**M.Chidanand (19001218)**

**Day 1-exercise 14th March 2023**

**1.How to use single line and multi-line comments in C#?**

**Ans.**

Single Line Comments:

**i**t is used to comment a single line.   
**Syntax :**

// Single Line Comment

Eg:

//  To print output in  program //

        Console. WriteLine(“hello”);

Multiline Comments **:**

It is used to comment more than one line

**Syntax :**

/\* Multiline

Comment \*/

Eg:

/\*   The program will have integer and

                                   Print the output \*/

        String sname  = “chida”

        Console. WriteLine(sname);

**2. Explain different data types with an example?**

Ans:

There are different datatypes in c#.They are

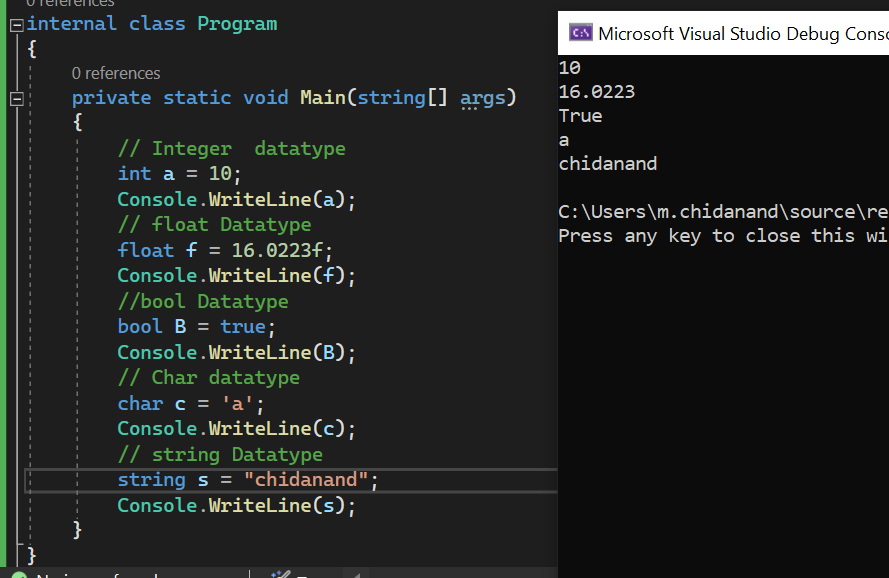
1**.int** – int means integer. The integer data type represents a positive whole number or its negative value

2.**float** -. This float data type represents a floating-point or decimal number

3.**bool** – bool stands for Boolean.This data type is used to store the values true and false.

4.**char** – char means character. This data type is used to store single alphabets or single characters

5.**string** - Strings are used for storing text/characters



**3.What are the rules for defining a variable?**

Rules for defining variables:

* variable can have alphabets, digits and underscore.
* variable name can start with alphabet and underscore only. It can't start with digit.
* No white space is allowed within variable name.
* C# is case-sensitive; thus, the names count and Count refer to two different variables
* Variable name must not be any reserved word or keyword e.g. char, float etc.

**4.How to convert a integer variable to a string variable. Give an example?**

Ans: for converting an integer to string in C#, We use the ToString() method.

