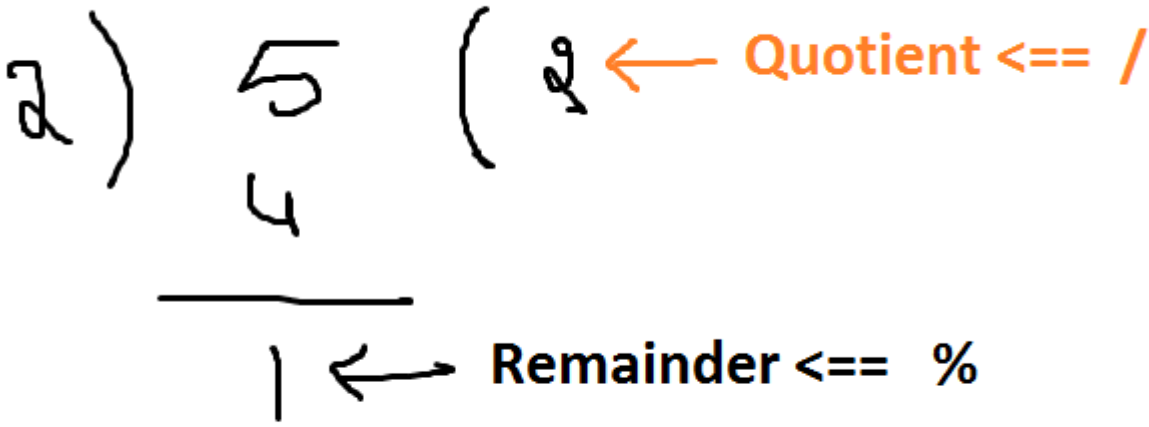
ARITHMETIC OPERATORS [+, -, *, %, /]

They are used to perform mathematical operations.

Eg: a+b, a-b, a*b,.....

% - modulo division [modules / remainder]:

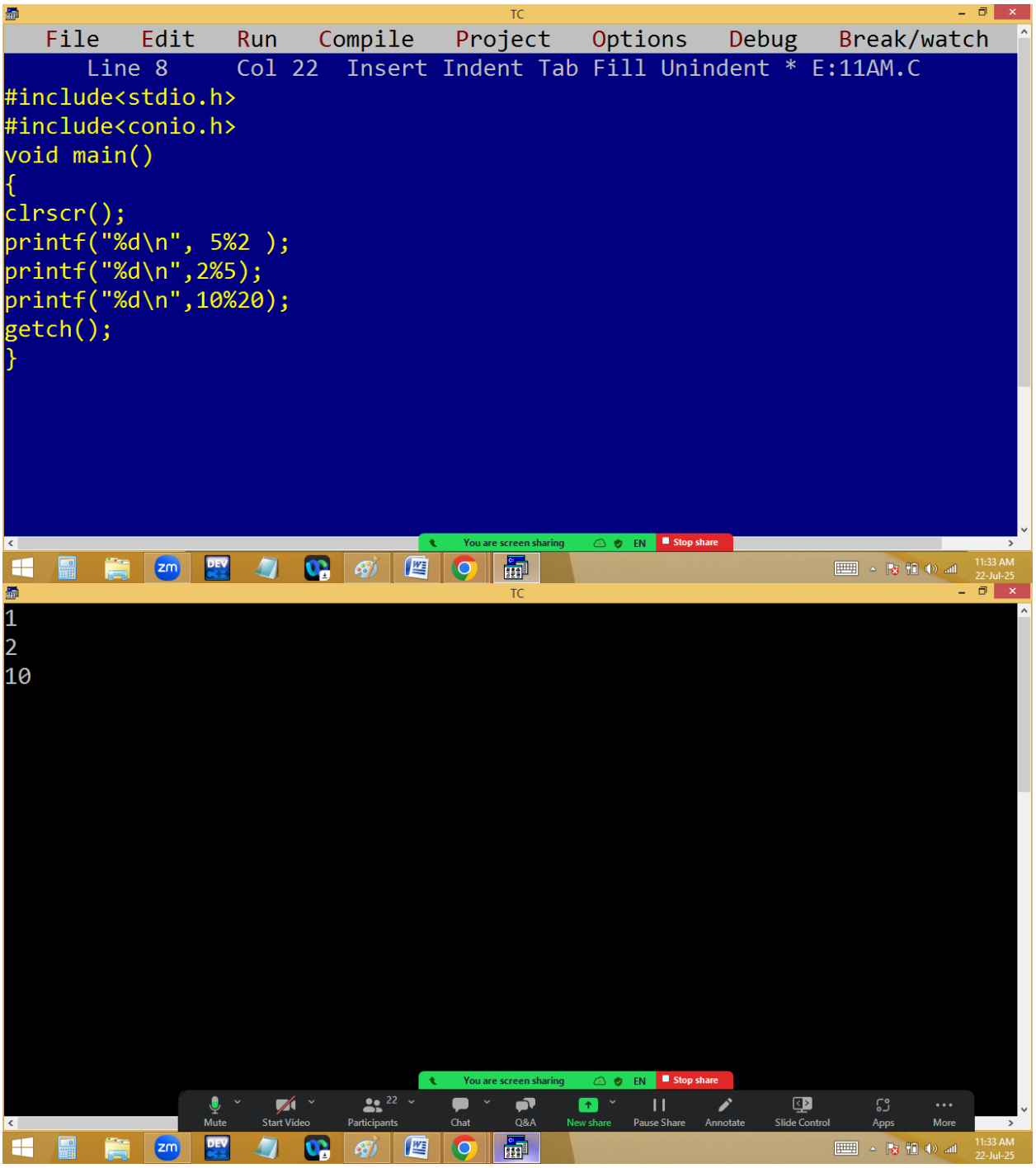
5 % 2 = 1



2 % 5 = 2

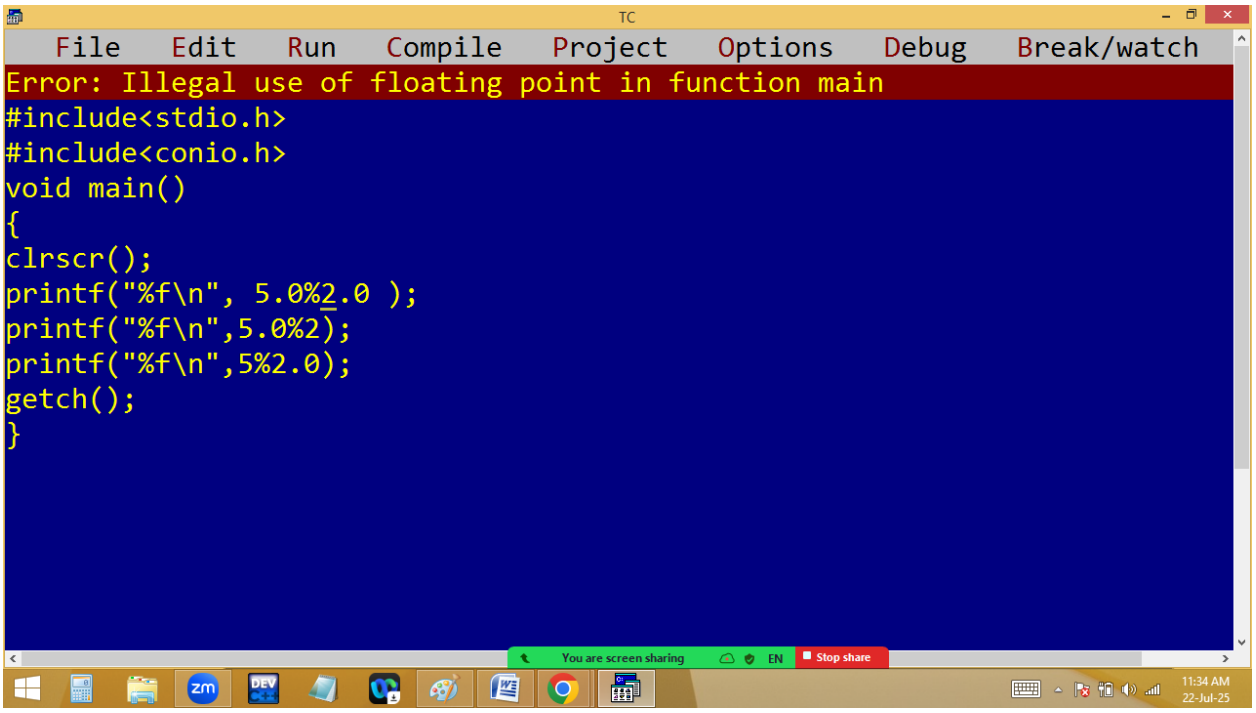
Note: if the divisor bigger than dividend then dividend is the answer.

