COMSATS UNIVERSITY ISLAMABAD ATTOCK CAMPUS

COMPUTER SCIENCE (SOFTWARE ENGINEERING)

THEORY ASSIGNMENT NO : 01

SUBJECT: ***VISUAL PROGRAMMING***

SUBMITTED BY: ***RAVIA IQBAL***

REGISTRATION NO: ***SP21-BSE-025/ATK***

SUBMITTED TO: ***SIR JAMAL***

SEMESTER: ***05***

DATE: ***19TH MARCH, 2023***

Program 01:

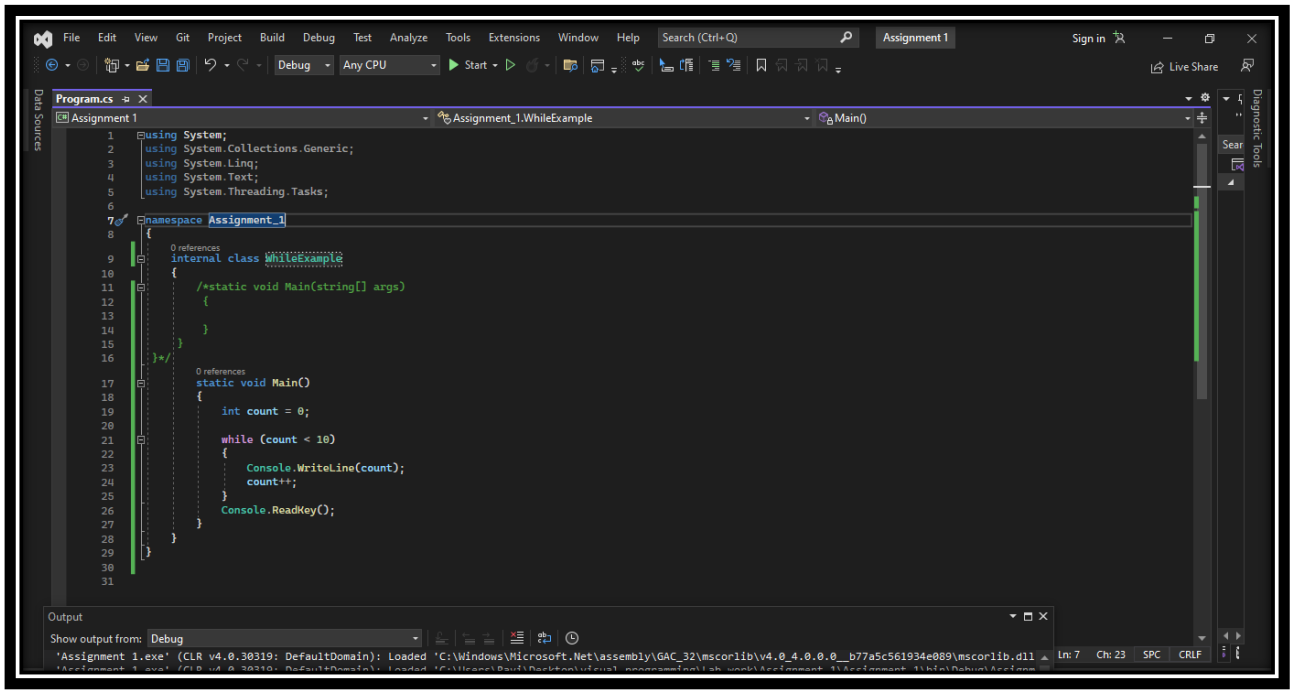
Given below is a program to print values from 0 to 9 using a while loop. Will the program generate the desired output? If not, why?

```
using System;

class WhileExample {

static void Main() {

int count = 0;

while (count <= 10) {

Console.WriteLine(count);

}}}
```

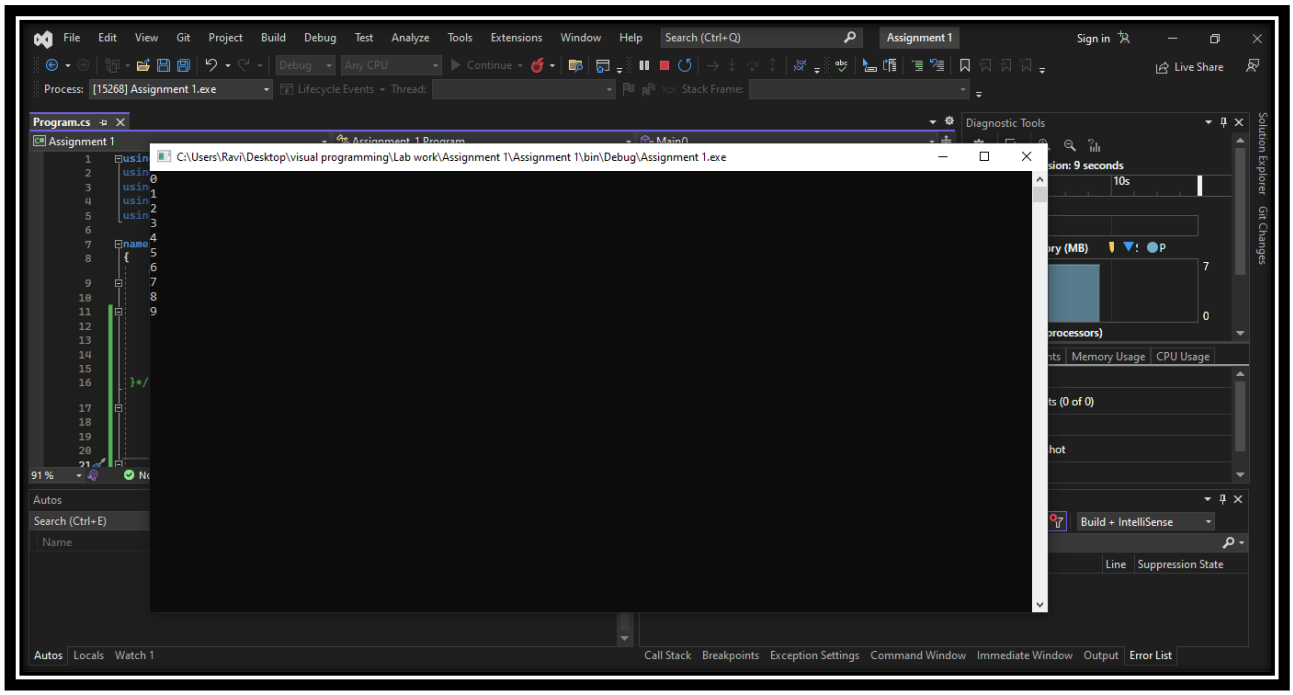
Solution:

The program will not generate the desire output because the while loop's condition includes the value 10, and the loop keeps running until the condition is false, the program will create values from 0 to 10 rather than from 0 to 9. The constraint should be modified to **count < 10** in order to report numbers from 0 to 9. And the other reason is that the while loop is running infinite time because the count++ are not written here so that's why it will generate the same output at infinite time.

Here is the corrected program:



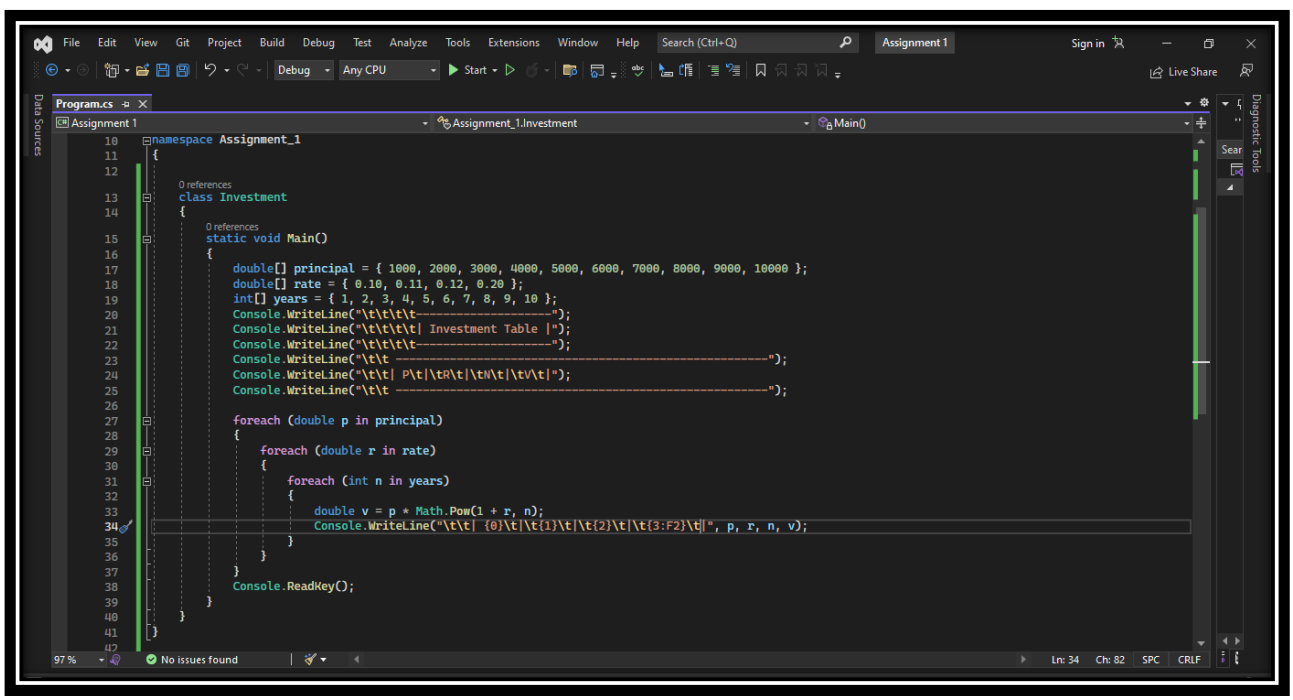
Output:



Program 02:

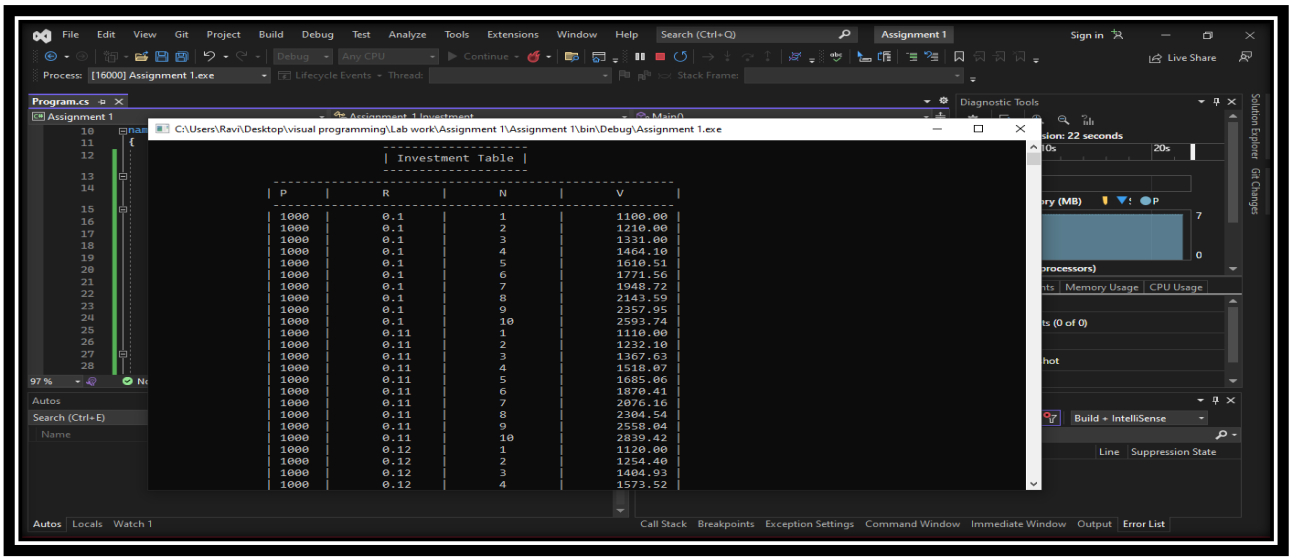
Write a program to evaluate the following investment equation V=P(1+r)" and print the tables which would give the value of V for various combination of the following values of P, rand n. P: 1000, 2000, 3000,... 10,000 r:0.10,0.11.0.12.0.20 n: 1,2,3,... 10

Solution:



```
namespace Assignment_1
{

    0 references
    class Investment
    {
        0 references
        static void Main()
        {
            double[] principal = { 1000, 2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000 };
            double[] rate = { 0.10, 0.11, 0.12, 0.20 };
            int[] years = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
            Console.WriteLine("\t\t\t\t--------------------");
            Console.WriteLine("\t\t\t\t| Investment Table |");
            Console.WriteLine("\t\t\t\t--------------------");
            Console.WriteLine("\t\t ------------------------------------------------------");
            Console.WriteLine("\t\t| P\t|\tR\t|\tN\t|\tV\t|");
            Console.WriteLine("\t\t ------------------------------------------------------");

            foreach (double p in principal)
            {
                foreach (double r in rate)
                {
                    foreach (int n in years)
                    {
                        double v = p * Math.Pow(1 + r, n);
                        Console.WriteLine("\t\t| {0}\t|\t{1}\t|\t{2}\t|\t{3:F2}\t|", p, r, n, v);
                    }
                }
            }
            Console.ReadKey();
        }
    }
}
```

Output:

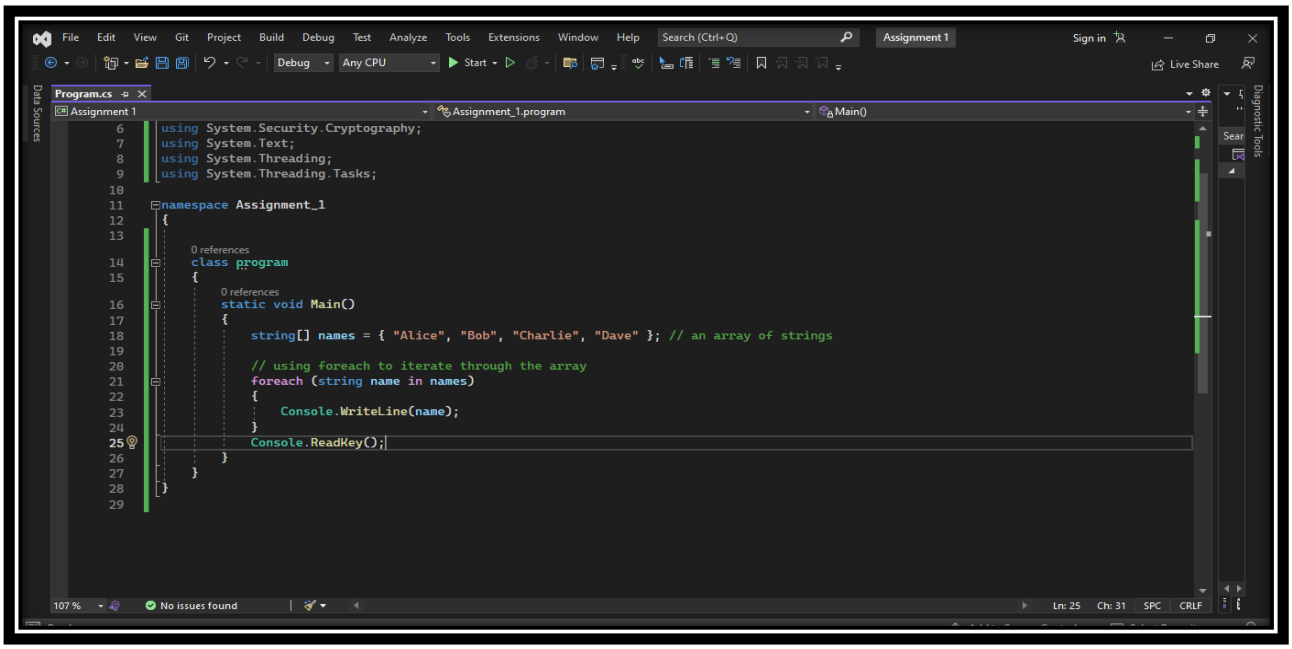


Program 3:

Illustrate the application of the foreach statement through a simple program.

Solution:

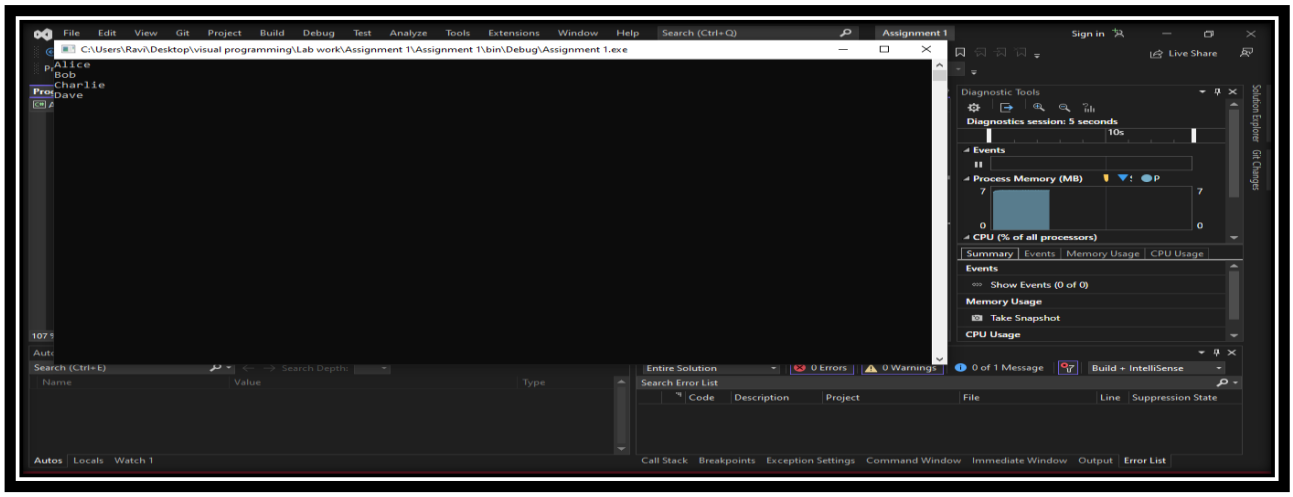
The foreach statement is used to iterate over a collection of element such as an array, list, or dictionary. Here's an example program that demonstrates the use of foreach with an array:



```
using System.Security.Cryptography;
using System.Text;
using System.Threading;
using System.Threading.Tasks;

namespace Assignment_1
{

    class program
    {
        static void Main()
        {
            string[] names = { "Alice", "Bob", "Charlie", "Dave" }; // an array of strings

            // using foreach to iterate through the array
            foreach (string name in names)
            {
                Console.WriteLine(name);
            }
            Console.ReadKey();
        }
    }
}
```

In this example, we have an array of strings named names. We then use the foreach statement to iterate through each element in the array and print it to the console. The output of this program will be:

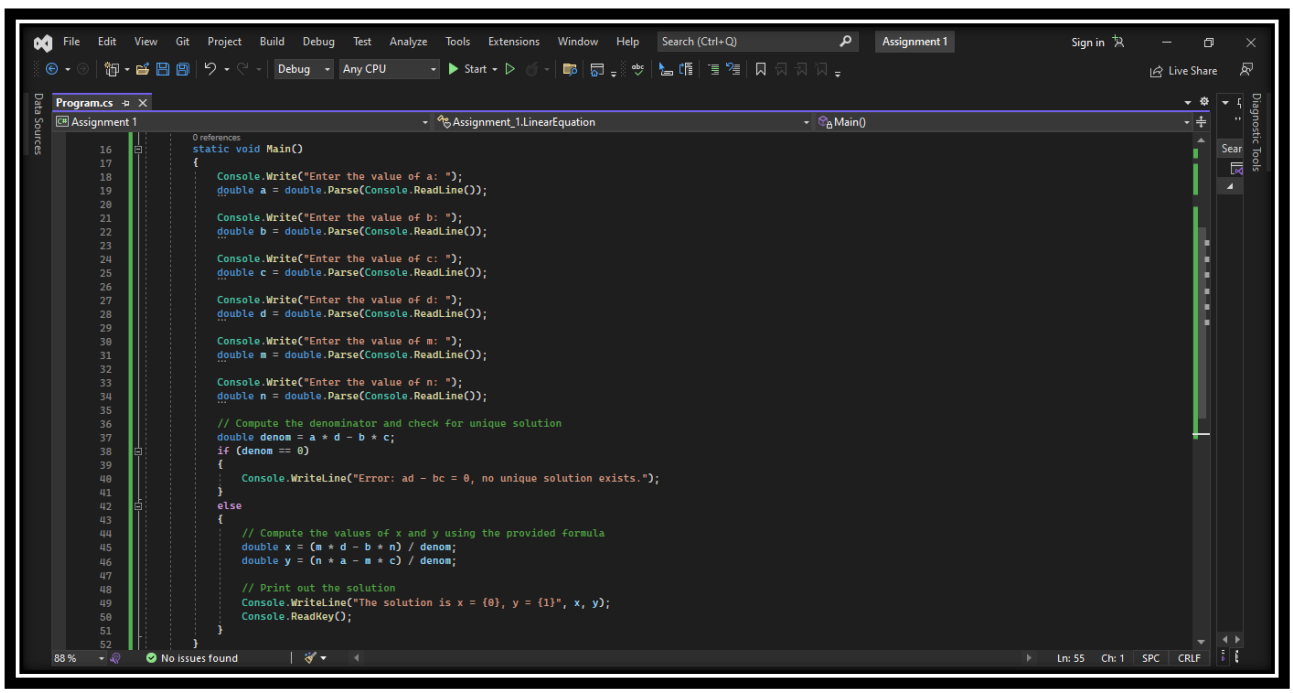
Output:



The foreach statement works by iterating over each element of a collection and assigning the value of the current element to a temporary variable (name in this case). The loop body then executes for each element in the collection. This makes it easier to iterate over collections, as it eliminates the need to use an index to access each element in the collection.

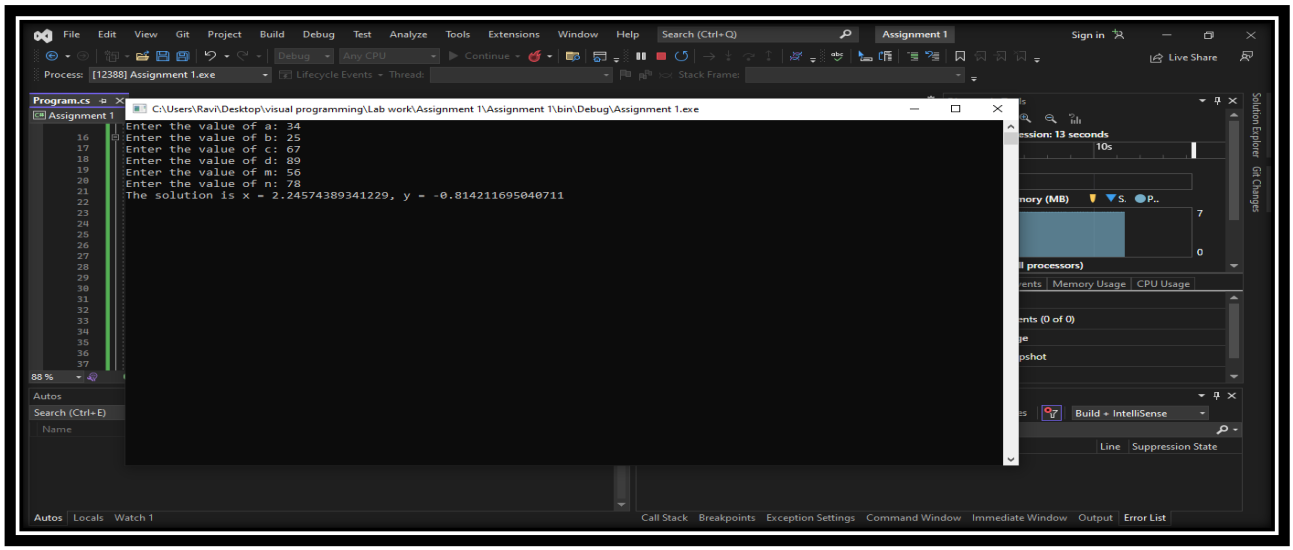
Program 4:

Write a program that will read the values of constants a, b, c, d, m and n and compute the values of x, and x.. An appropriate message should be printed if ad-cb=0.



```
static void Main()
{
    Console.Write("Enter the value of a: ");
    double a = double.Parse(Console.ReadLine());

    Console.Write("Enter the value of b: ");
    double b = double.Parse(Console.ReadLine());

    Console.Write("Enter the value of c: ");
    double c = double.Parse(Console.ReadLine());

    Console.Write("Enter the value of d: ");
    double d = double.Parse(Console.ReadLine());

    Console.Write("Enter the value of m: ");
    double m = double.Parse(Console.ReadLine());

    Console.Write("Enter the value of n: ");
    double n = double.Parse(Console.ReadLine());

    // Compute the denominator and check for unique solution
    double denom = a * d - b * c;
    if (denom == 0)
    {
        Console.WriteLine("Error: ad - bc = 0, no unique solution exists.");
    }
    else
    {
        // Compute the values of x and y using the provided formula
        double x = (m * d - b * n) / denom;
        double y = (n * a - m * c) / denom;

        // Print out the solution
        Console.WriteLine("The solution is x = {0}, y = {1}", x, y);
        Console.ReadKey();
    }
}
```

Output:



Program 5:

Given a list of marks ranging from 0 to 100, write a program to compute and print the number of students who have obtained marks

(a) in the range 81 to 100, (c) in the range 41 to 60, and (b) in the range 61 to 80,

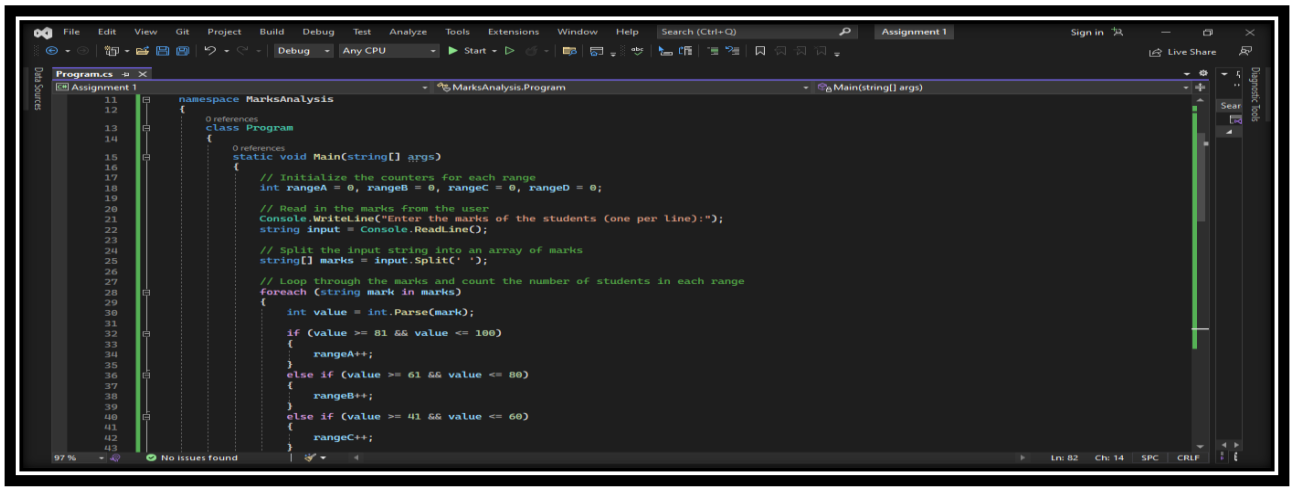
(d) in the range 0 to 40. The program should use a minimum number of if statements. 6.6 Admission to a professional course is subject to the following conditions.

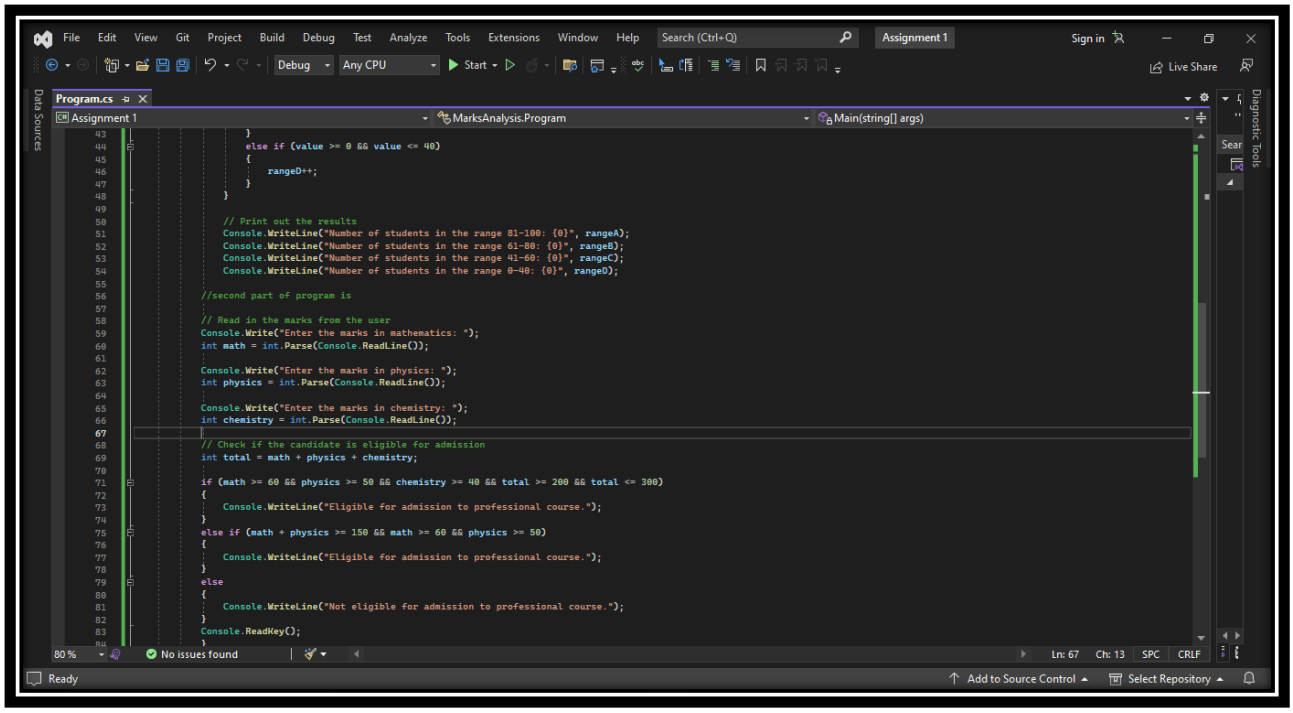
(a) Marks in mathematics >=60 (b) Marks in physics >=50 (c) Marks in chemistry >=40

