SIT – 232 OBJECT ORIENTED DEVELOPMENT

TASK 1.2 P

QUESTION 4:

Find existing issues for all the code snippets presented below. It is highly important to try working this out on paper by tracing through the statements and then input the code into your IDE. Run it to see if you are correct and then see if you can fix the code. You may wish to add additional Console.WriteLine statements to check that variables contain the values you expect. Note that there may be more than one mistake.

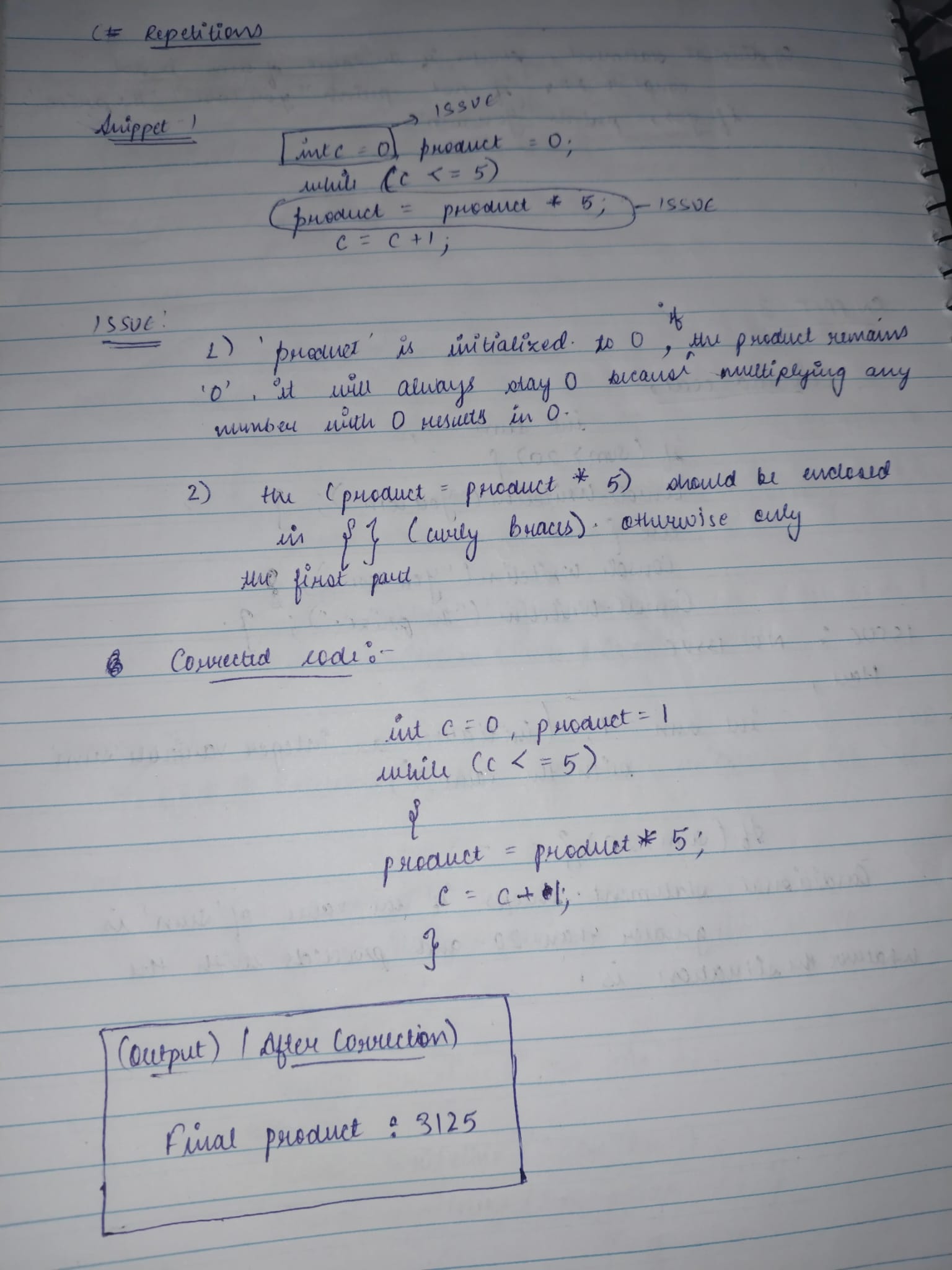
SNIPPET 1:

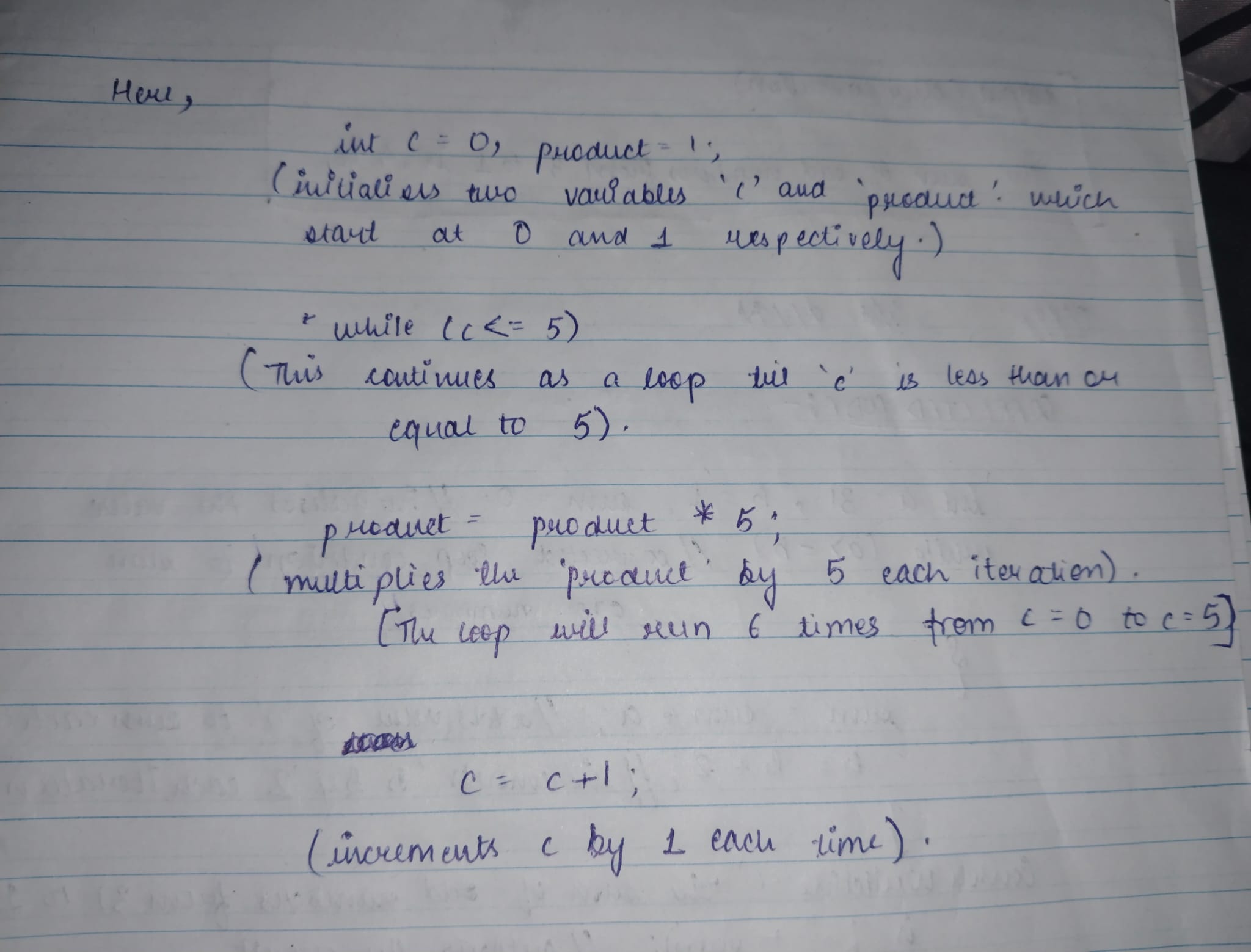
int c = 0, product = 0;

while (c <= 5)

product = product \* 5;

c = c + 1;





ISSUE:

1. **Initialization of product**:
   * product is initialized to 0. In a multiplication operation inside the loop (product = product \* 5;), if product remains 0, it will always stay 0 because multiplying any number by 0 results in 0. This suggests that the intent might have been to start with 1 instead of 0.
2. **Loop Structure**:
   * The loop body (product = product \* 5;) is not correctly enclosed in braces { }. In C#, if braces are omitted in a loop or conditional statement, only the immediate next statement is considered part of the loop. In this case, only product = product \* 5; is part of the loop, and c = c + 1; is executed after the loop completes. This results in an infinite loop because c is never incremented within the loop.

CORRECTED CODE:

int c = 0, product = 1;

while (c <= 5)

{

product = product \* 5;

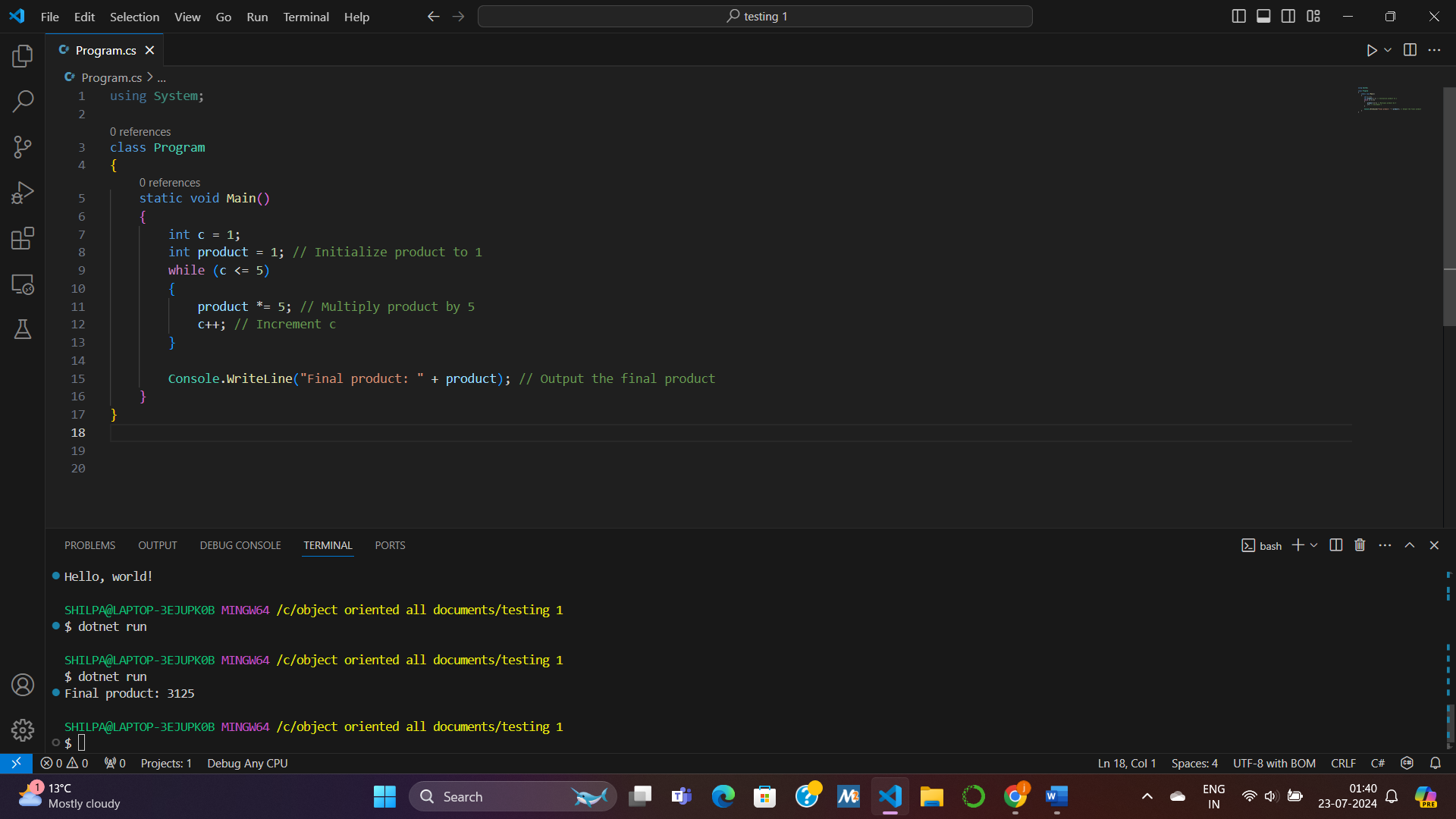
c = c + 1;