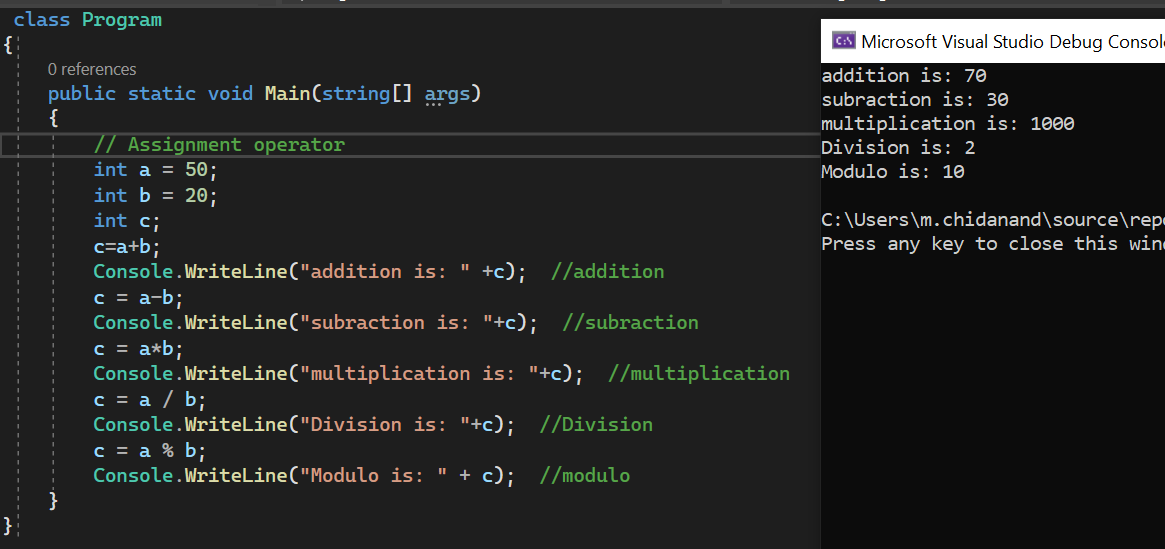
Text

Description automatically generated

**5. Explain different types of operators with an example?**

There are following types of operators to perform different types of operations in C# language.

1. **Arithmetic Operators** –[ Add(+),subract(-),multiply(\*),divide(/) ,modulo(%) ]



1. **Relational Operators**- [ Less than(<),less than or equal to(<=),greater than(>),greater than or equal to(>=),equal to(==),not equal to(/=)]

Text

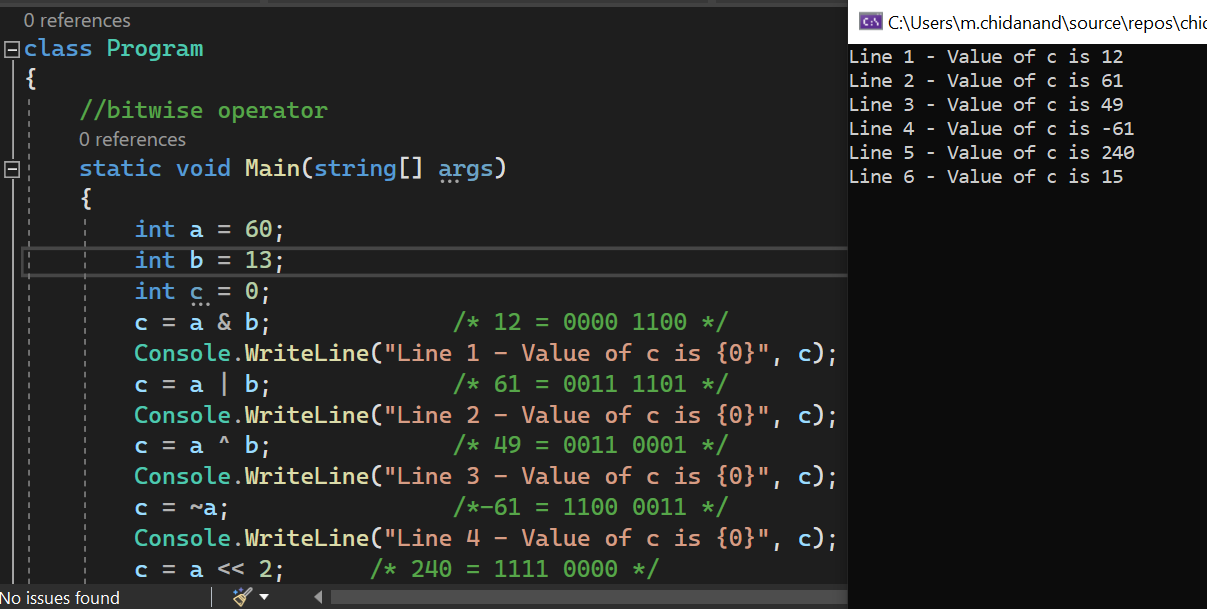
Description automatically generated

1. **Logical Operators-** [AND(&&),OR(||),NOT,(!)]