(d) Total in all three subjects >=200 >=150 (or) Total in mathematics and physics

Given the marks in the three subjects, write a program to process the applications to list the eligible candidates.

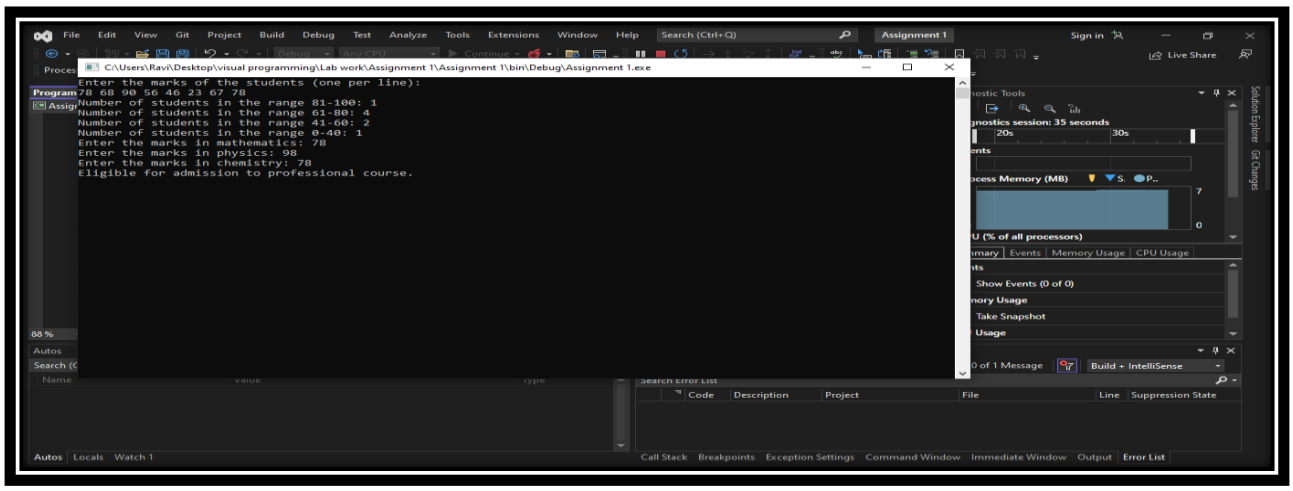
Solution:





```
            }
            else if (value >= 0 && value <= 40)
            {
                rangeD++;
            }
        }

        // Print out the results
        Console.WriteLine("Number of students in the range 81-100: {0}", rangeA);
        Console.WriteLine("Number of students in the range 61-80: {0}", rangeB);
        Console.WriteLine("Number of students in the range 41-60: {0}", rangeC);
        Console.WriteLine("Number of students in the range 0-40: {0}", rangeD);

    //second part of program is

    // Read in the marks from the user
    Console.Write("Enter the marks in mathematics: ");
    int math = int.Parse(Console.ReadLine());

    Console.Write("Enter the marks in physics: ");
    int physics = int.Parse(Console.ReadLine());

    Console.Write("Enter the marks in chemistry: ");
    int chemistry = int.Parse(Console.ReadLine());

    // Check if the candidate is eligible for admission
    int total = math + physics + chemistry;

    if (math >= 60 && physics >= 50 && chemistry >= 40 && total >= 200 && total <= 300)
    {
        Console.WriteLine("Eligible for admission to professional course.");
    }
    else if (math + physics >= 150 && math >= 60 && physics >= 50)
    {
        Console.WriteLine("Eligible for admission to professional course.");
    }
    else
    {
        Console.WriteLine("Not eligible for admission to professional course.");
    }
    Console.ReadKey();
```

output :



```
C:\Users\Ravi\Desktop\visual programming\Lab work\Assignment 1\Assignment 1\bin\Debug\Assignment 1.exe
Enter the marks of the students (one per line):
78 68 90 56 46 23 67 78
Number of students in the range 81-100: 1
Number of students in the range 61-80: 4
Number of students in the range 41-60: 2
Number of students in the range 0-40: 1
Enter the marks in mathematics: 78
Enter the marks in physics: 98
Enter the marks in chemistry: 78
Eligible for admission to professional course.
```

Program 6:

Find the error in the following program.

using System;

class TerneryExample {

static void Main() [

 int num=3;

string result = (num < 2) ? "True"; "False"; Console.WriteLine(result);

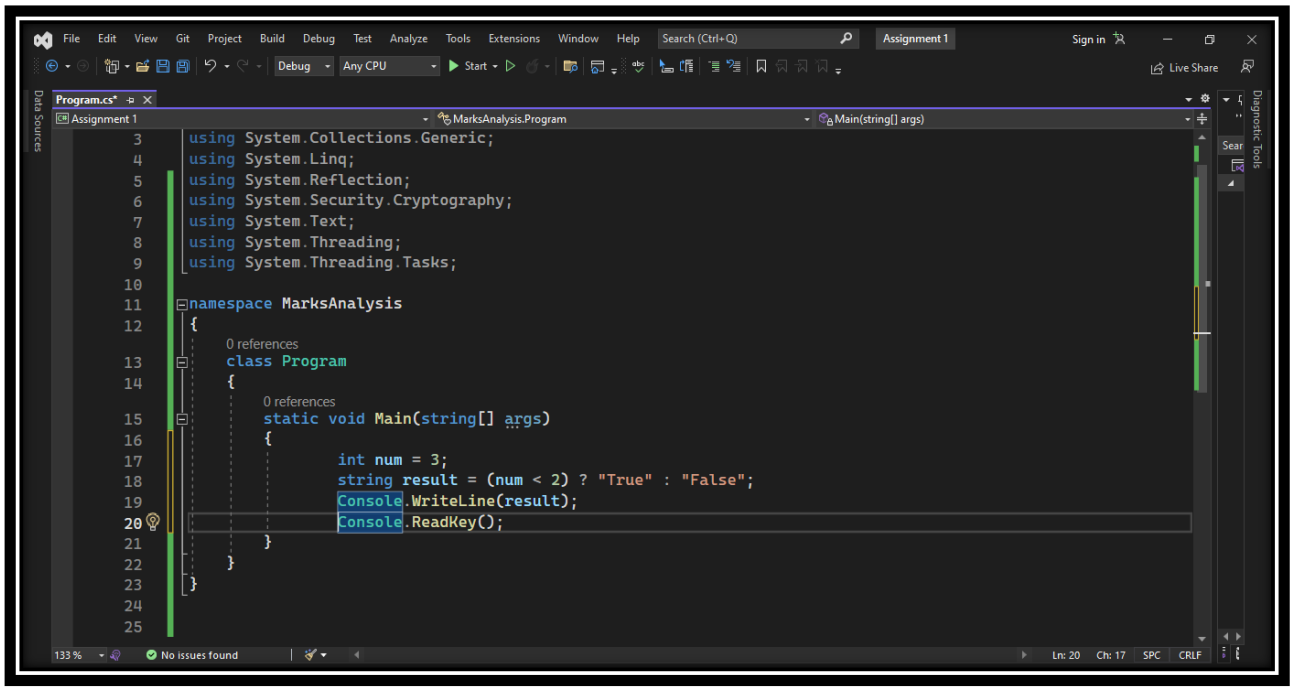
}}

Solution:

The following program has a syntax error. The error is in the Main method, where the opening curly brace is incorrect, it should be a round curly brace { instead of a square bracket [. Similarly, the string result line should be split into two lines, each contain a value for the true/false outcomes of the ternary operator. There is also a missing colon between the expressions for true and false.