}

Console.writeline(“final product:” +product);

}

}



SNIPPET 2:

int a = 31, b = 0, sum = 0;

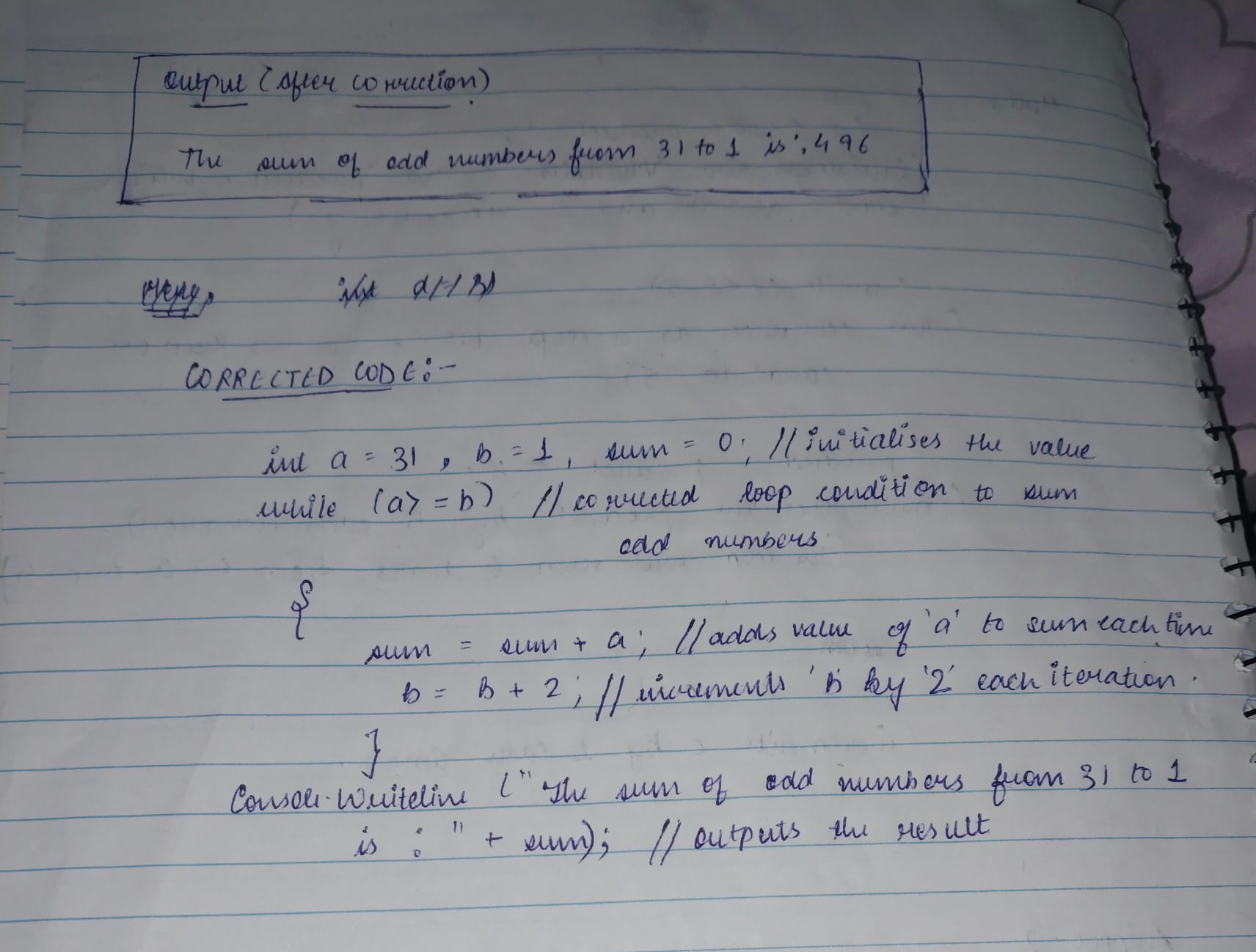
while (a != b) {

sum = sum + a;

b = b + 2;

}





ISSUE:

The loop condition while (a != b) suggests that the loop will continue executing until a becomes equal to b. However, b is incremented by 2 in each iteration (b = b + 2;), and a remains constant at 31. Therefore, a will never equal b, leading to an infinite loop.

Given that a is never modified within the loop and b is incremented by 2 each time, the loop will repeatedly execute sum = sum + 31;, which will lead to sum increasing indefinitely.

CORRECTED CODE:

int a = 31, b = 1, sum = 0;

while (a >= b) // Corrected loop condition to sum odd numbers

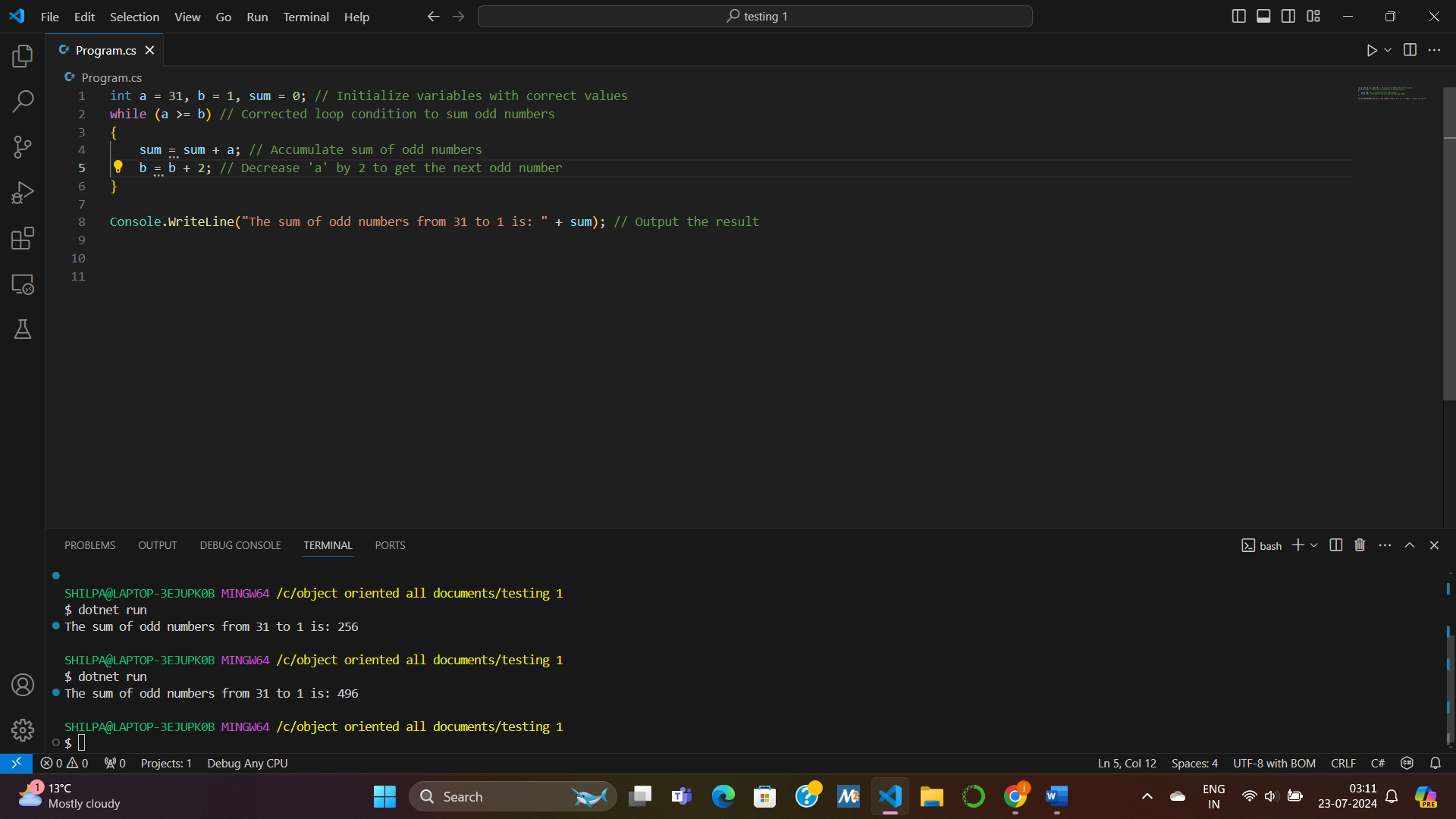
{

sum = sum + a;

b = b + 2;

}

Console.WriteLine("The sum of odd numbers from 31 to 1 is: " + sum); // Output the result



SNIPPET 3:

int x = 1;

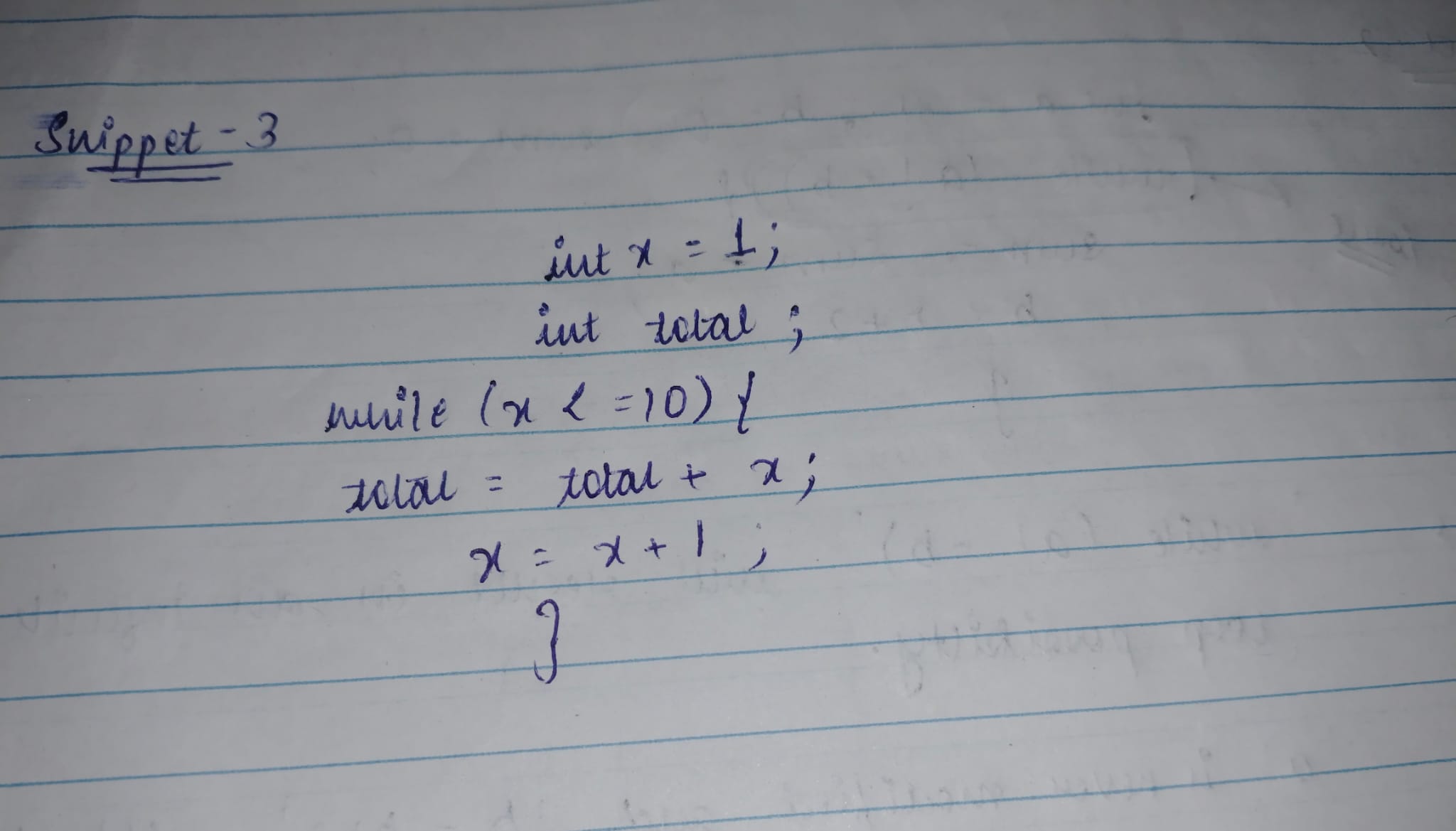
int total;