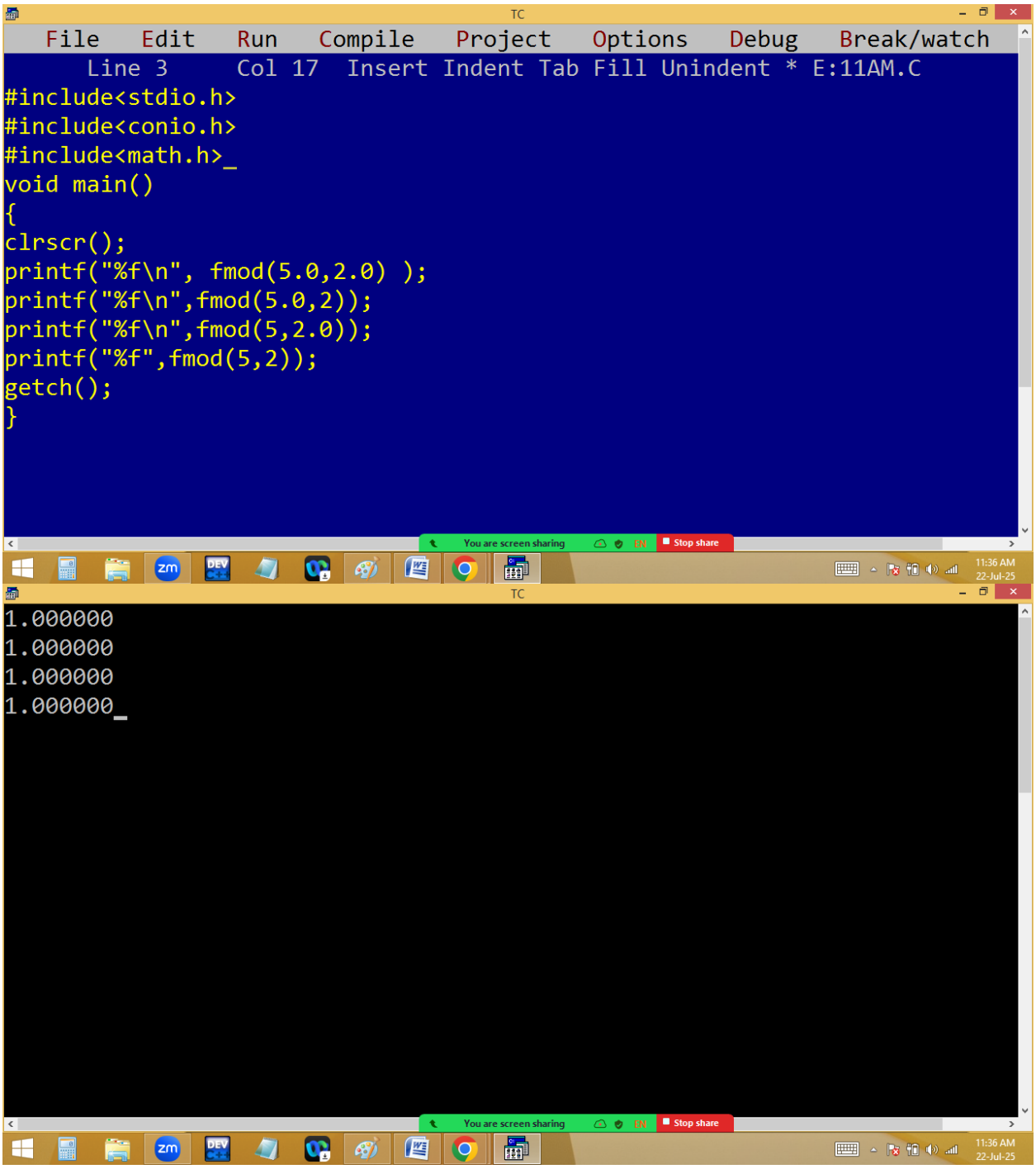
dividend divisor



5.0 % 2.0 = Error

Note: In C & C++ % operator can't perform floating modules. For this we have to use the predefined function fmod() available in <math.h>


TC
File Edit Run Compile Project Options Debug Break/watch
Error: Illegal use of floating point in function main
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
printf("%f\n", 5.0%2.0);
printf("%f\n",5.0%2);
printf("%f\n",5%2.0);
getch();
}
You are screen sharing
EN
Stop share
11:34 AM
22-Jul-25

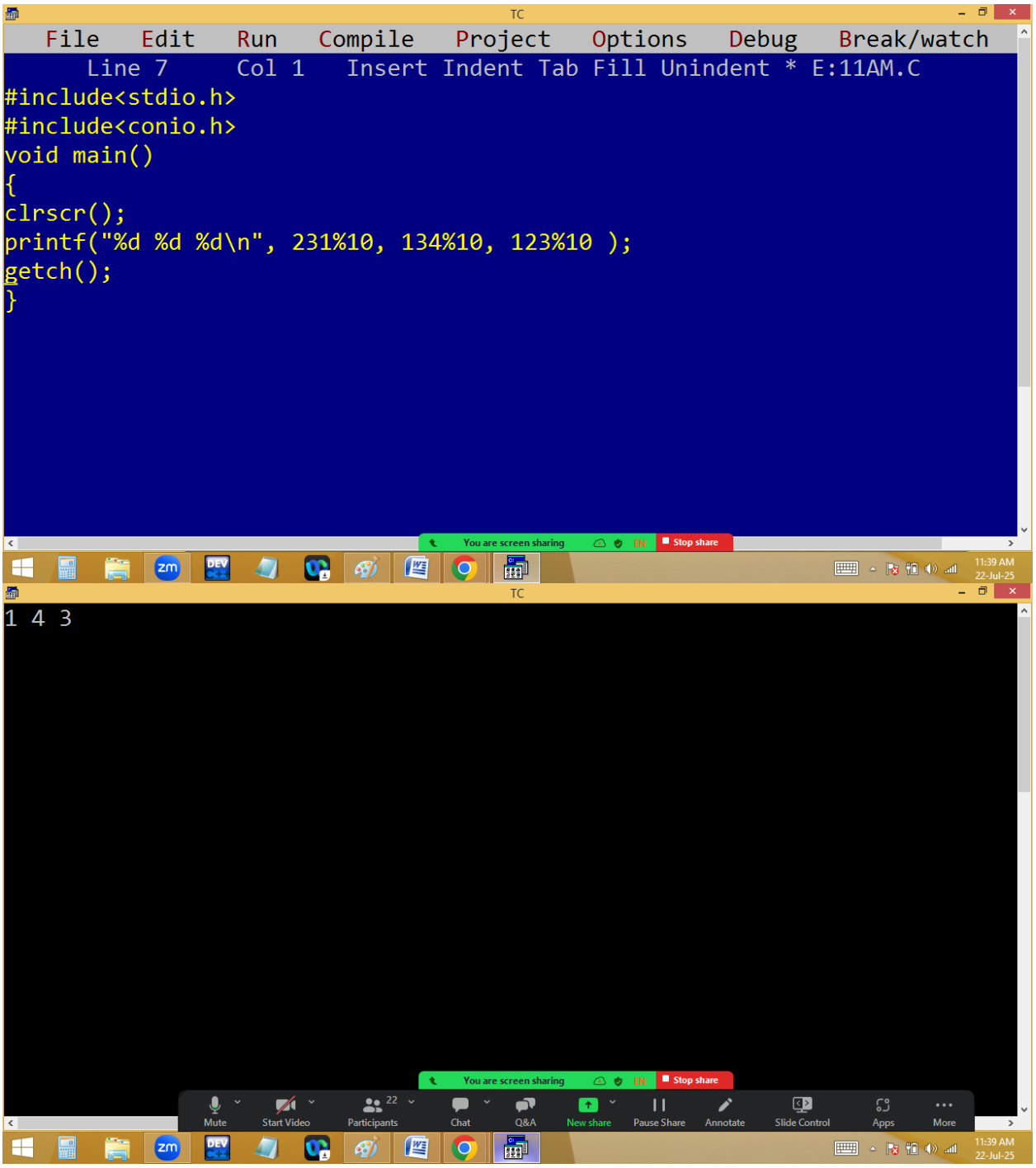

TC
File Edit Run Compile Project Options Debug Break/watch
Line 3 Col 17 Insert Indent Tab Fill Unindent * E:11AM.C
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
clrscr();
printf("%f\n", fmod(5.0,2.0));
printf("%f\n",fmod(5.0,2));
printf("%f\n",fmod(5,2.0));
printf("%f",fmod(5,2));
getch();
}
You are screen sharing
Stop share
11:36 AM
22-Jul-25
TC
1.000000
1.000000
1.000000
1.000000
You are screen sharing
Stop share
11:36 AM
22-Jul-25

231%10=1

124%10=4

123%10=3

Note: Any no%10 gives last digit.