The correct program should be:

Code:



Program 7:

Debug the program given below. using System;

class Student Record

{ }

static void Main(string[] args)[}

Console.WriteLine("Enter the name of the student: "); string name = Console.ReadLine();

Console.WriteLine("Enter the grade for the student: "); string grade = Console.ReadLine():

if (grade="a")

Console.WriteLine("{0} is an Outstanding student.", name);

Console.WriteLine("You have received a scholarship.");

else if (grade "b")

Console.WriteLine("{(0) is a Good student.",name); else if (grade == "C")

Console.WriteLine("{0) is an Average student.",name); else if (grade == "d")

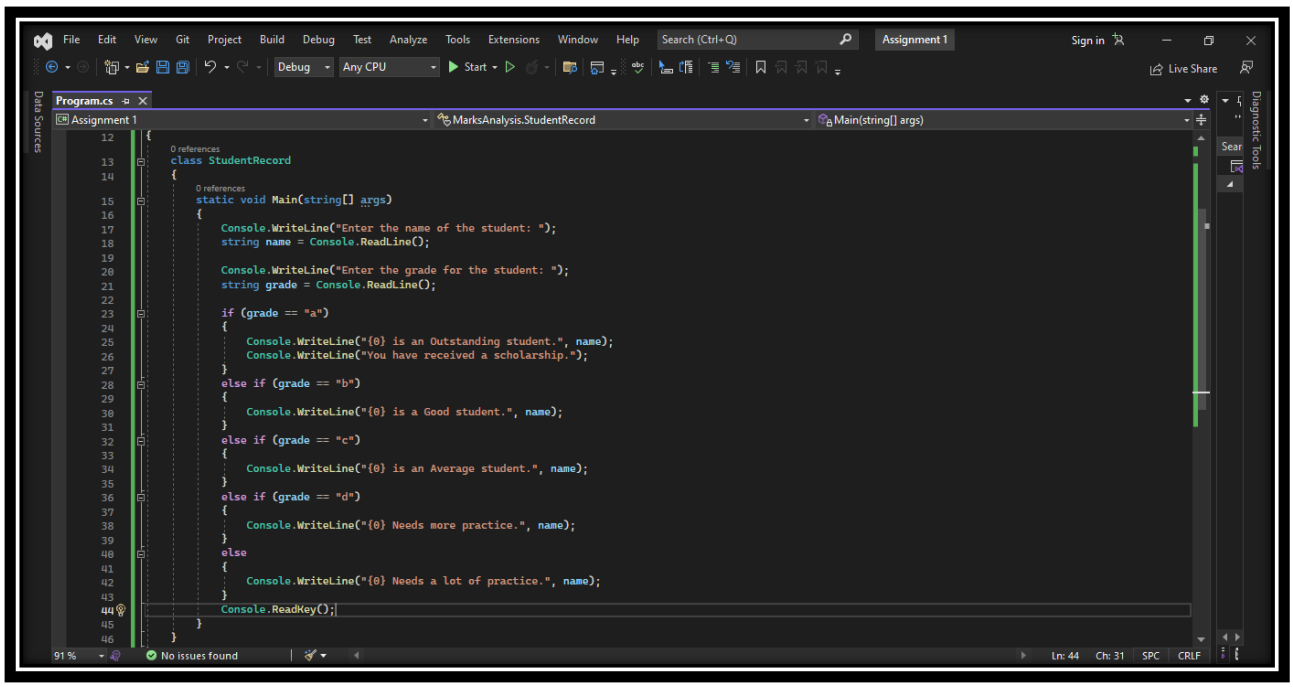
Console.WriteLine("{0) Needs more practice.", name);

else

Console.WriteLine("(0) Need a lot of practice.",name);

Solution:

The program given below contains multiple errors. Here is the debugged code:



The following are the errors and their corrections:

i. The class name contains a space. This is not allowed in C#. The class name has been corrected to StudentRecord.
ii. The opening brace for the Main method is misplaced. It should come immediately after the method signature, not after the class closing brace.
iii. There is a colon instead of a semicolon at the end of the line that reads the grade from the user's input. This has been corrected.
iv. The comparison operators in the if statements are assignment operators. The = operator is used for assignment, while the == operator is used for comparison. The comparison operators have been corrected.

v. The placeholders in the Console.WriteLine statements are not closed properly. They have been corrected by using curly braces instead of parentheses.

Program 8:

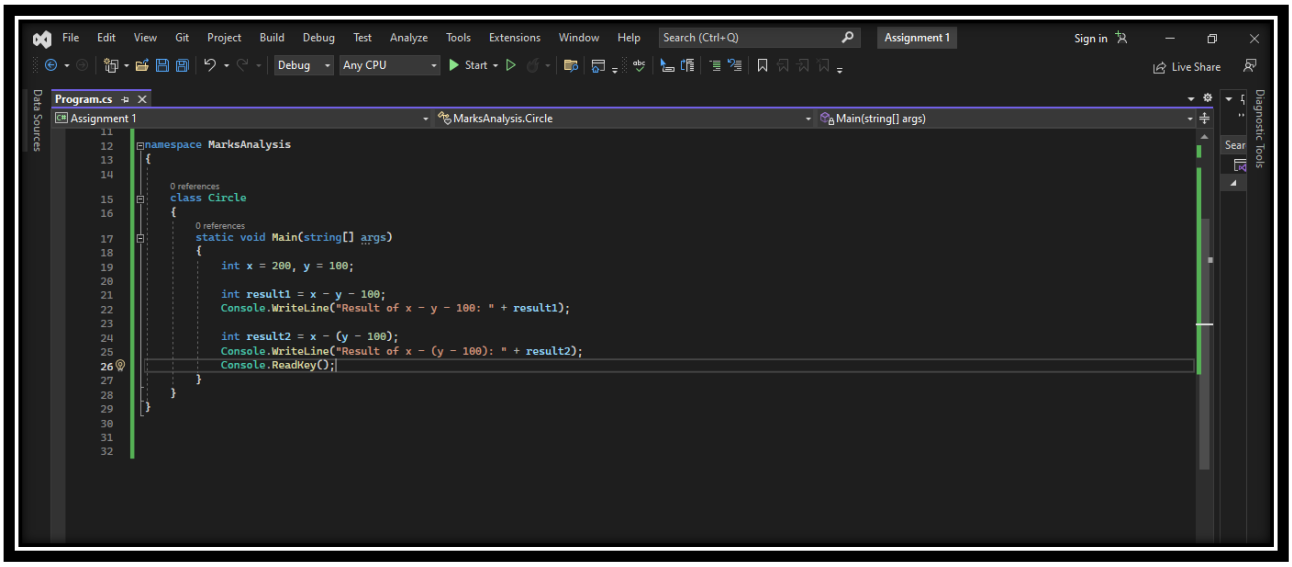
State why the expression x-y-100 is invalid but the expression x-(y-100) is valid. Execute a program to demonstrate your answer.

Solution:

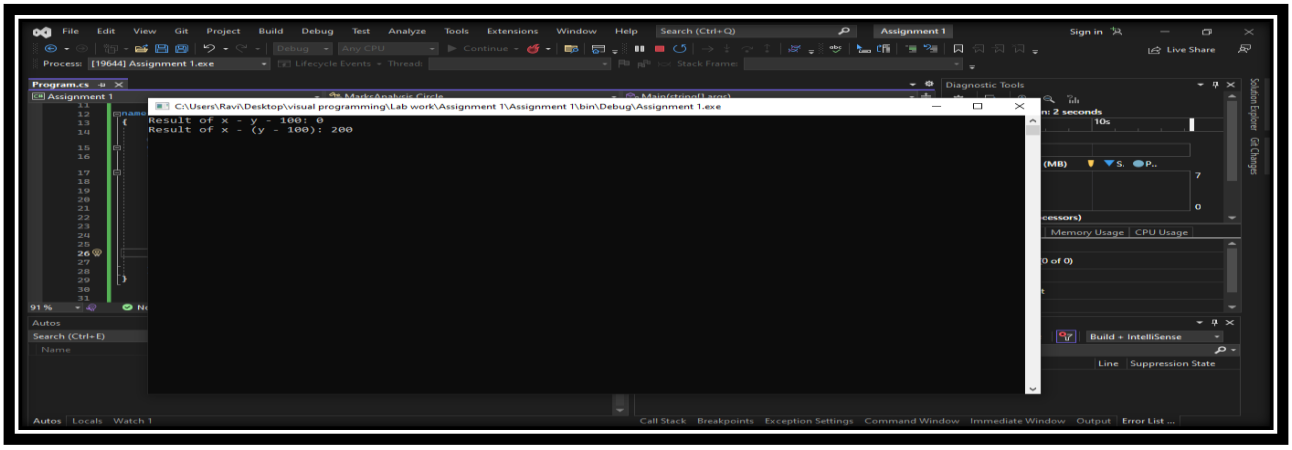
The expression x-y-100 is invalid because it is ambiguous and can have multiple interpretations based on the order of operations. It is not clear whether we should subtract y from x first or subtract 100 from y first. This can lead to different results depending on the interpretation.

On the other hand, the expression x-(y-100) is valid because it is unambiguous and its meaning is clear. We should first subtract 100 from y and then subtract the result from x.

Program:



Output:



Program 9:

The straight-line method of computing the yearly depreciation of the value of an item is given by Purchase price-Salvage value Depreciation = Years of service Write a program to determine the salvage value of an item when the purchase price, years of service and the annual depreciation are given.

Solution:

```
namespace MarksAnalysis
{

    0 references
    class FahrenheitToCelsius
    {
        0 references
        static void Main()
        {
            double purchasePrice = 1000.0;
            double annualDepreciation = 100.0;
            int yearsOfService = 5;

            // calculate salvage value
            double salvageValue = purchasePrice - annualDepreciation * yearsOfService;

            // display result
            Console.WriteLine("Salvage value: {0}", salvageValue);
            Console.ReadKey();
        }
    }
}
```