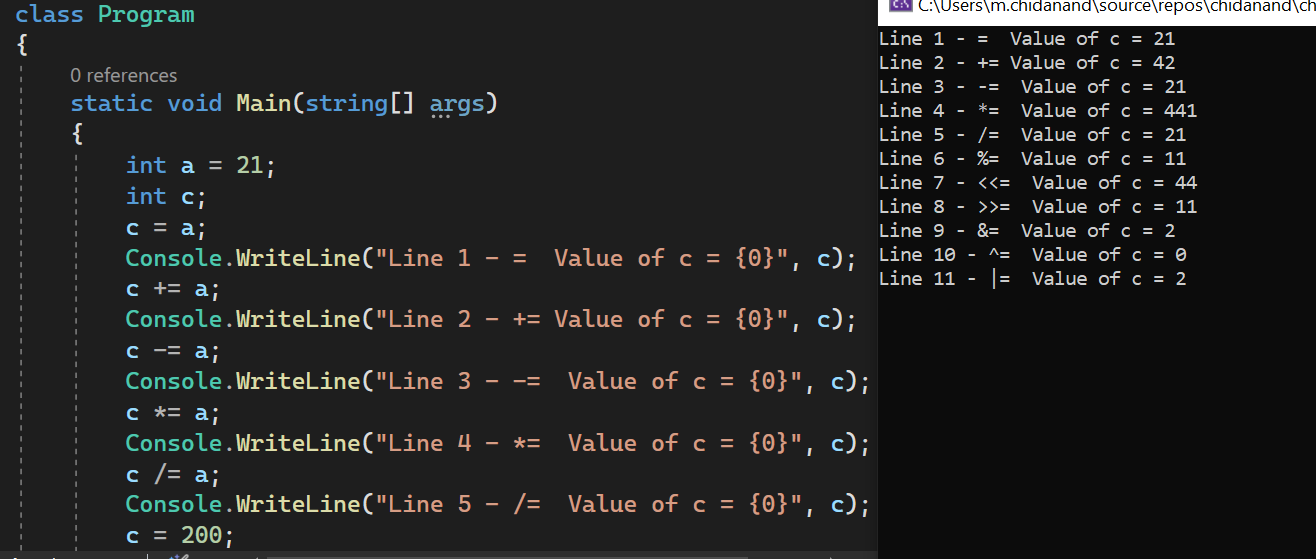
Text

Description automatically generated

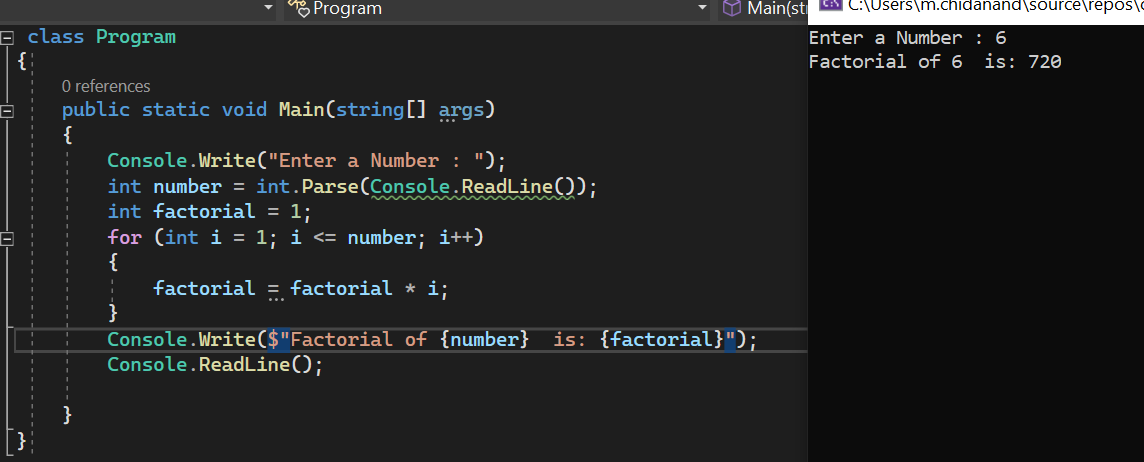
1. **Bitwise Operators**-[OR(|),AND(&),XOR(^),Complemental(~),Right shift (>>),left shift(<<)]