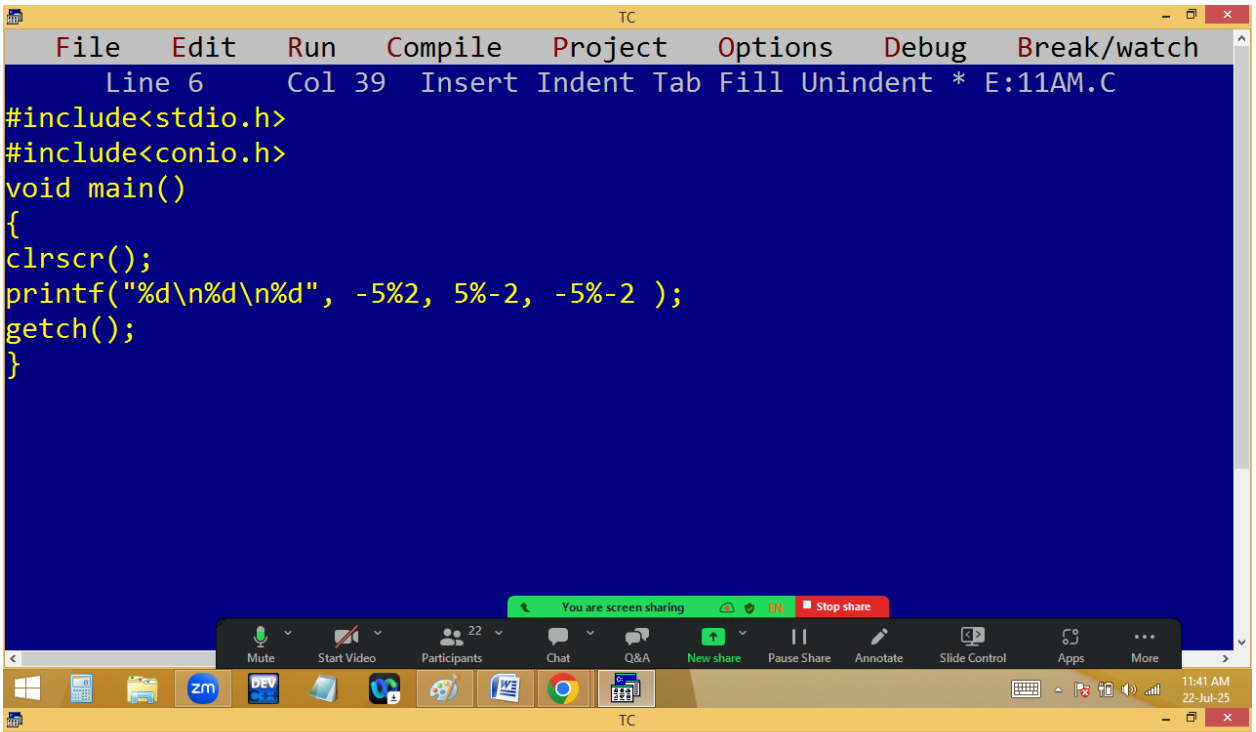

TC
File Edit Run Compile Project Options Debug Break/watch
Line 7 Col 1 Insert Indent Tab Fill Unindent * E:11AM.C
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
printf("%d %d %d\n", 231%10, 134%10, 123%10);
getch();
}
You are screen sharing
Stop share
11:39 AM
22-Jul-25
TC
1 4 3
You are screen sharing
Stop share
Mute
Start Video
Participants
Chat
Q&A
New share
Pause Share
Annotate
Slide Control
Apps
More
11:39 AM
22-Jul-25

-5%2= -1

5%-2= 1

-5%-2= -1

Note: In modules if the numerator is negative then result also negative.



/ - division [Quotient]:

5/2=2

Note: In division both operands are int then result also int. any one or both floats then result also float.