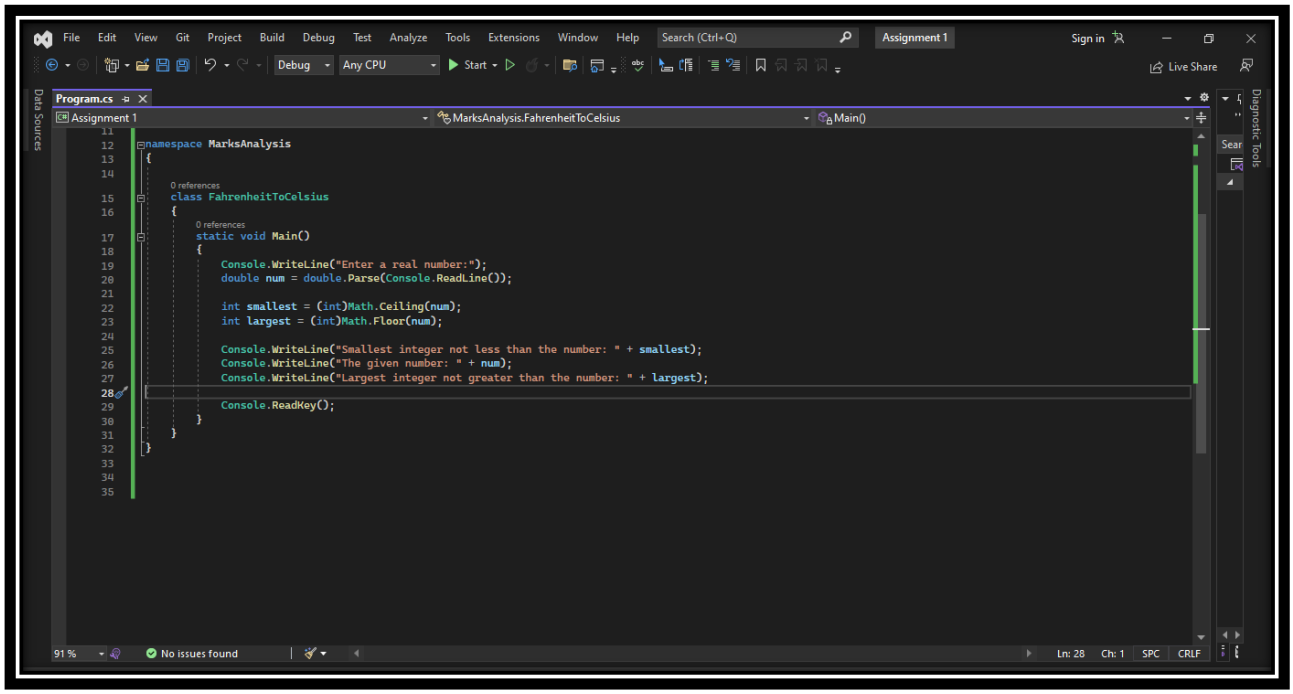
Output:

```
C:\Users\Ravi\Desktop\visual programming\Lab work\Assignment 1\Assignment 1\bin\Debug\Assignment 1.exe
Salvage value: 500
```

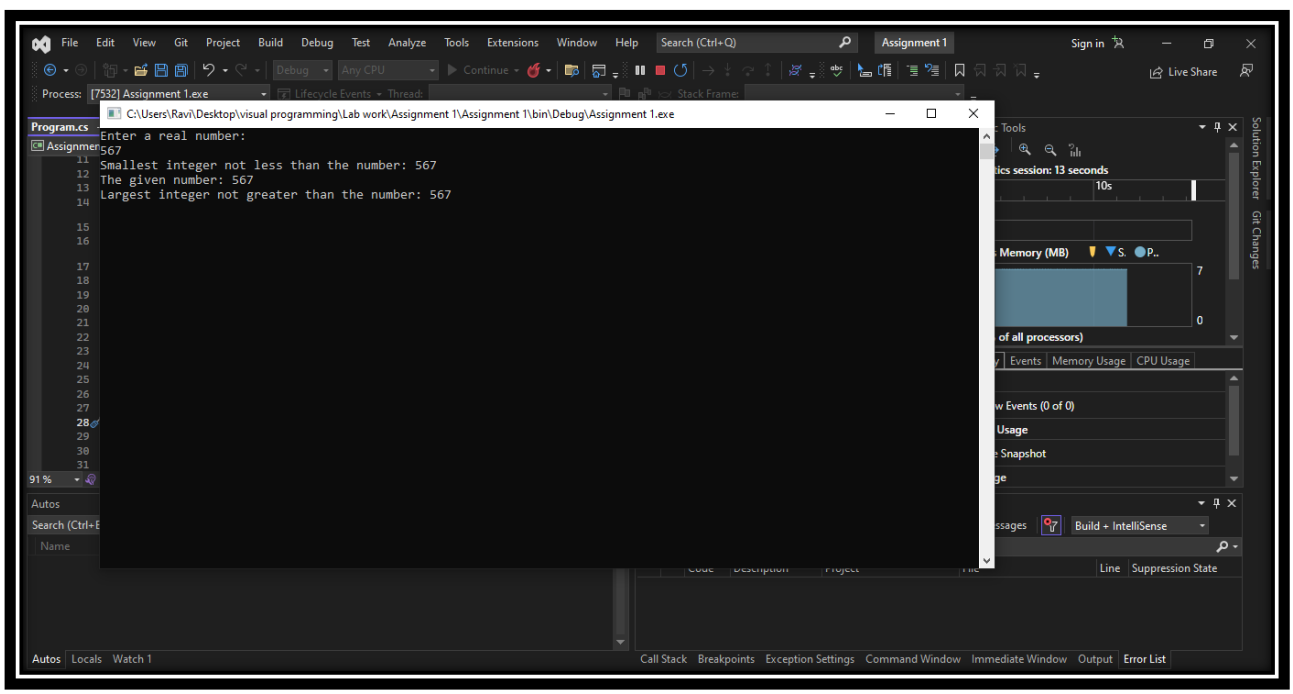
Program 10:

Write a program that will read a real number from the keyboard and print the following output in one line: Smallest integer not less than than the number The given number Largest integer not greater than the number.

Solution:



Output:



Program 11:

The total distance travelled by a vehicle in t seconds is given by distance = ut (at2)/2 where u is the initial velocity (metres per second), a is the acceleration (metres per second2). Write a program to evaluate the distance travelled at regular intervals of time, given the values of u and a. The program should provide the flexibility to the user to select his own time intervals and repeat the calculations for different values of u and a.

Solution:

```
class FahrenheitToCelsius
{
    static void Main()
    {
        double u, a, t, distance;
        int timeInterval;

        Console.Write("Enter the initial velocity (u): ");
        u = double.Parse(Console.ReadLine());

        Console.Write("Enter the acceleration (a): ");
        a = double.Parse(Console.ReadLine());

        Console.Write("Enter the time interval: ");
        timeInterval = int.Parse(Console.ReadLine());

        Console.WriteLine("Time (s)\tDistance (m)");

        for (t = 0; t <= timeInterval; t++)
        {
            distance = u * t + 0.5 * a * t * t;
            Console.WriteLine(t + "\t\t" + distance);
        }
            Console.ReadKey();
    }
}
}
```

Output:

```
Enter the initial velocity (u): 18
Enter the acceleration (a): 45
Enter the time interval: 10
Time (s)        Distance (m)
0               0
1               40.5
2               126
3               256.5
4               432
5               652.5
6               918
7               1228.5
8               1584
9               1984.5
10              2430
```

Program 12:

Write a program to calculate the frequency for different values of C starting from 0.01 to 0.1 in steps of 0.01. C#

Solution:

```
using System.Text;
using System.Threading;
using System.Threading.Tasks;

namespace MarksAnalysis
{

    class DampedNaturalFrequency
    {
        static void Main()
        {

            double Frequency(double c)
            {
                return 1 / (2 * Math.PI * Math.Sqrt(c * 1e-9 * 0.000000001 * 1000000));
            }
                for (int i = 1; i <= 10; i++)
                {
                    double c = i / 100.0;
                    double freq = Frequency(c);
                    Console.WriteLine("C = {0:F2} uF, F = {1:F2} Hz", c, freq);
                }
            Console.ReadKey();

        }
    }
}
```

Output:

```
C:\Users\Ravi\Desktop\visual programming\Lab work\Assignment 1\Assignment 1\bin\Debug\Assignment 1.exe
C = 0.01 uF, F = 1591549.43 Hz
C = 0.02 uF, F = 1125395.40 Hz
C = 0.03 uF, F = 918881.49 Hz
C = 0.04 uF, F = 795774.72 Hz
C = 0.05 uF, F = 711762.54 Hz
C = 0.06 uF, F = 649747.33 Hz
C = 0.07 uF, F = 601549.14 Hz
C = 0.08 uF, F = 562697.70 Hz
C = 0.09 uF, F = 530516.48 Hz
C = 0.10 uF, F = 503292.12 Hz
```

Program 13:

Find error, if any, in the following segment.

for (int m = 1; m< 100; m)

{ }

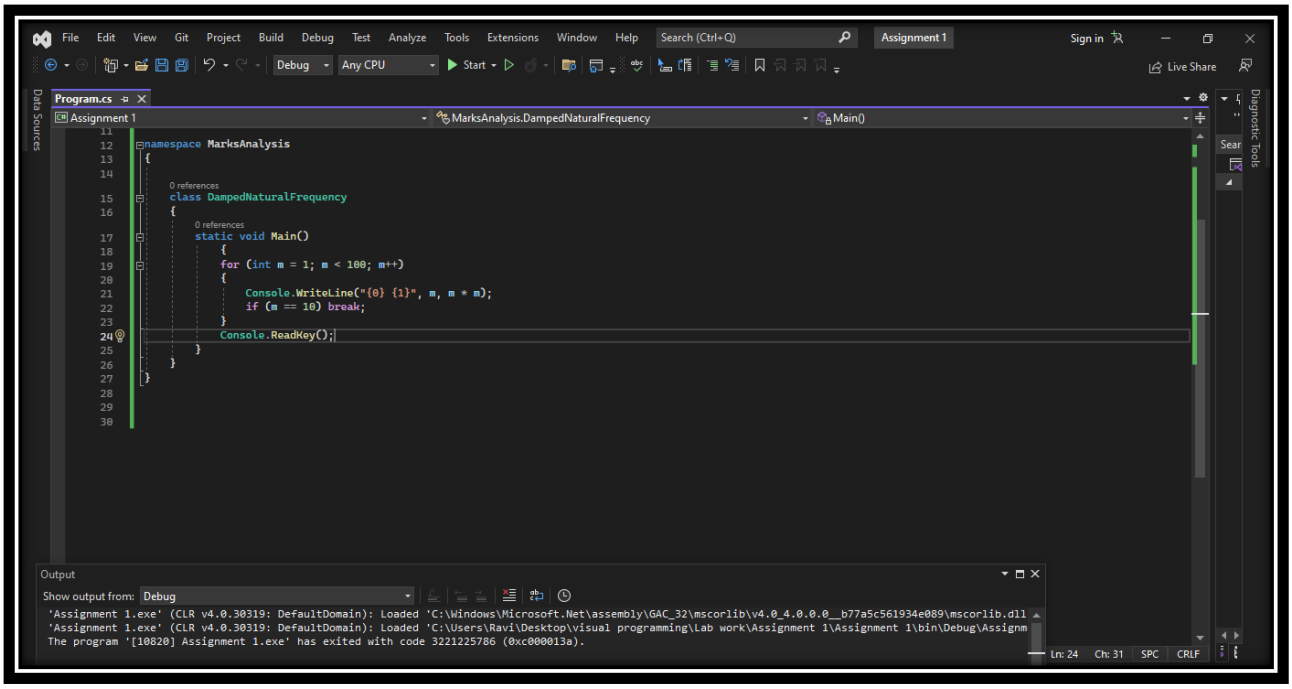
Console.WriteLine (m, m*m); if (m=10) break;

Solution:

There are a few errors in the following code segment:

i. The condition of the for loop is m, which is not being incremented. This will result in an infinite loop as m will always be less than 100.
ii. The Console.WriteLine statement is outside the scope of the for loop, so m will not be accessible at that point.
iii. The if statement is also outside the scope of the for loop, so the break statement will not function as intended.