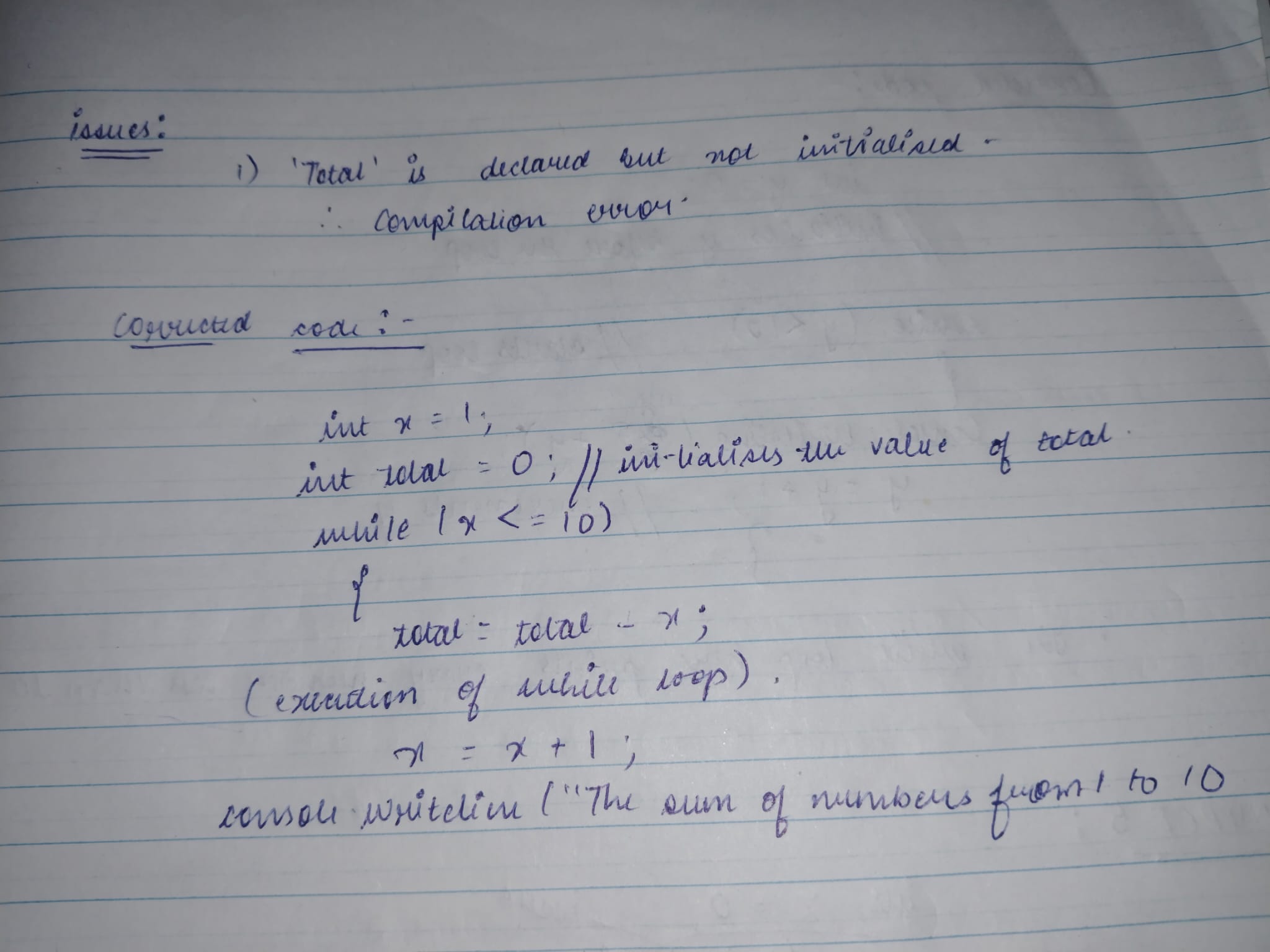
while (x <= 10) {

total = total + x;

x = x + 1;

}

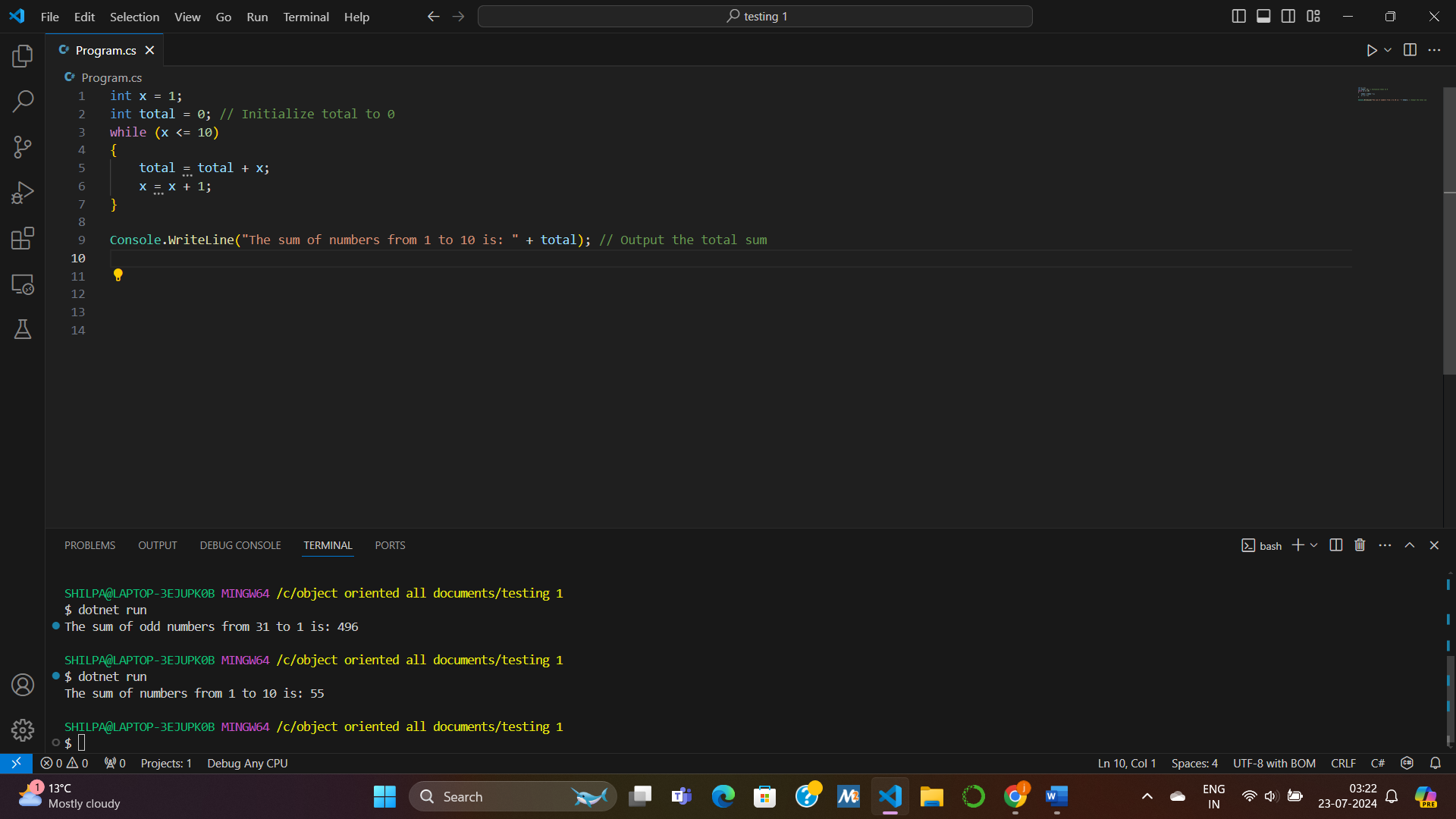




ISSUE:

total is declared but not initialized. This will lead to a compilation error because in C#, variables must be initialized before they can be used.

CORRECTED CODE:



SNIPPET 4:

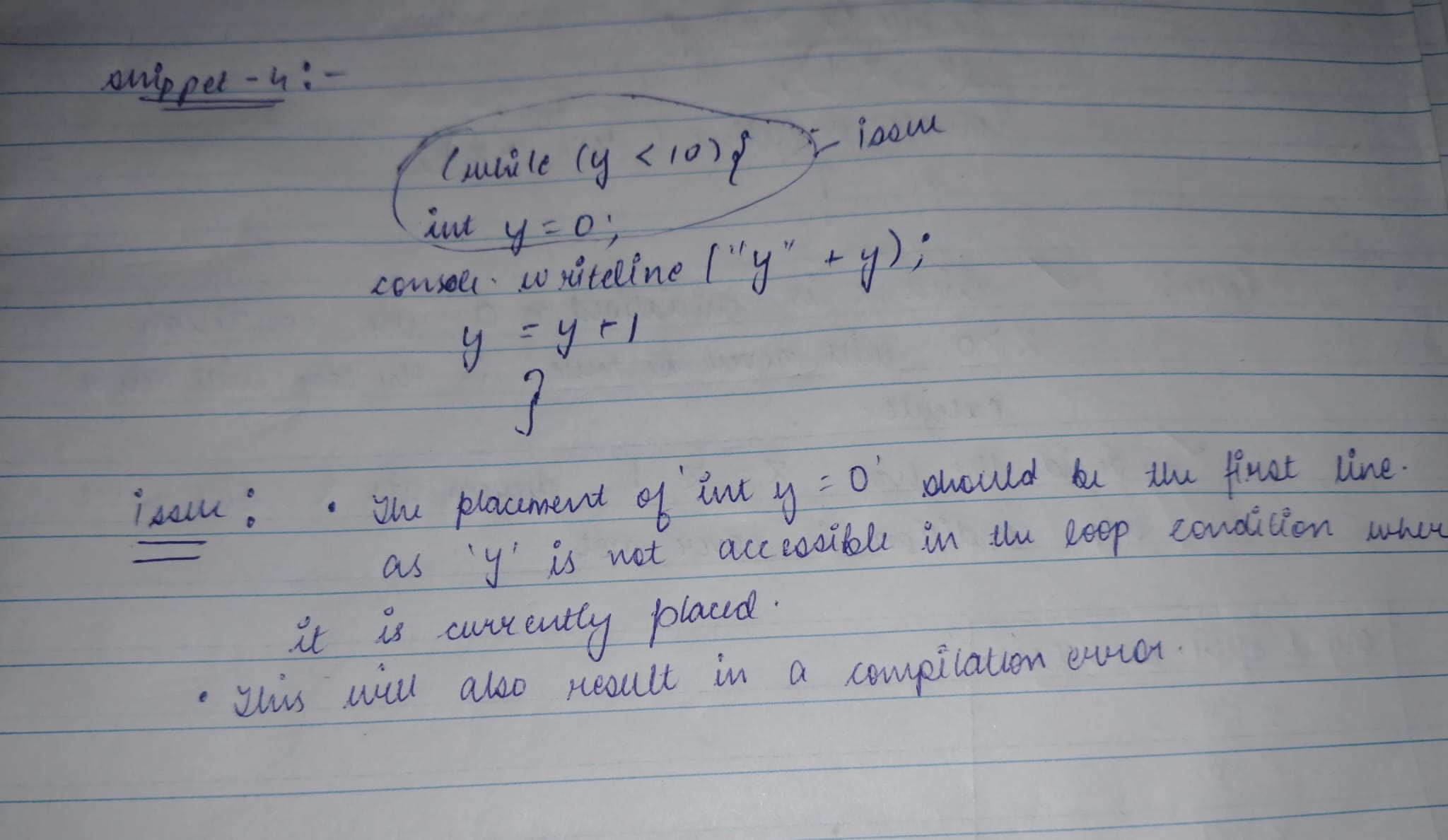
while (y < 10) {

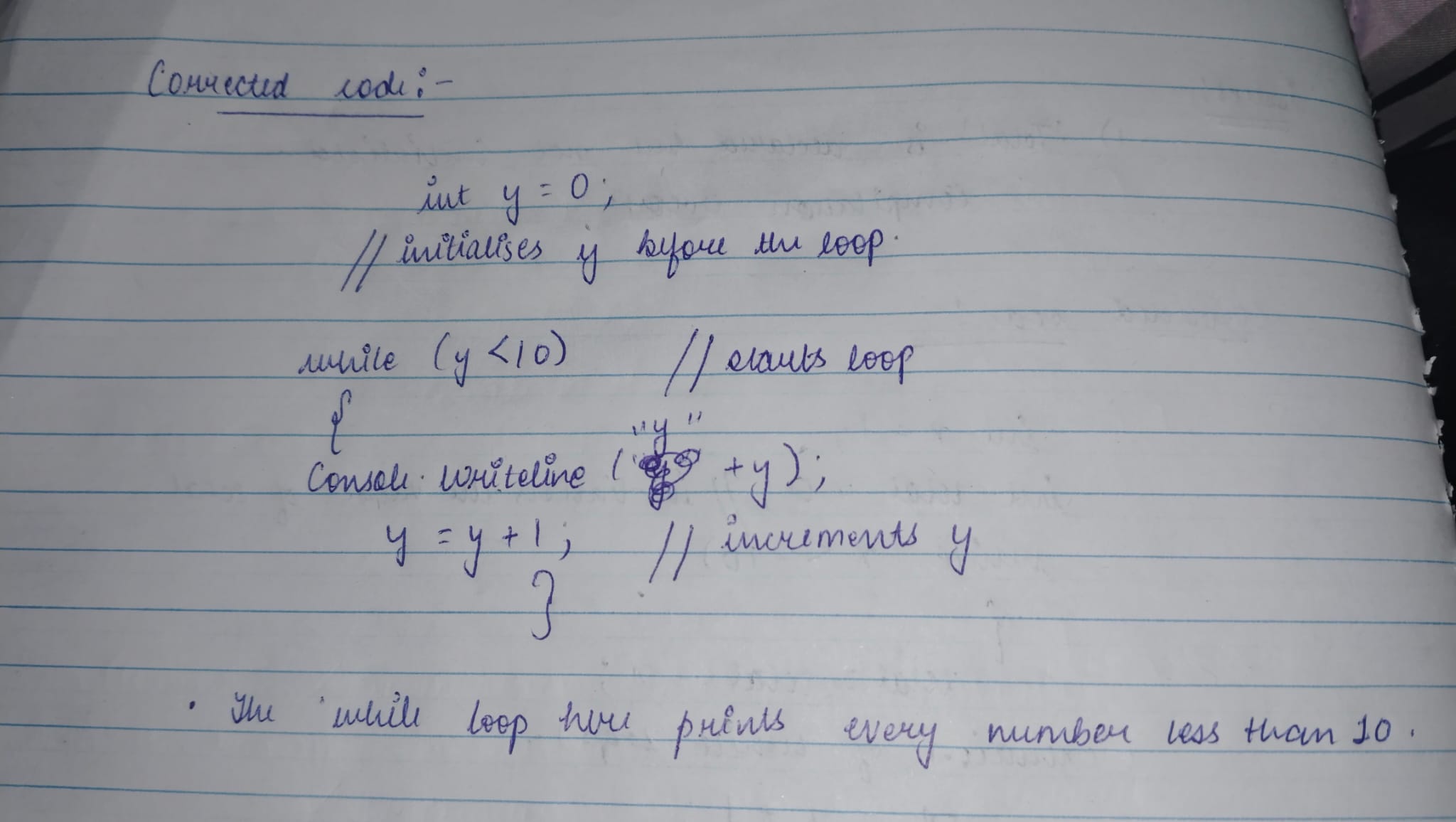
int y = 0;

Console.WriteLine(“y” + y);

y = y + 1;