5.0/2=2.500000

5/2.0=2.500000

5.0/2.0=2.500000

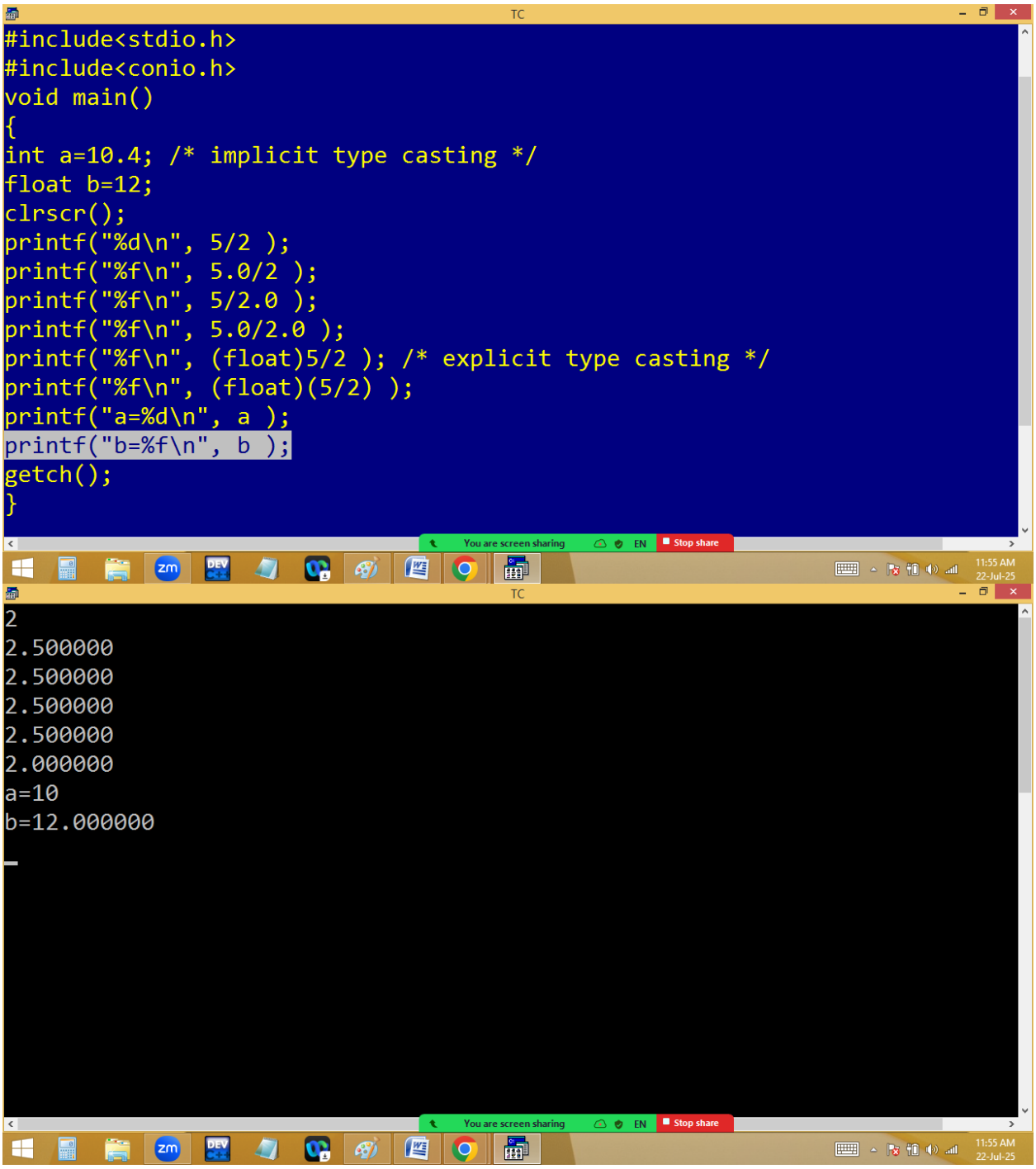
Int a = 5~~.2~~ ; ➔ a=5 ← implicit type casting

