program.l

```
%{
#include<stdio.h>
int a,b;
%}
//Patterns for C Tokens
letter [A-Za-z]
digit [0-9]
keyword if|else|do|while|int
identifier {letter}({letter}|{digit})*
relational_op "<"|">"|"<="|">="|"<>"
arithmetic_op "+"|"-"|"*"|"/"
single_line_comment "//".*
multi_line_comment "/*".*"*/"
%%
{keyword} {printf("Keyword %s is found",yytext);}
{identifier} {printf("Identifier %s is found",yytext);}
{letter} {printf("Letter %s is found",yytext);}
{digit} {printf("Digit %s is found",yytext);}
{relational_op} {printf("Relational operator %s is found",yytext);}
{arithmetic_op} {printf("Arithmetic operator %s is found",yytext);}
{single_line_comment} {printf("Single line comment is found");}
{multi_line_comment} {printf("Multi line comment is found");}
%%
int main()
{
```

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

```
// code.c
#include <stdio.h>
int main() {
int num1, num2, sum;
printf("Enter two integers: ");
scanf("%d %d", &num1, &num2);
// calculating sum
sum = num1 + num2;
printf("%d + %d = %d", num1, num2, sum);
return 0;
}
```

```
/*
Execution Steps:
1. Compile lex program
lex file.l ----→ lex.yy.c
2. Compile c program lex.yy.c
cc lex.yy.c ---→ a.out
3. Execute object program
```

./a.out

*/

student@student-Veriton-M200-H61:~$ lex rn12.l

student@student-Veriton-M200-H61:~$ cc lex.yy.c or // gcc lex.yy.c

student@student-Veriton-M200-H61:~$ ./a.out <rn12.c