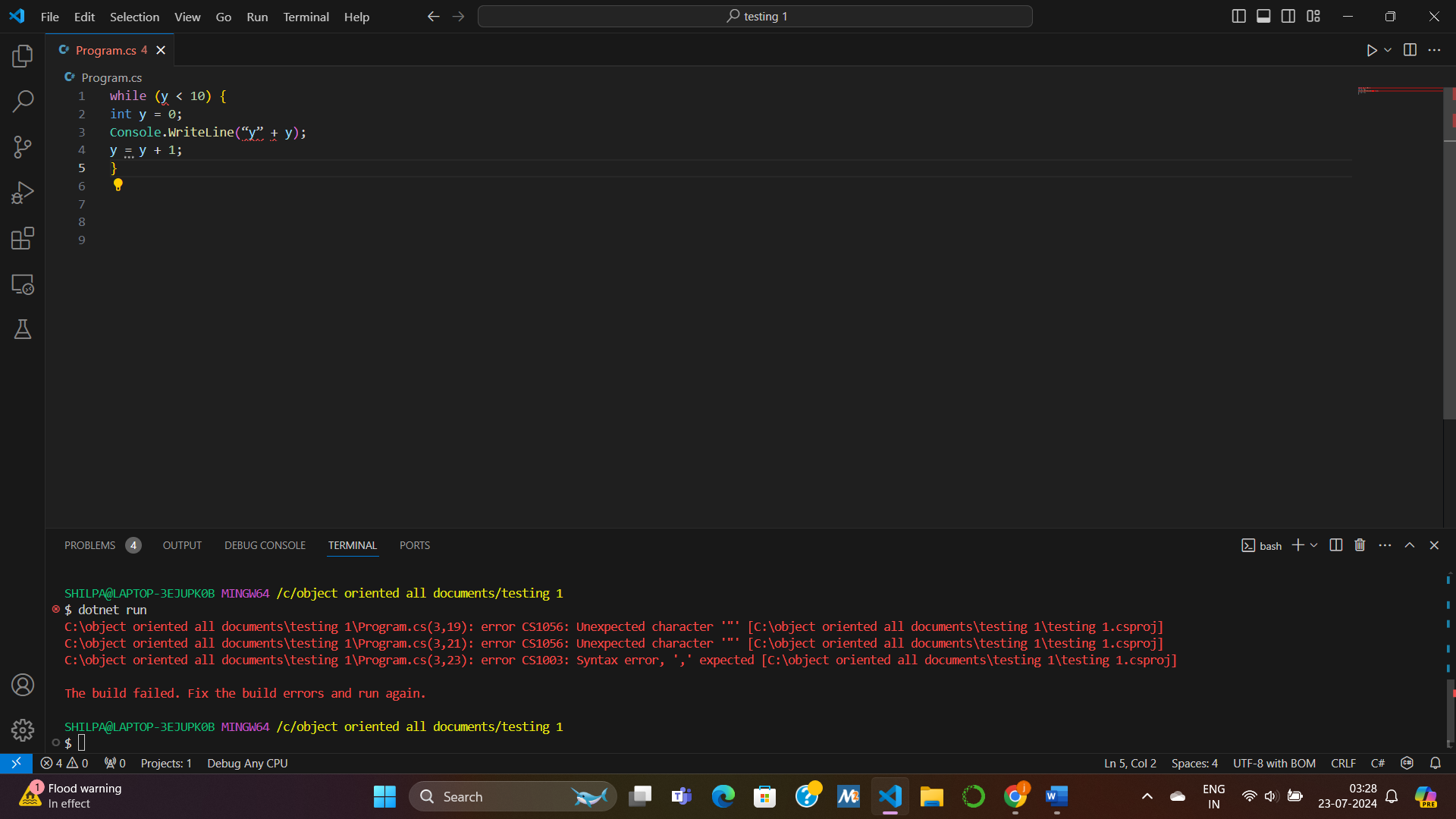
}



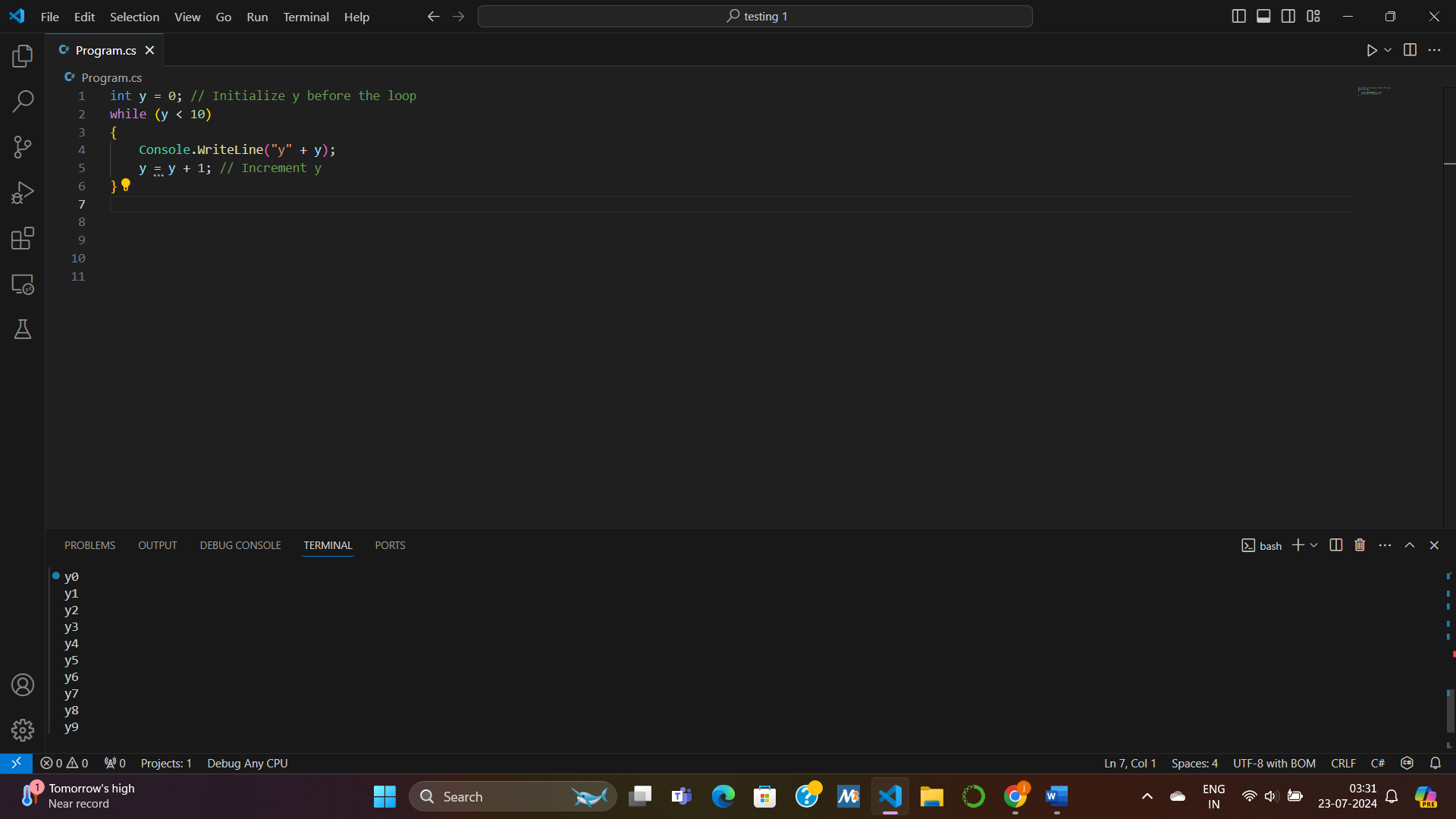


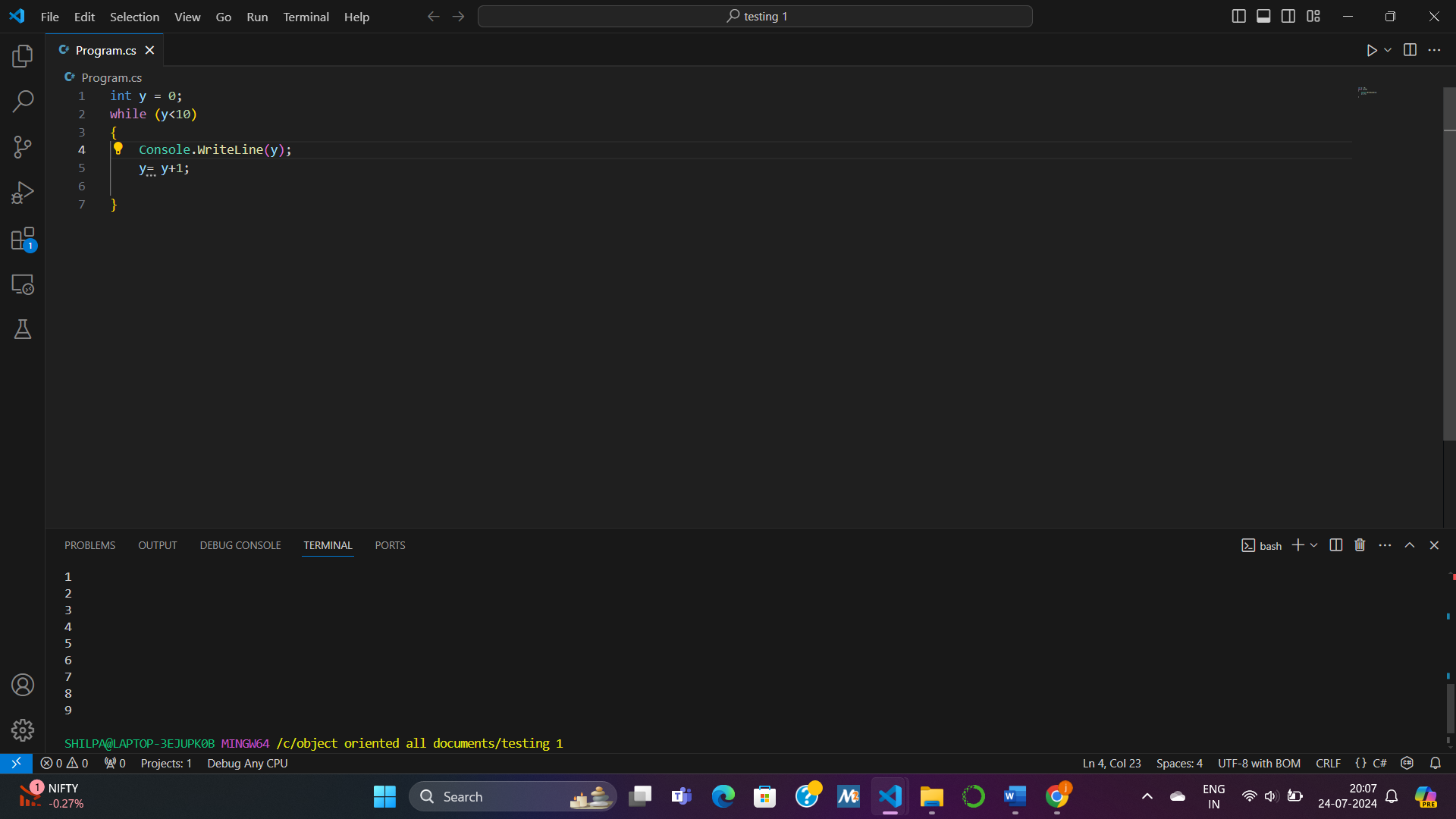
ISSUE:

* 1. The placement of int y = o should be before the while statement as ‘y’ is not accessible in the loop condition where It is currently placed.
  2. the Console.WriteLine("y" + y); statement should output the value of y, but since y is declared and initialized inside the loop block after the loop condition, it will cause a compilation error.



CORRECTED CODE:



SNIPPET 5:

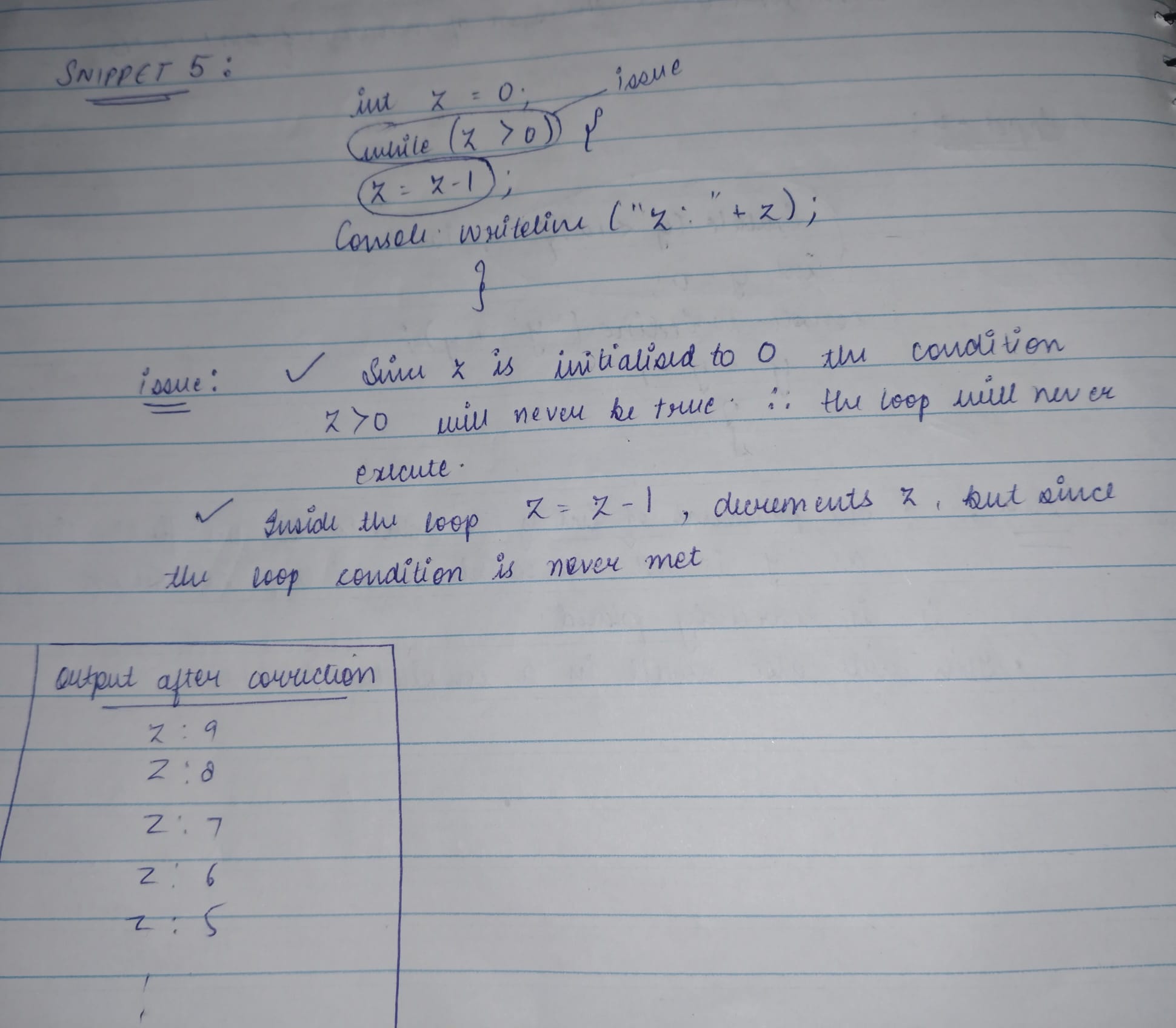
int z = 0;

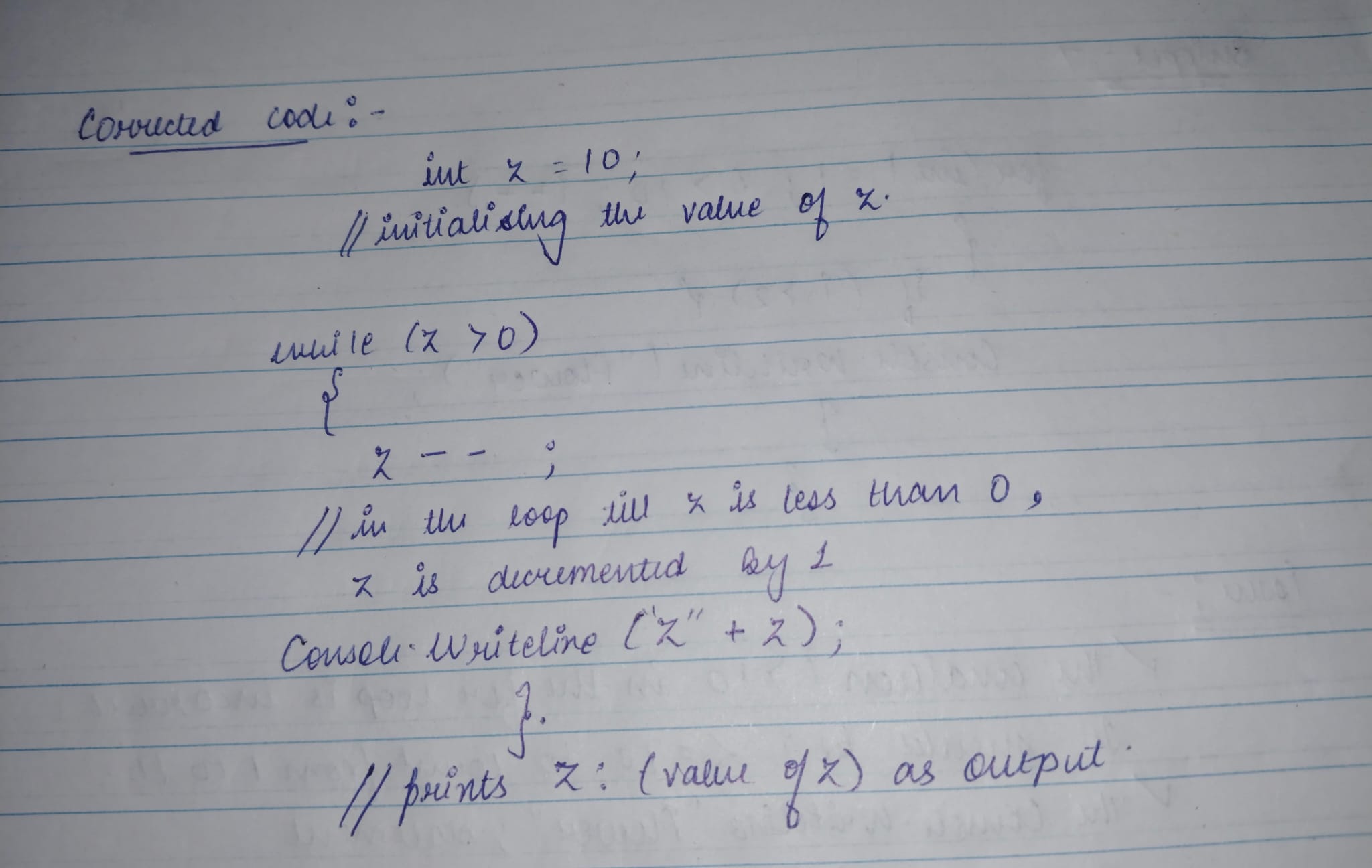
while (z > 0) {

z = z ‐ 1;

Console.WriteLine("z: " +z);