float b = 12; ➔ b=12.000000

(int) 5.0/2 = 2 /* explicit type casting */

(float)5/2=2.500000 /* explicit type casting */

(float) (5/2) = 2.000000

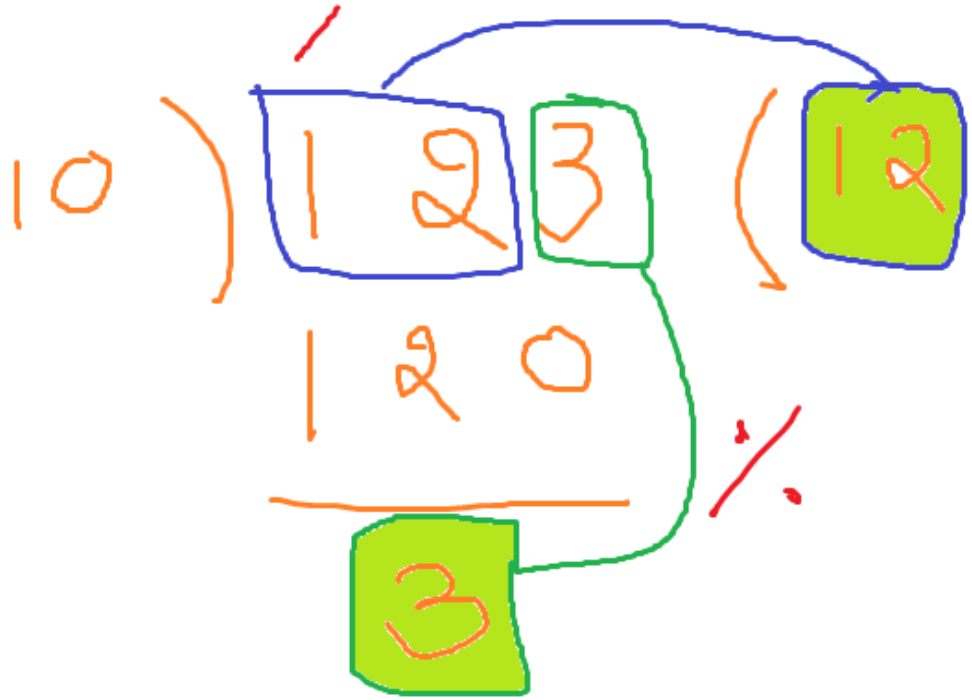


123/10=12

12/10=1

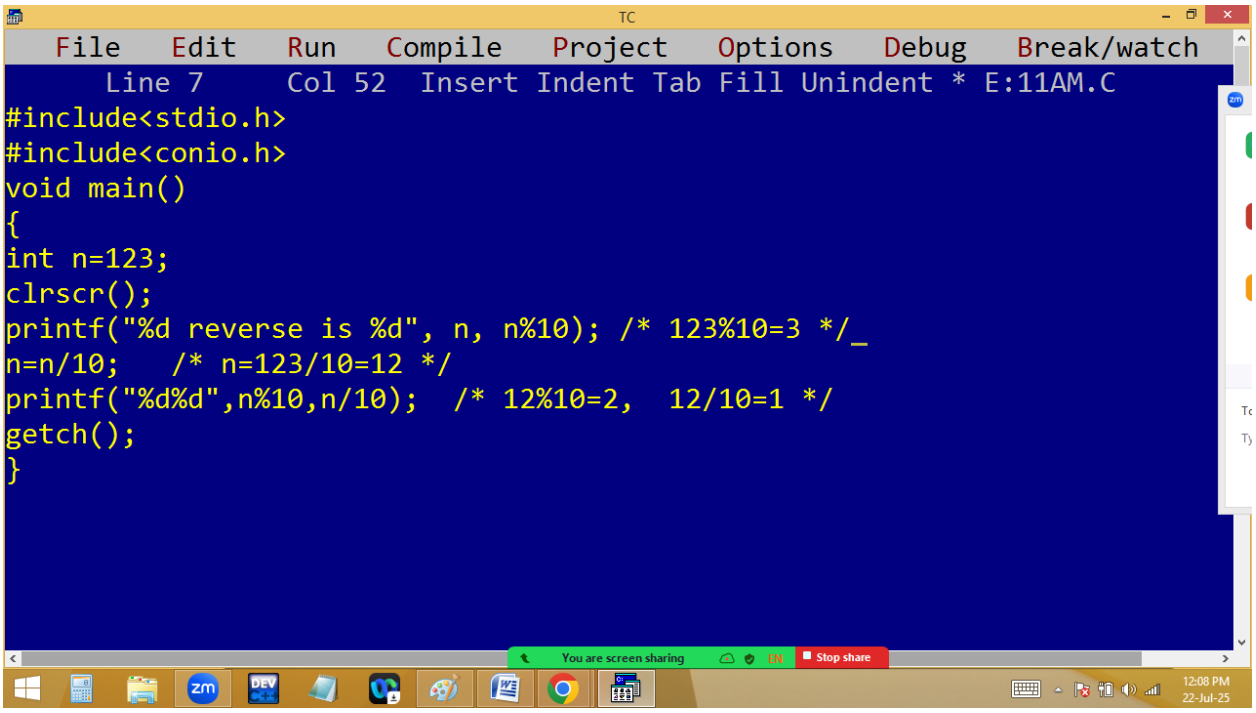
1/10=0

Note: **Any no/10 removes the last digit.**



Write a C program to print a 3 digit no in reverse or without using loops.

Eg: 123 reverse is 321


TC
File Edit Run Compile Project Options Debug Break/watch
Line 7 Col 52 Insert Indent Tab Fill Unindent * E:11AM.C
#include<stdio.h>
#include<conio.h>
void main()
{
int n=123;
clrscr();
printf("%d reverse is %d", n, n%10); /* 123%10=3 */
n=n/10; /* n=123/10=12 */
printf("%d%d",n%10,n/10); /* 12%10=2, 12/10=1 */
getch();
}
You are screen sharing
Stop share
12:08 PM
22-Jul-25