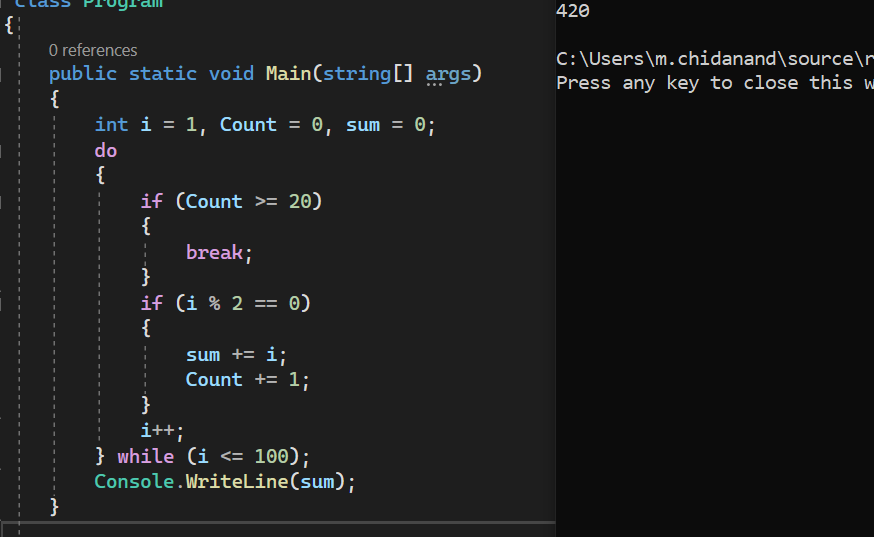
1. **Assignment Operators**- [(=),(+=),(-=),(\*=),(/=)(%=) ]



**6. Write a program to find the Factorial of a number using For Loop?**



**7. Write a program to sum the first 20 Even numbers from the 1 to 100 using Do While...Loop Statement?**

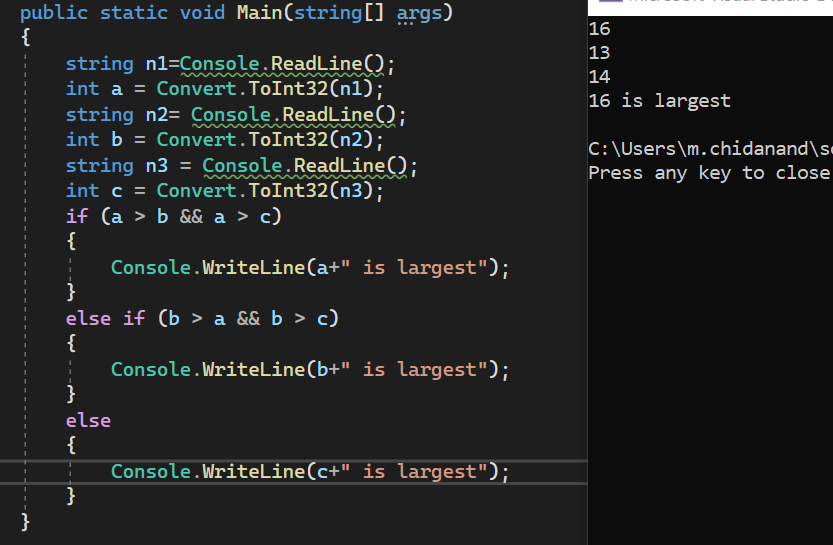


**8. Write a program to get the input number between 1 to 7 from user and give the output in day of the week. For Example: Get Input Number: 5, then the output should be FRIDAY.**

Text

Description automatically generated

**9. Write a C# Sharp program to find the largest of three numbers?**



1. **What is break, goto and continue statement in C#. Explain with an example?**

Ans:

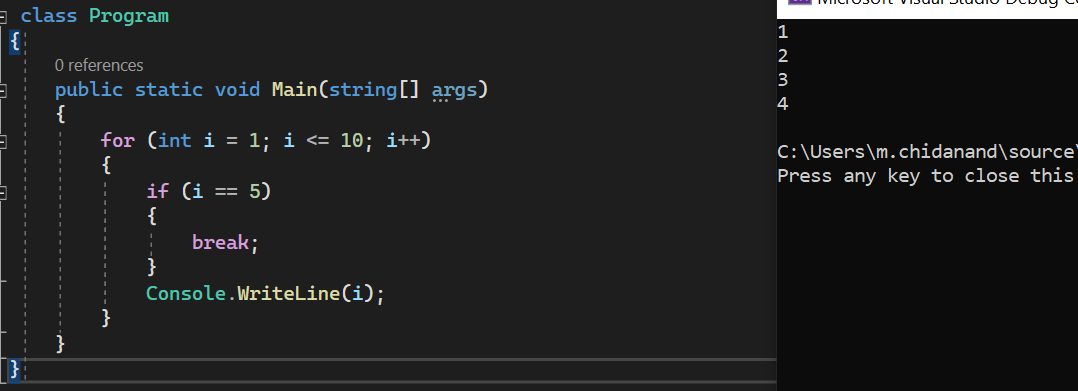
**Break statement:**

The C# break is used to break loop or switch statement. It breaks the current flow of the program at the given condition. In case of inner loop, it breaks only inner loop.