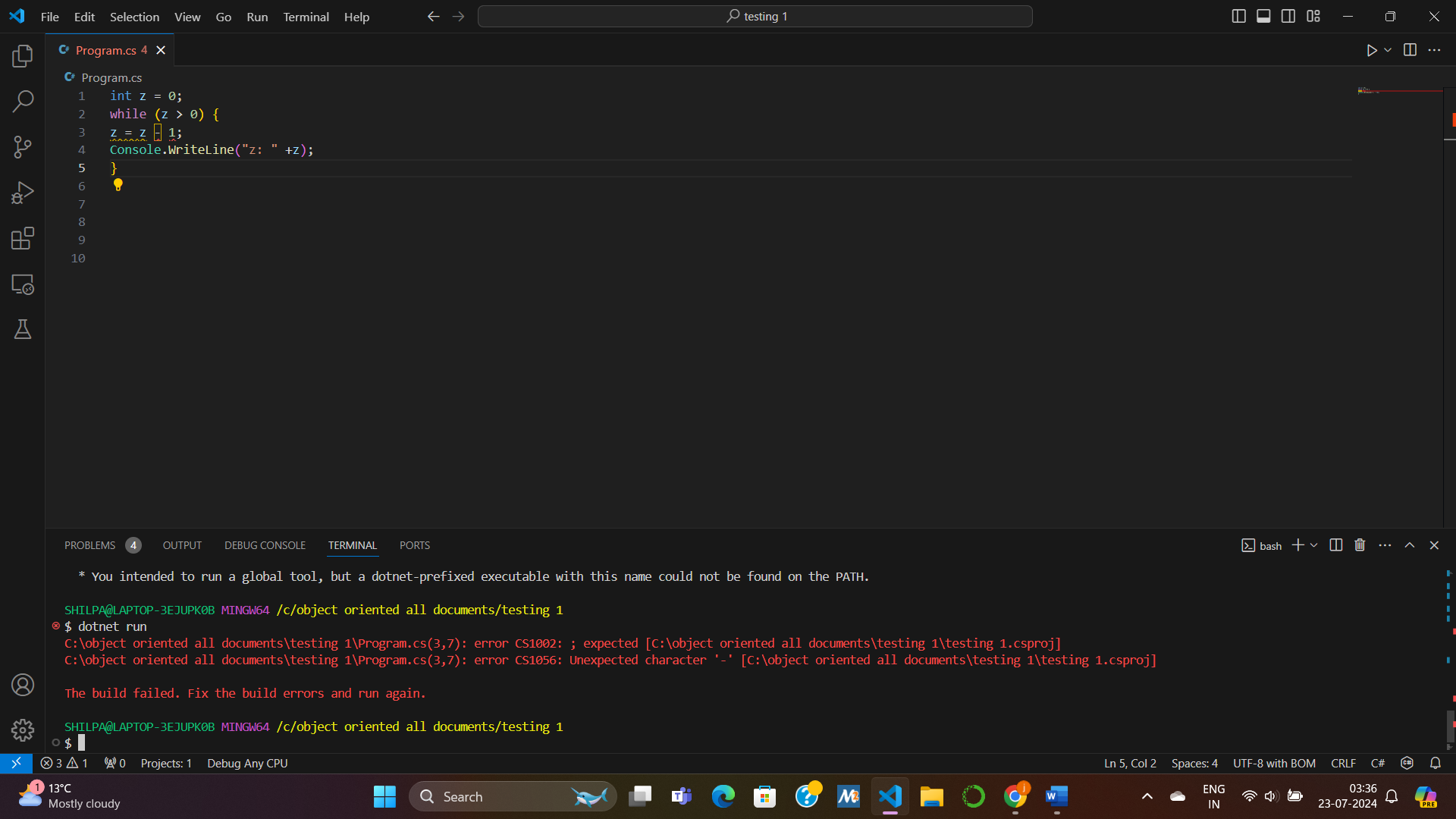
}



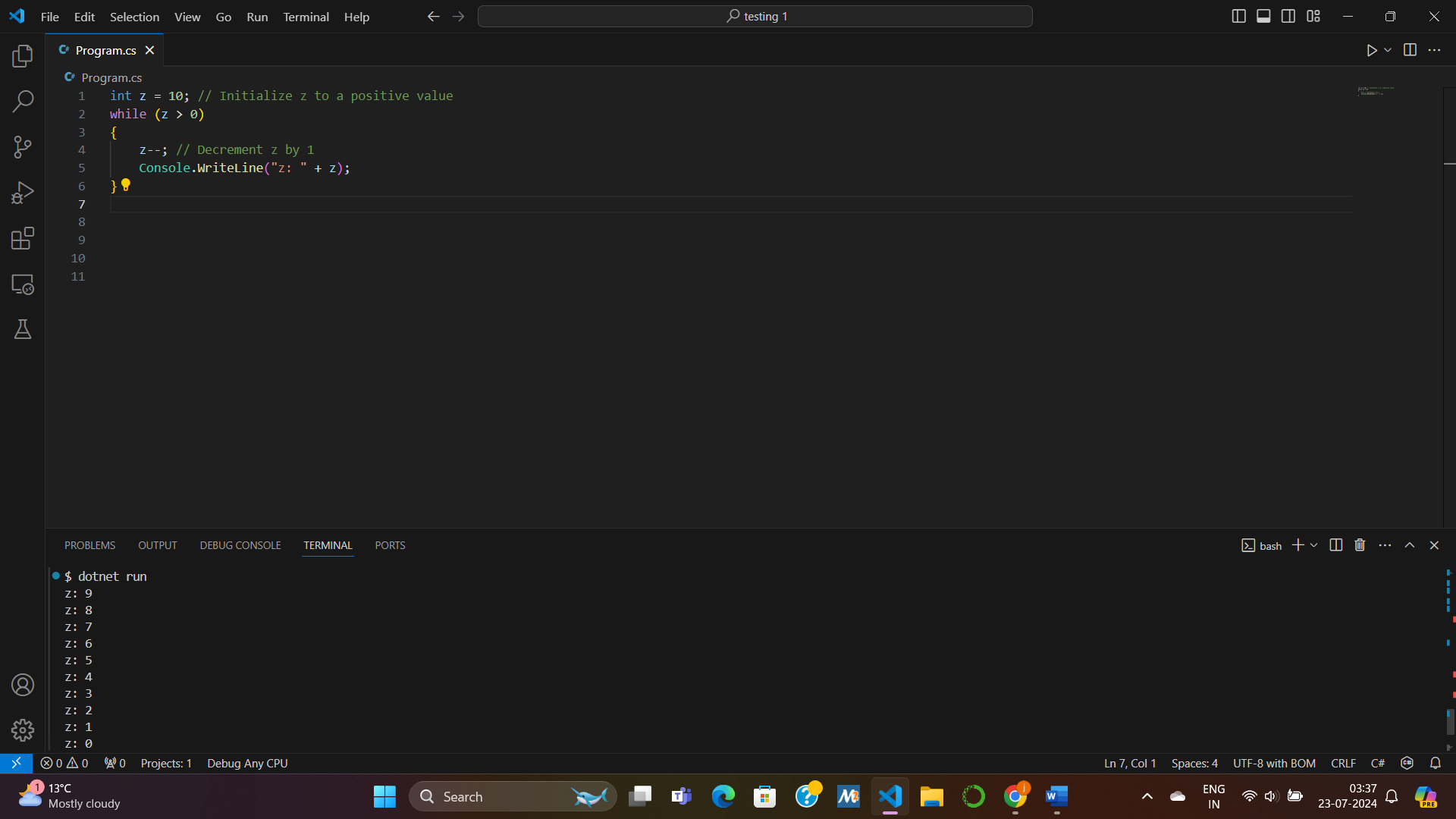


ISSUE:

1. The condition while (z > 0) will never be true initially because z is initialized to 0. Therefore, the loop will never execute.
2. Inside the loop, z = z ‐ 1; decrements z, but since the loop condition is never met (z > 0), the loop will not execute and hence Console.WriteLine("z: " + z); will not be executed



CORRECTED CODE:

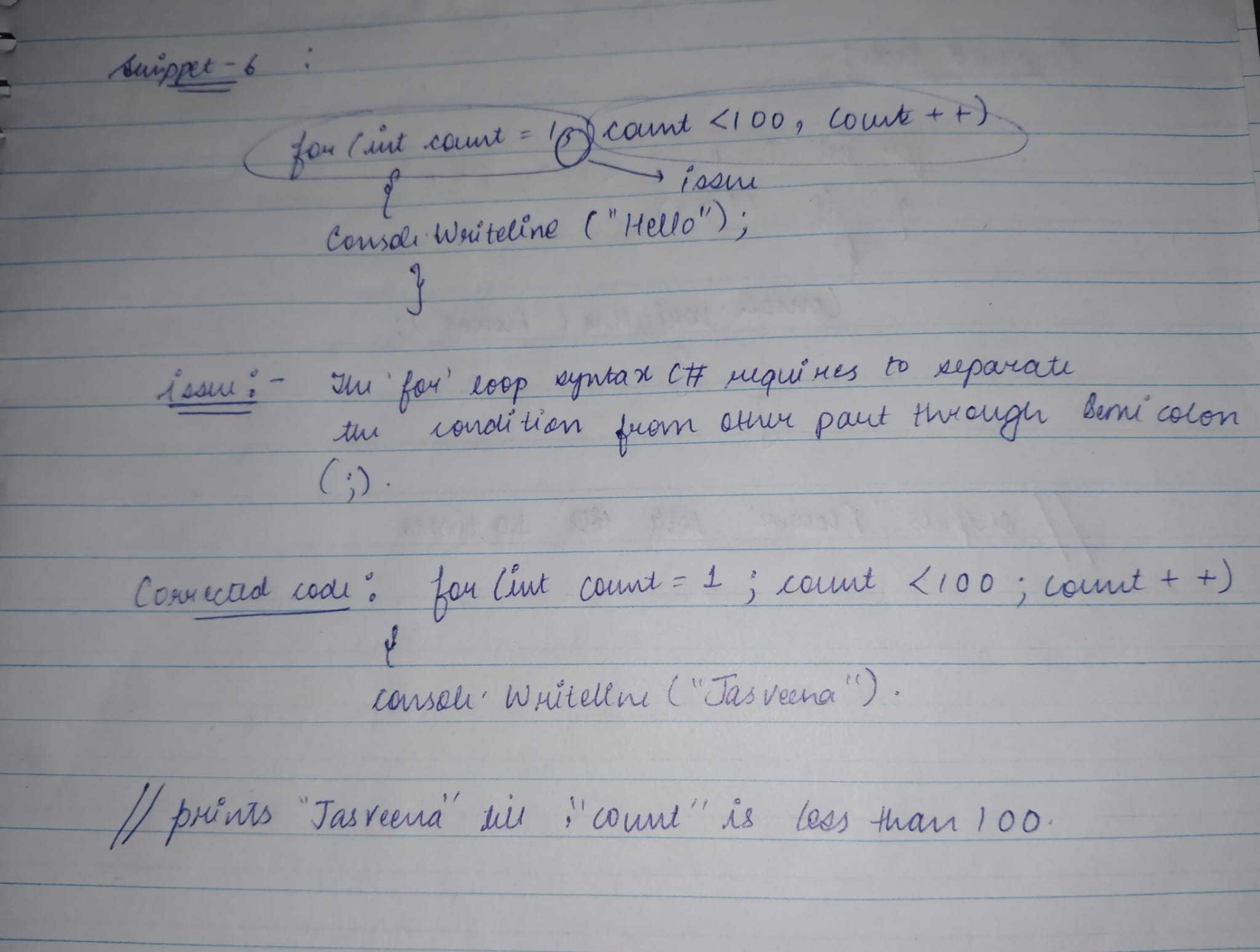


SNIPPET 6:

for(int count = 1, count < 100, count++) {

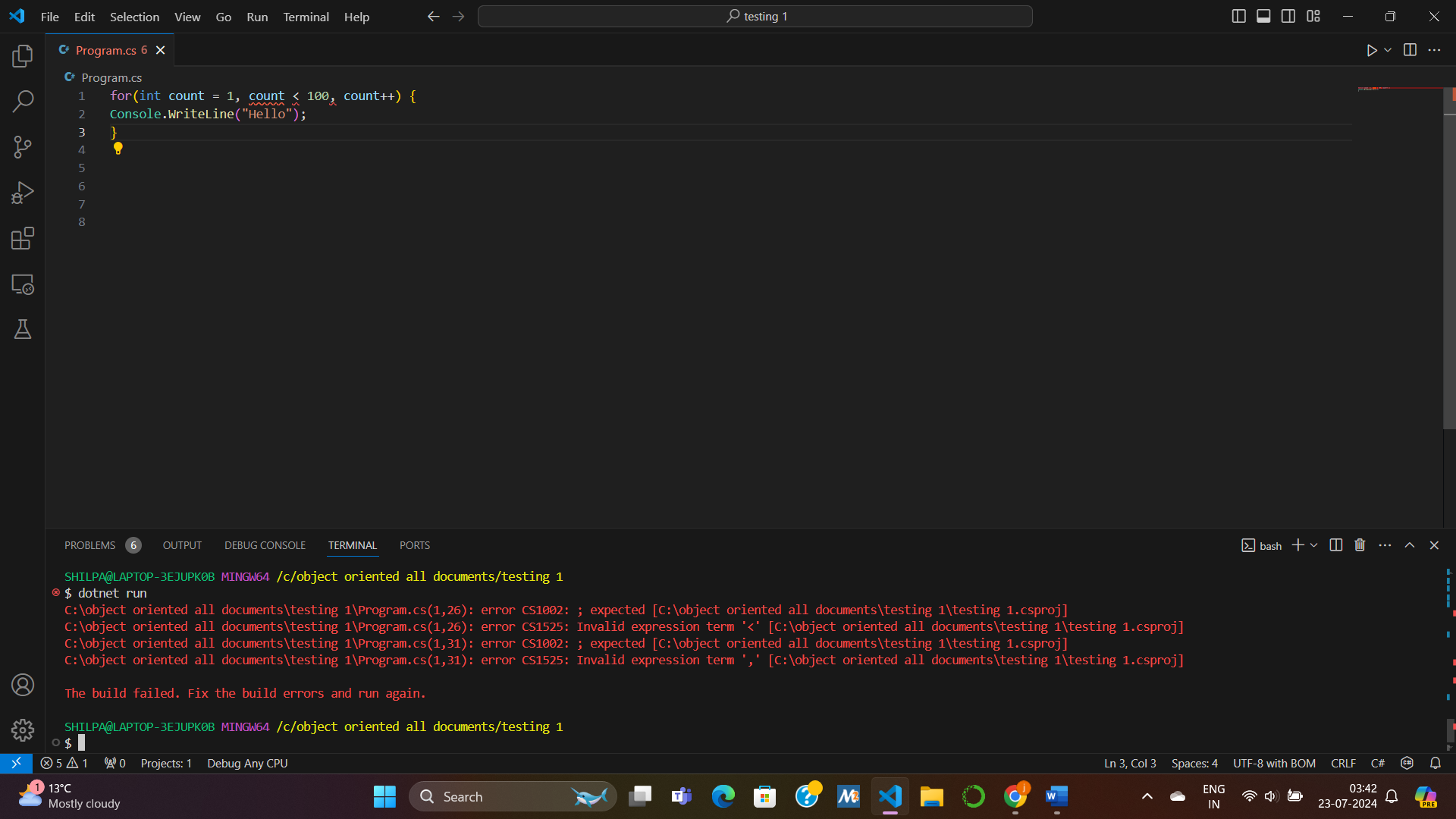
Console.WriteLine("Hello");

}

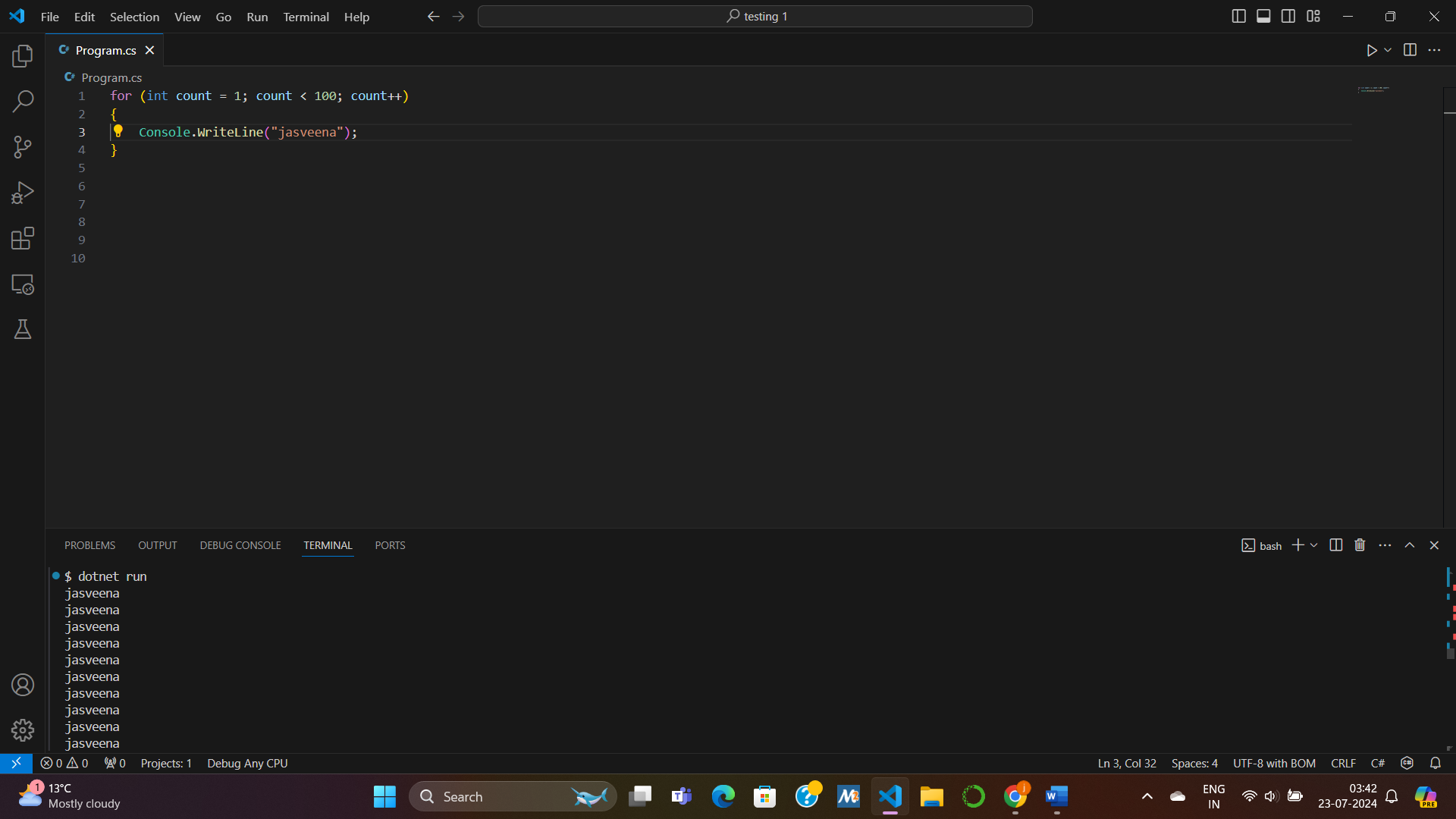


ISSUE:

1. The for loop syntax in C# requires three parts: initialization, condition, and iterator. These parts are separated by semicolons (;).
2. The condition part should use a semicolon (;) to separate the initialization from the condition.
3. Instead of separating the parts of the for loop with commas, they should be separated by semicolons (;).



CORRECTED CODE:



SNIPPET 7

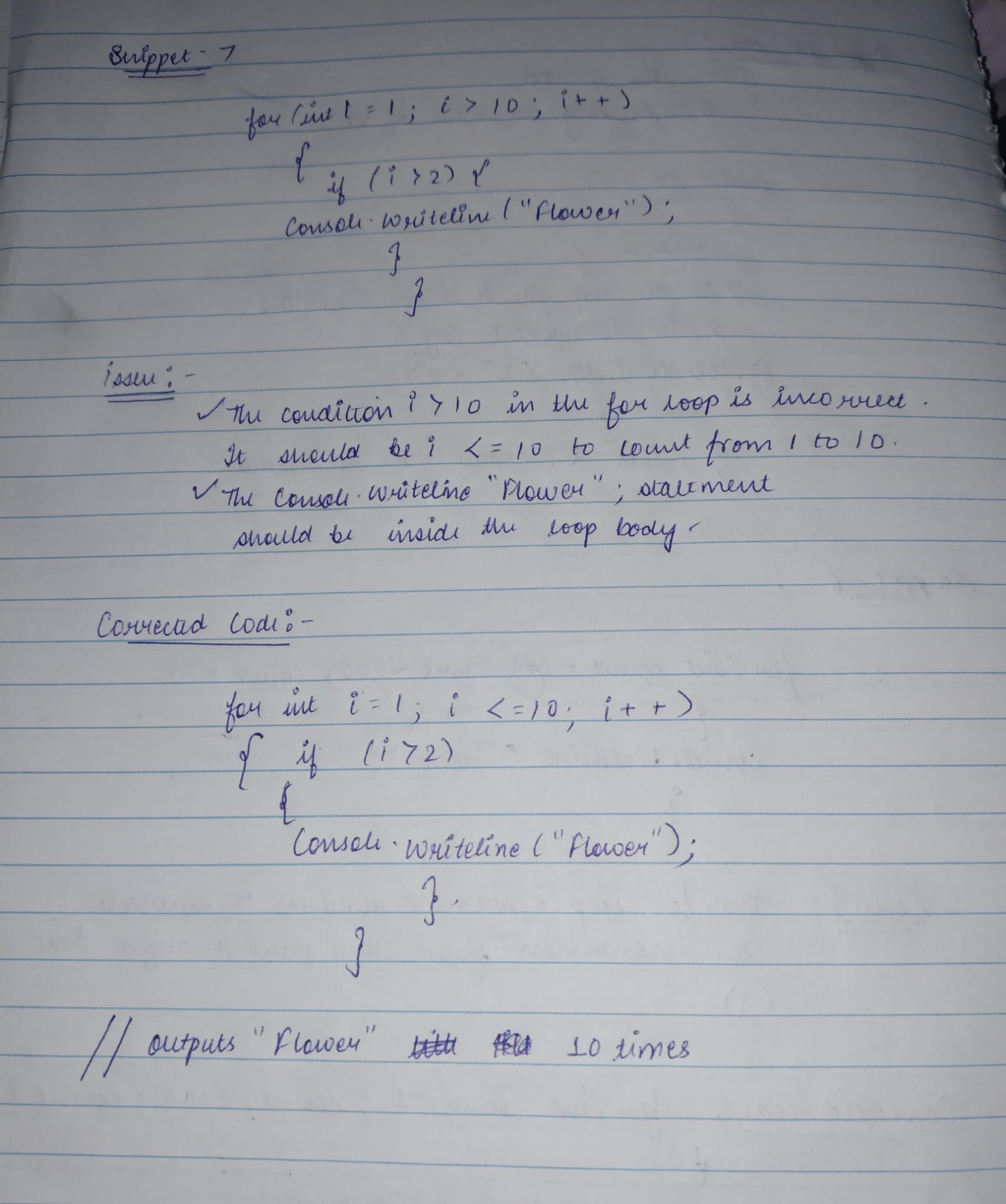
for(int I =1; i>10; i++) {

if (i>2) {

Console.WriteLine (“Flower”);