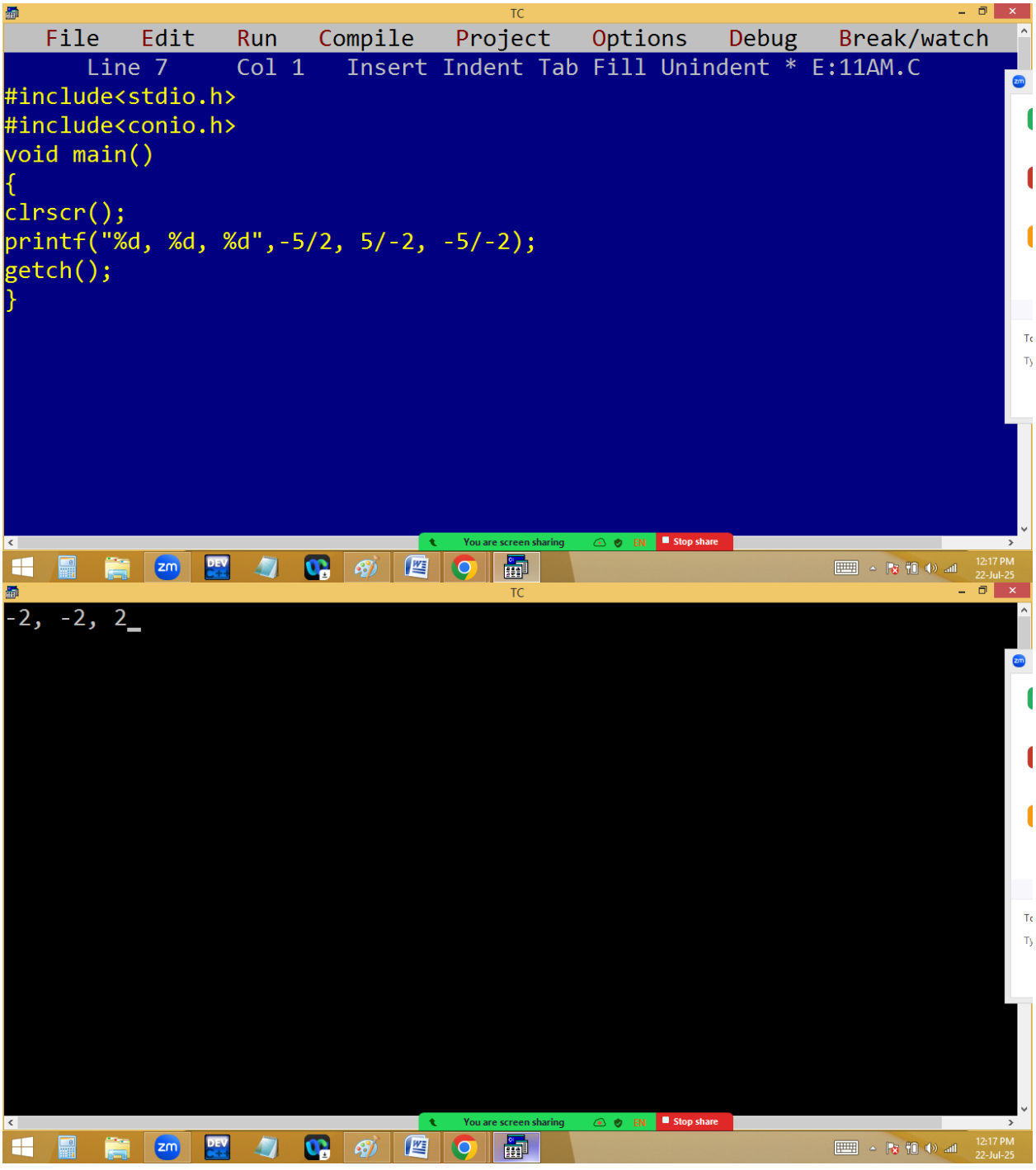

TC
123 reverse is 321
You are screen sharing
Stop share
Mute
Start Video
Participants
Chat
Q&A
New share
Pause Share
Annotate
Slide Control
Apps
More
12:08 PM
22-Jul-25

