}
srihari@LAPTOP-AJMTFS87:~/compiler_design/week5$ 
```



```
srihari@LAPTOP-AJMTFS87:~/compiler_design/week5$ ./a.out
def f(x):
Keyword def encountered
 Identifier f encountered
Special character ( encountered
Identifier x encountered
Special character ) encountered
Special character : encountered

if x>= 1:
Keyword if encountered
 Identifier x encountered
Relational Operator >= encountered
 Number 1 encountered
Special character : encountered

return x*x
Keyword return encountered
 Identifier x encountered
Operator * encountered
Identifier x encountered

else:
Identifier else encountered
Special character : encountered

return x
Keyword return encountered
 Identifier x encountered

print 3
Keyword print encountered
 Number 3 encountered
```