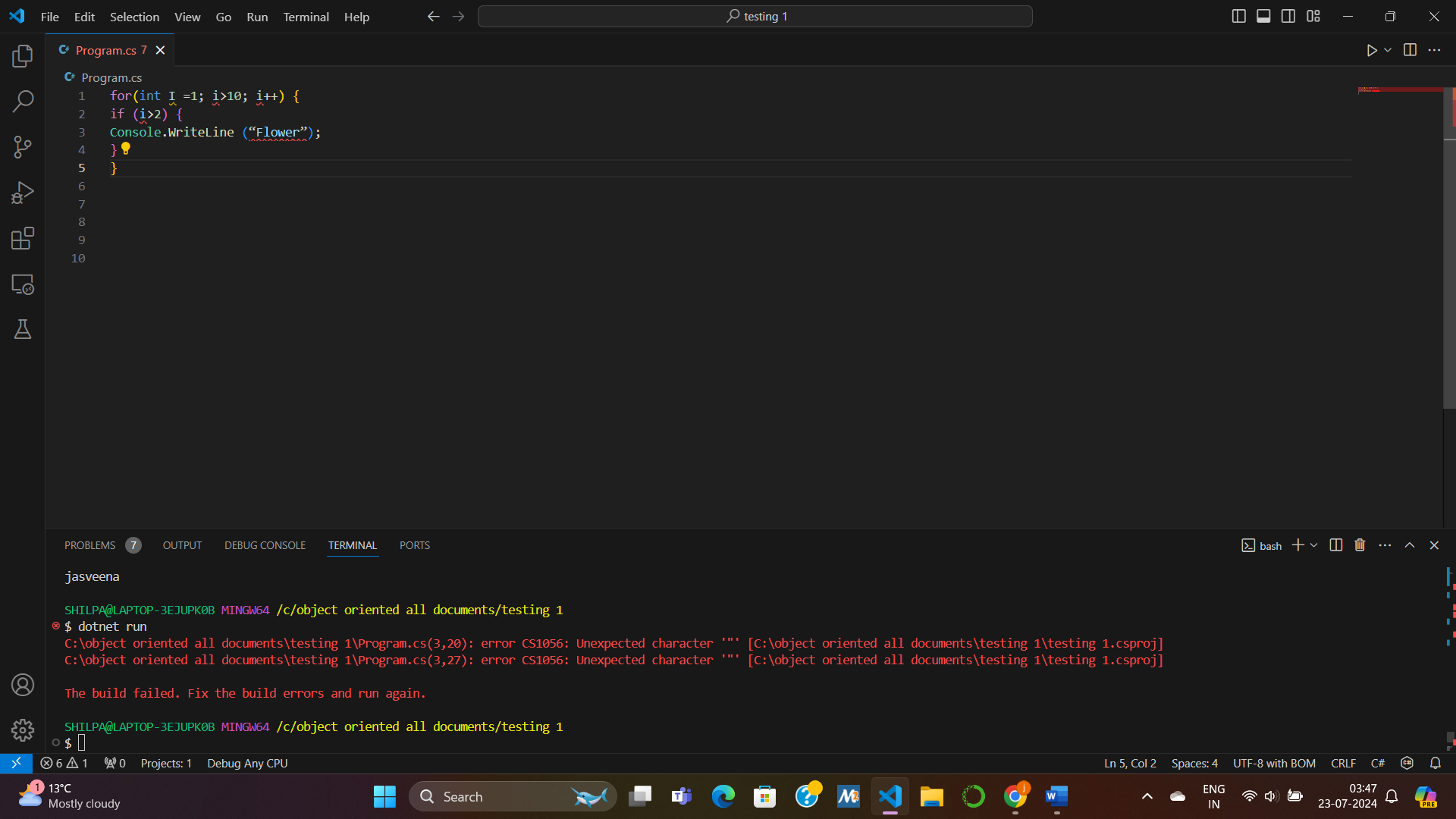
}

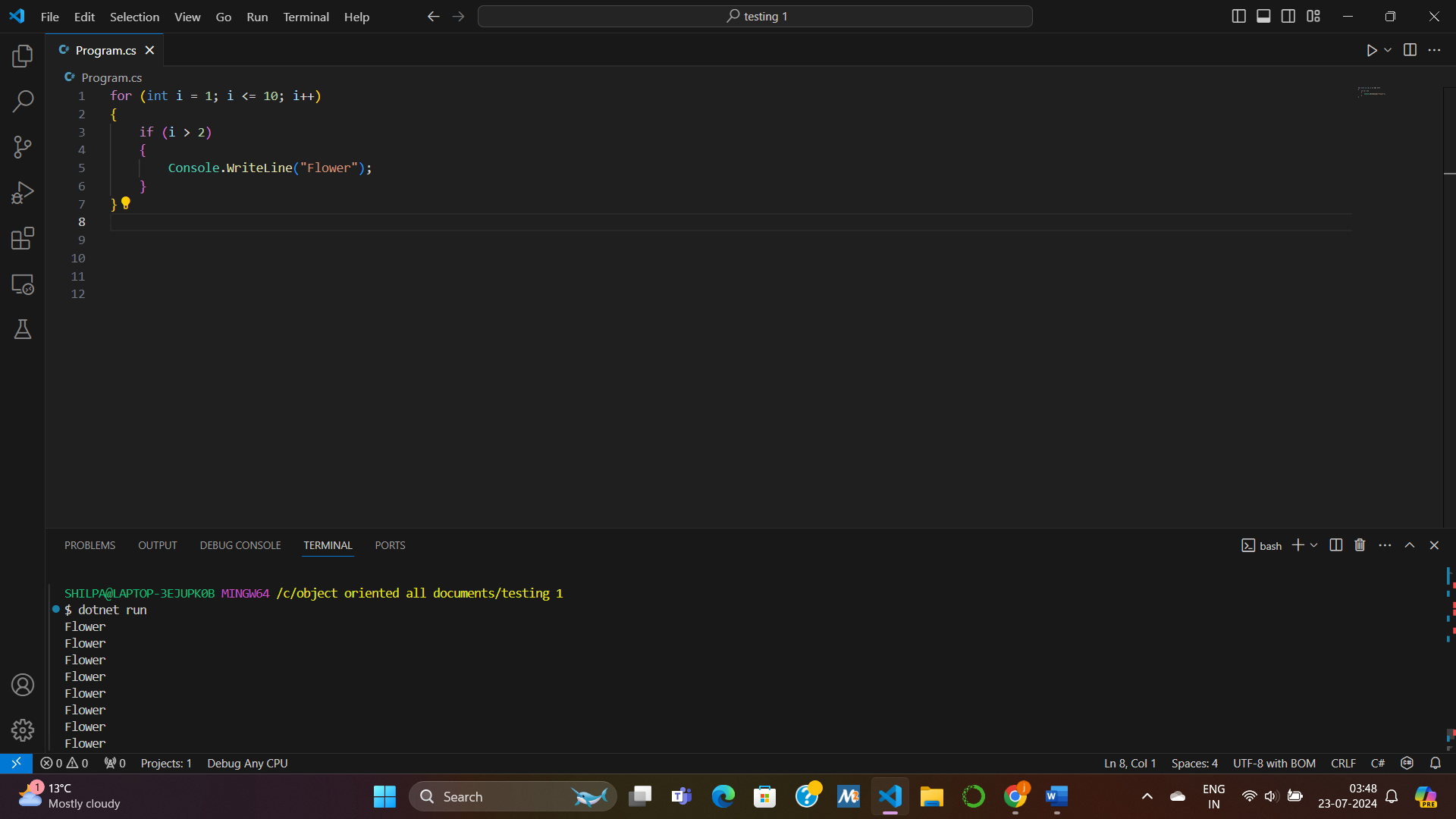
}



ISSUE:

1. The condition i > 10 in the for loop is incorrect. It should be i <= 10 to iterate from 1 to 10.
2. The Console.WriteLine("Flower"); statement should be inside the loop body and should be conditioned on i > 2.

CORRECTED CODE:

QUESTION 5:

SNIPPET 1:

int sum = 0 ;

int j = ‐5 ;

while ( sum <= 350 ) {

sum += j ;

j += 5 ;

}

CODE FRAGMENT USING FOR LOOP:

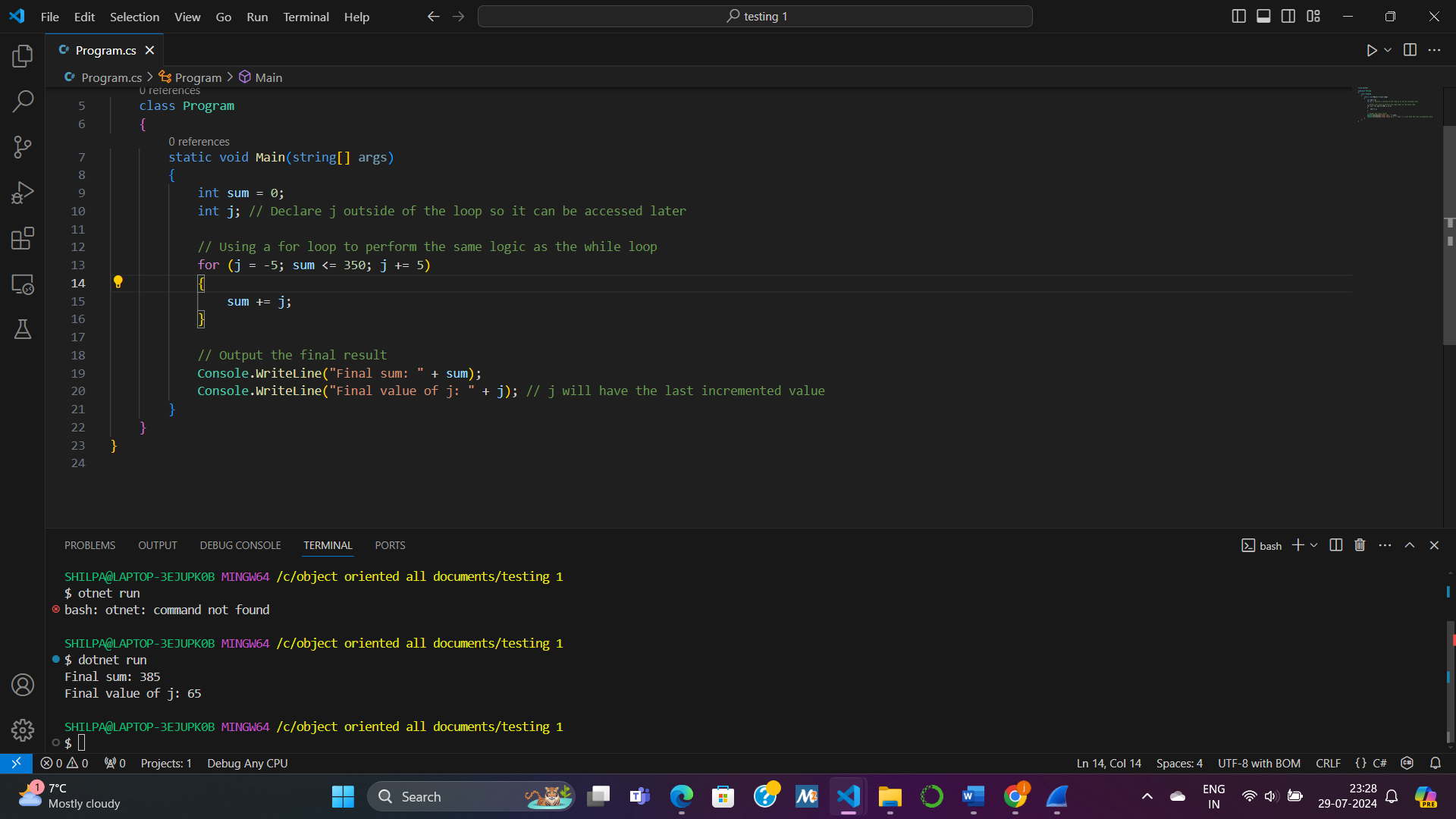
int sum = 0;

// Using a for loop to perform the same logic as the while loop

for (int j = -5; sum <= 350; j += 5)

{ sum += j;

}



SNIPPET 2:

int x = 0;

while ( x < 500 ) {

Console.WriteLine( x );

x = x + 5;

}

CODE USING FOR LOOP:

int x = 0;

while (x < 500)

{

Console.WriteLine(x);

x = x + 5;

}