-5/2= -2

5/-2= -2

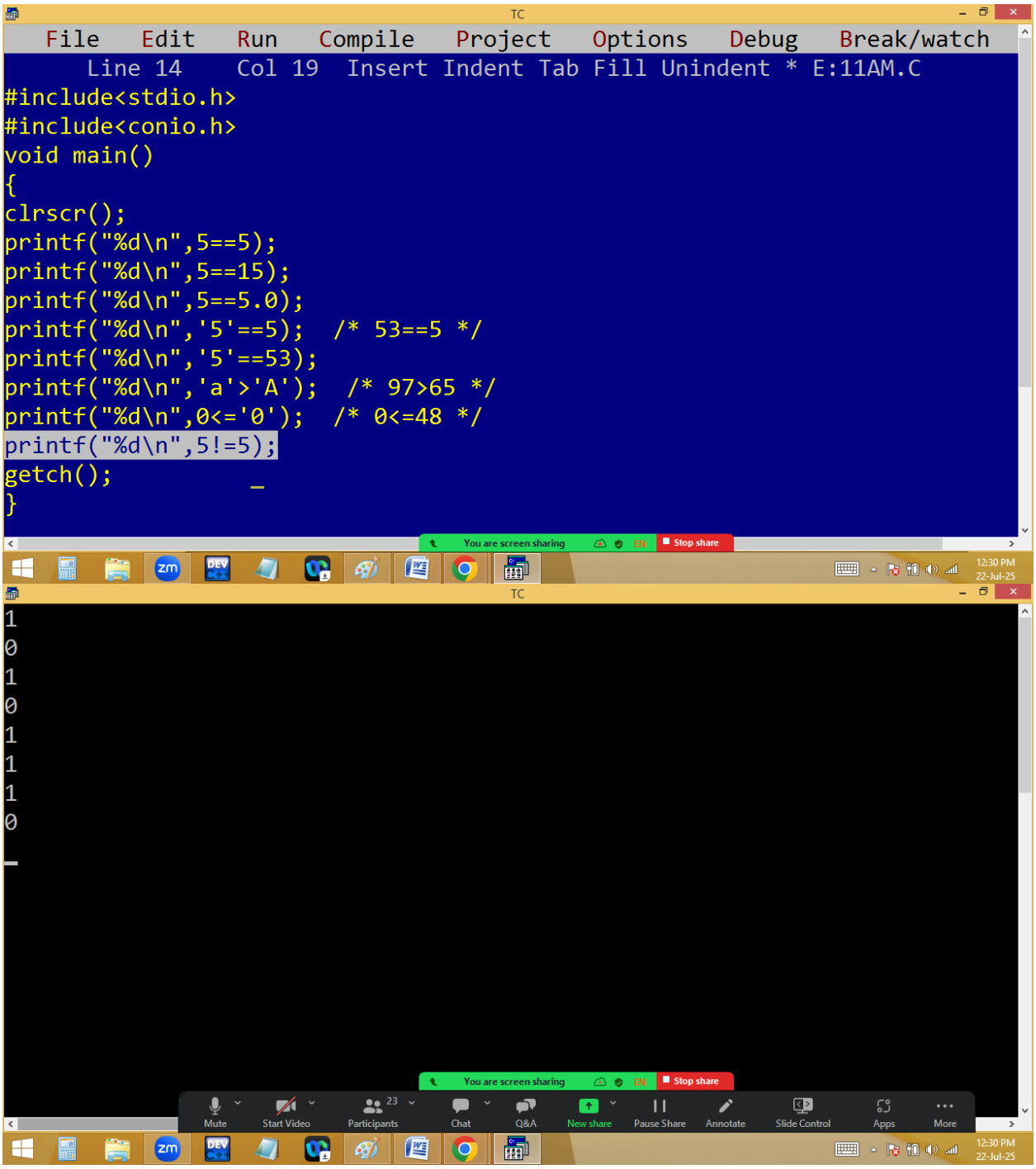
-5/-2= 2

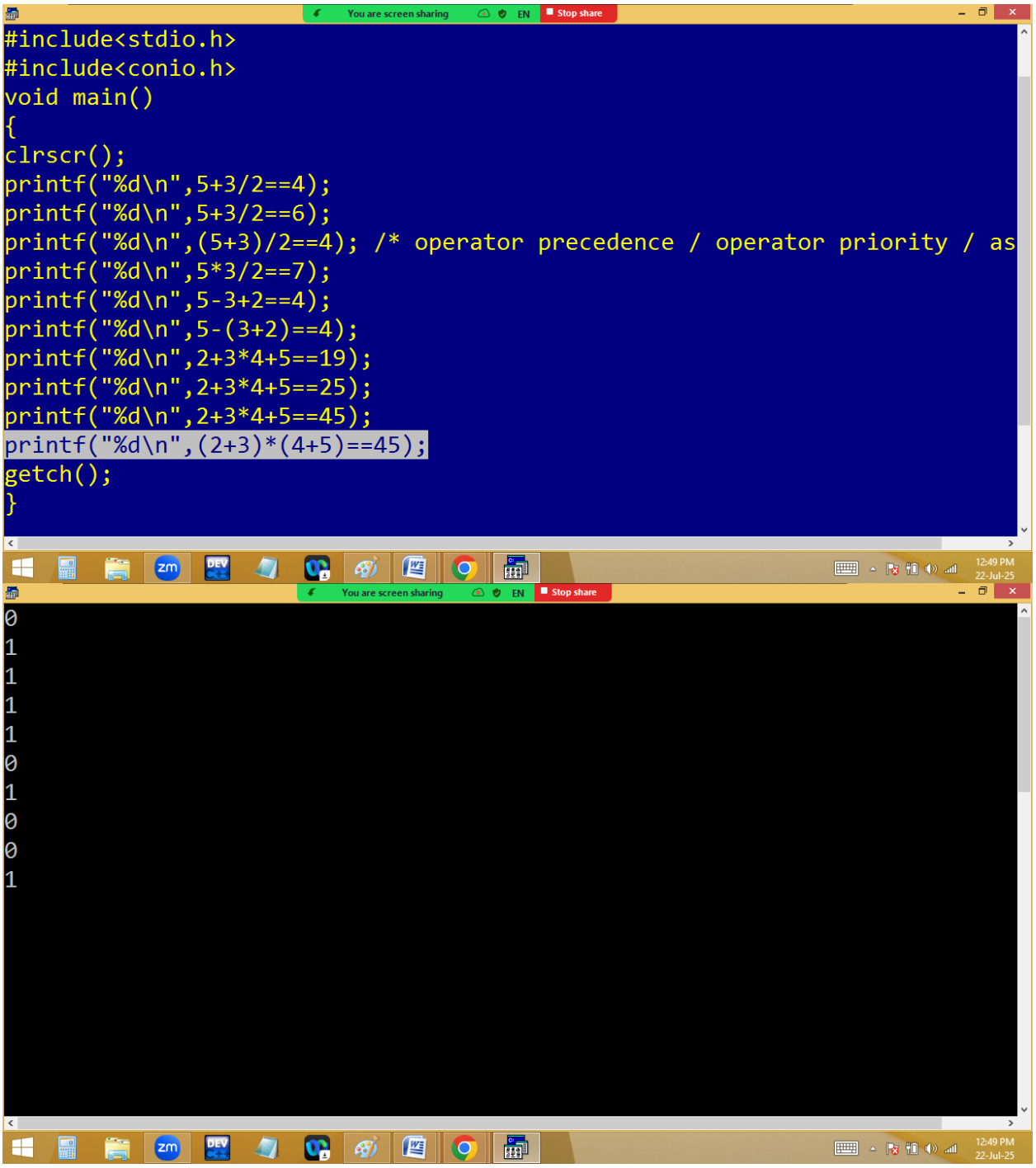
Note: In division any one operand is negative then result also negative. If both are negative then result is positive.



Relational operators [== (comparison), <, >, <=, >=, != (not equal)]:

They are used to check the given expression is true or false. If the expression is true always the answer is 1. If expressions false the answer is 0.


TC
File Edit Run Compile Project Options Debug Break/watch
Line 14 Col 19 Insert Indent Tab Fill Unindent * E:11AM.C
#include<stdio.h>
#include<conio.h>
void main()
{
clrscr();
printf("%d\n",5==5);
printf("%d\n",5==15);
printf("%d\n",5==5.0);
printf("%d\n",'5'==5); /* 53==5 */
printf("%d\n",'5'==53);
printf("%d\n",'a'>'A'); /* 97>65 */
printf("%d\n",0<='0'); /* 0<=48 */
printf("%d\n",5!=5);
getch();
}
You are screen sharing
Stop share
12:30 PM
22-Jul-25
TC
1
0
1
0
1
1
1
0
You are screen sharing
Stop share
Mute
Start Video
Participants
Chat
Q&A
New share
Pause Share
Annotate
Slide Control
Apps
More
12:30 PM
22-Jul-25



Operator precedence / Operator priority

(ASSOCIATION OF OPERATORS)

1. ( )

2. +, - , ! (sign operators, unary operators)
3. ++, -- (pre increment & decrement)
4. *, / , %
5. +, - (Binary)
6. = =, !=
7. &&
8. ||
9. **? :** (ternary operator)
10. =
11. ++, -- (Post increment & decrement)
12. , (comma)