Correct code:



Program 14:

Which of the conversions in the given program are invalid?

using System;

class ConversionExample

{ }

public static void Main() [

int ilnt=22;

long ILongint 44; double dDouble =1.406;

ILongint =ilnt; dDouble=ilnt;

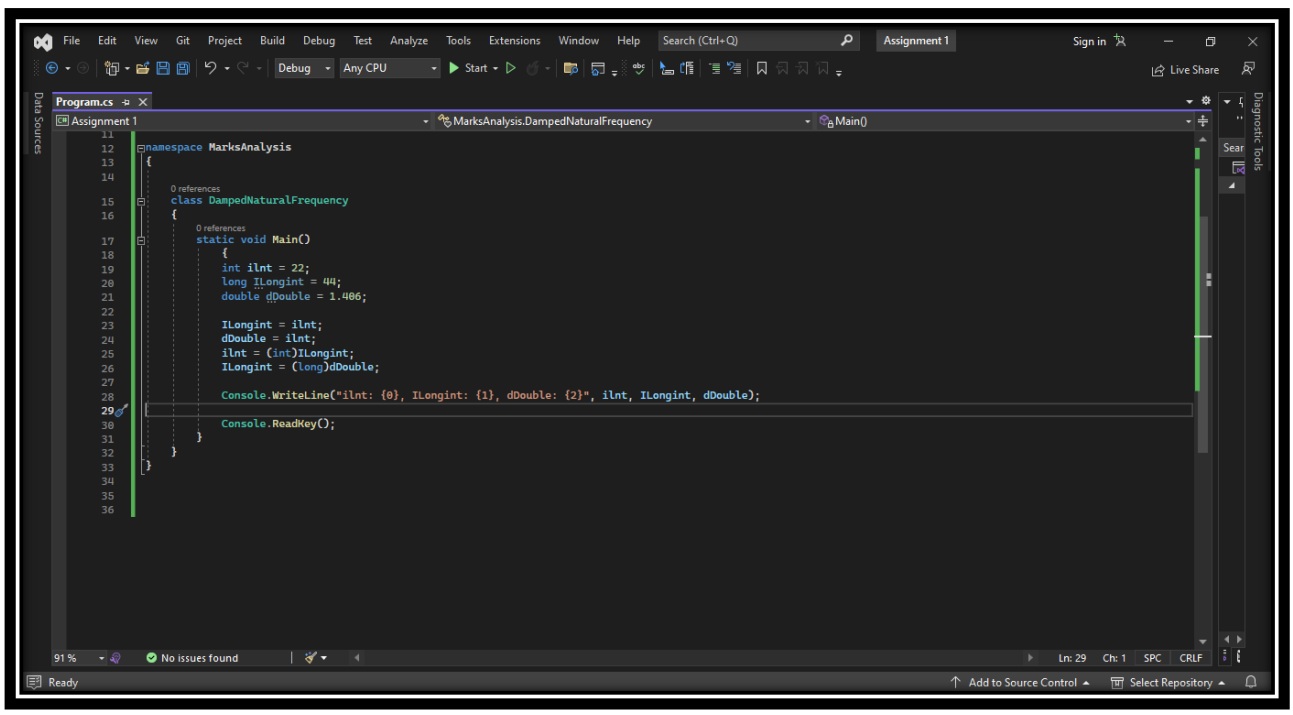
ilnt=[Longint; ILongint-dDouble; }

Solution:

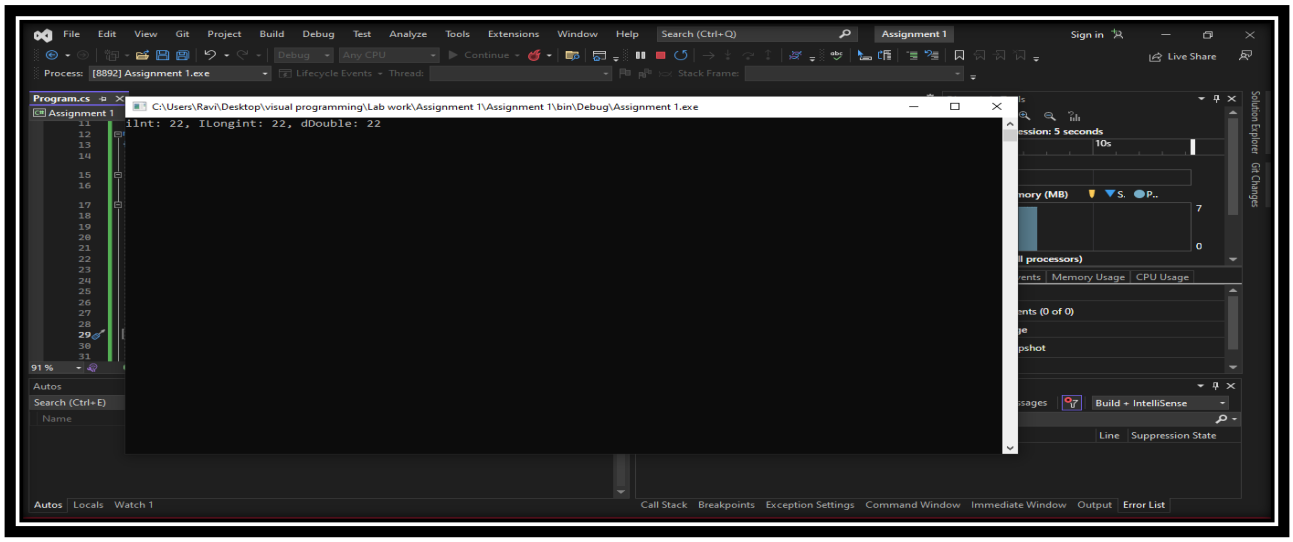
There are a few invalid conversions in the given program:

i. long ILongint 44; should be long ILongint = 44;. This is a syntax error, as there should be an equals sign between the variable name and the value being assigned.
ii. ilnt=[Longint; should be ilnt=ILongint;. This is a syntax error, as the variable names are not consistent with their declaration.
iii. ILongint-dDouble; should be ILongint = (long)dDouble;. This is an invalid conversion, as you cannot implicitly convert a double value to a long value. Instead, you need to explicitly cast the double value to a long value.

Correct code:



Output:

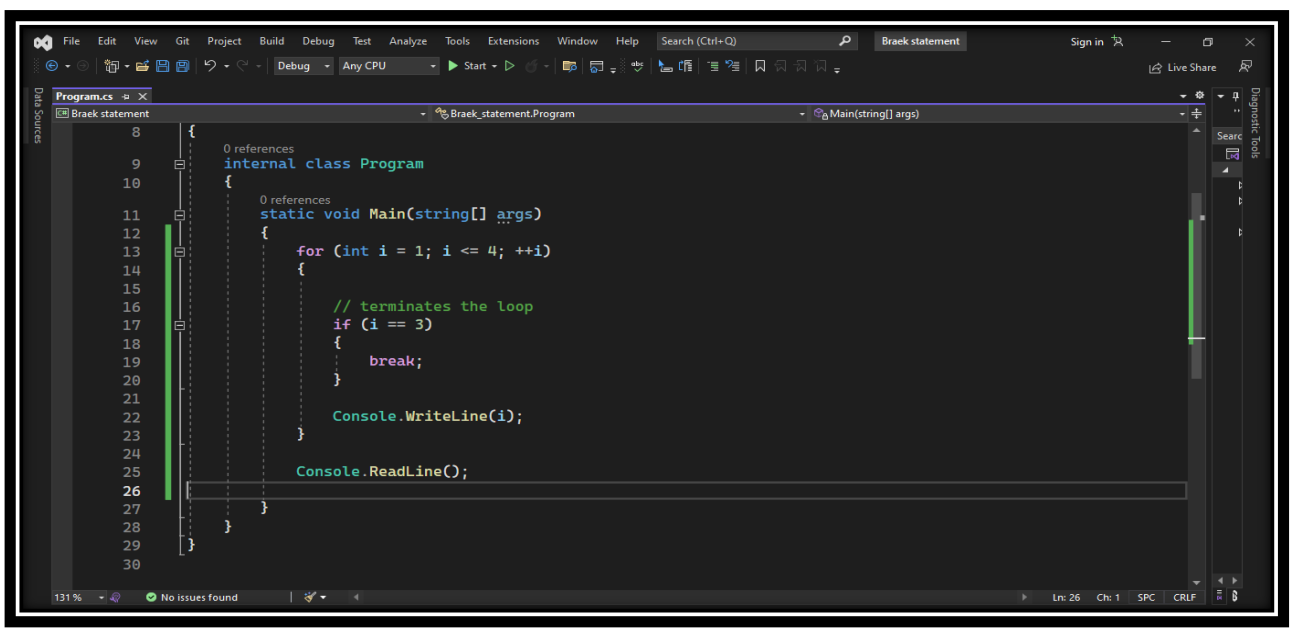


Program 15:

Demonstrate the typical use of the following jump statement:

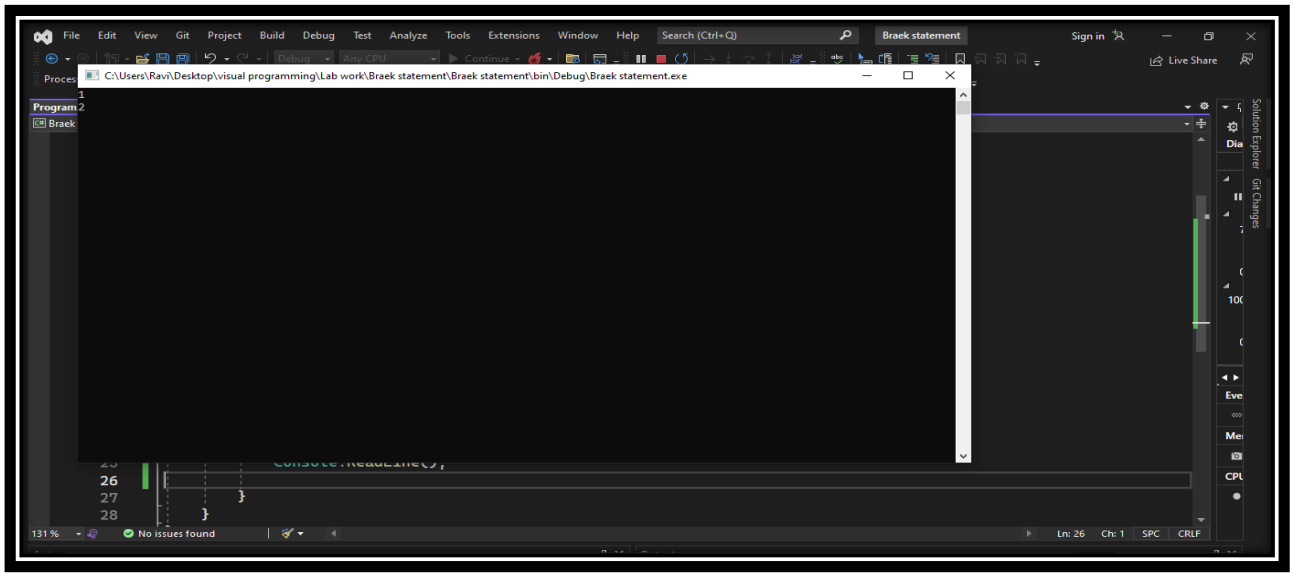
Break: break statement is used to jump out from any loop. As we know, loops iterate over a block of code until the test expression is false. However, sometime we may need to terminate the loop immediately without checking the test expression. In such cases, the break statement is used. The syntax of break statement is **"break".** For example:

Code:



```
{
    0 references
    internal class Program
    {
        0 references
        static void Main(string[] args)
        {
            for (int i = 1; i <= 4; ++i)
            {

                // terminates the loop
                if (i == 3)
                {
                    break;
                }

                Console.WriteLine(i);
            }

            Console.ReadLine();

        }
    }
}
```

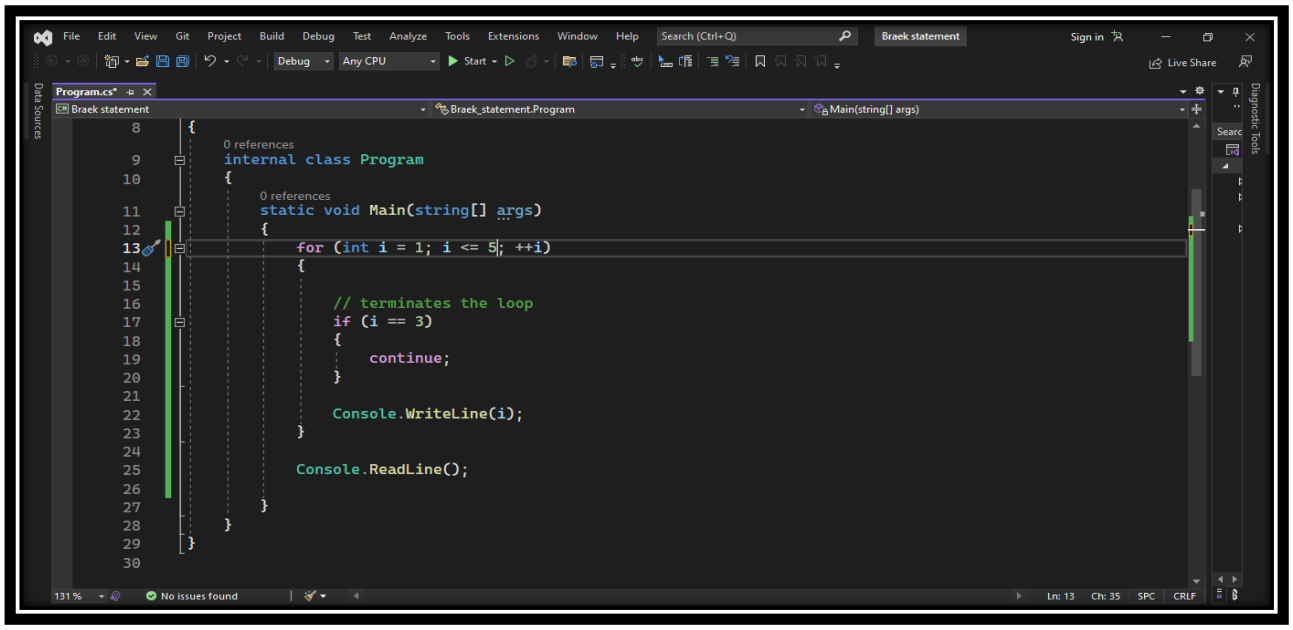
Output:



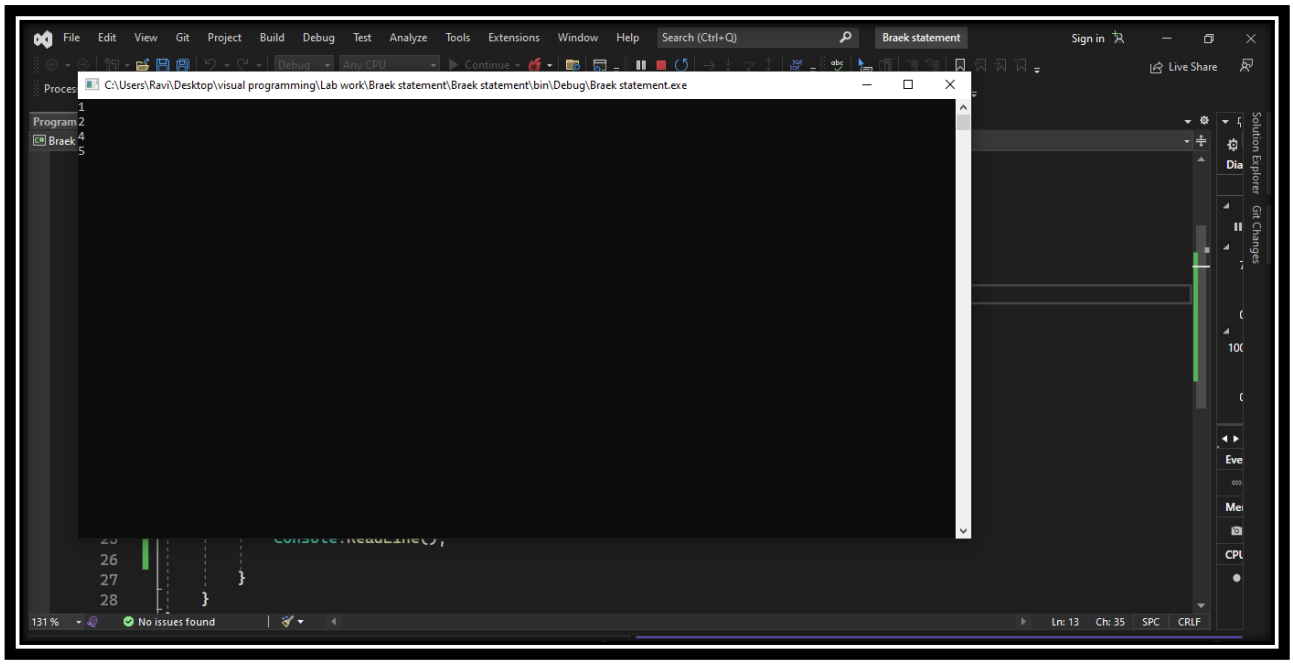
In the above program, the **for** loop run **4** times **i=1** to **4**. However, when **i** is equal to **3**, then break statement is encountered. Now, the loop is terminate suddenly. So, we only get **1** and **2** as output.

Continue: The continue statement breaks one iteration (in the loop), if a specified condition occurs, and continues with the next iteration in the loop.

Code:



Output:



In the above program, the **for** loop run **5** times **i=1** to **5**. However, when **i** is equal to **3**, then **continue** statement is executing and then skip the value of **3**. So, we only get **1,2,4** and 5 as output.

Goto: the goto statement transfer control to some other part of the program. For example,

```
goto label;
...
...
label:
   ...
   ...
```

Here, label is the identifier. When **goto label**: is encountered, then control of the program is transfer to **label:** then the code below **label:** is executed.

Code:

```
0 references
static void Main(string[] args)
{
// label
repeat:
    Console.WriteLine("Enter a number less than 10");
    int num = Convert.ToInt32(Console.ReadLine());

    if (num >= 10)
    {
        // transfers control to repeat
        goto repeat;
    }

    Console.WriteLine(num + " is less than 10");
    Console.ReadLine();

}
}
}
```

Output:

```
Enter a number less than 10
333
Enter a number less than 10
3
3 is less than 10
```

In the above program, we have a **goto** statement inside the if statement.

If the entered number is not less than 10, **goto repeat:** transfers the control of the code to repeat: Then, the code below **repeat:** is executed. The control of code will be transferred to the **repeat:** label unless the entered number is less than 10.