**Syntax**:

jump-statement;

break;

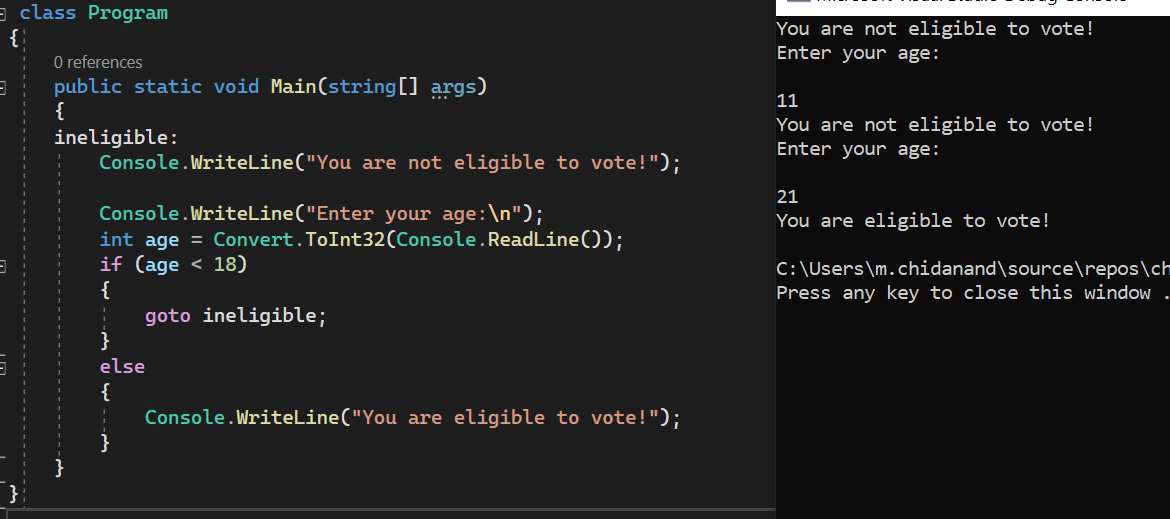


**Go-to Statement:**

goto statement is also known jump statement.

It is used to transfer control to the other part of the program & unconditionally jumps to the specified label.Used to transfer control from deeply nested loop or switch case label.

Eg:



**Continue Statement:**

The C# continue statement is used to continue loop.

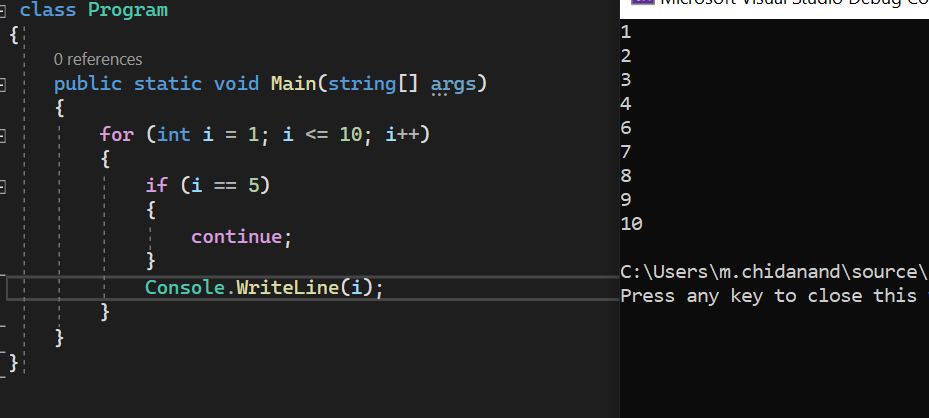
It continues the current flow of the program and skips the remaining code at specified condition

**Syntax:**

Jump-statement;

Continue;

Eg:



1. **Write a program to perform a simple calculation?  
   Get two numbers from User:  
   Enter the first Integer :10  
   Enter the second Integer :2  
     
   Provide the options to the user:  
   1-Addition.  
   2-Substraction.  
   3-Multiplication.  
   4-Division.  
   5-Exit.  
     
   Get the user option:  
   Input your choice :3  
     
   Display the result:  
   The Multiplication of 10 and 2 is: 20**

Ans:

class Calculator

{

static void Main()

{

Console.WriteLine("Enter the first Integer :");

int ninput1 = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter the second Integer :");

int ninput2 = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("1 - Addition.\n2 - Substraction.\n3 - Multiplication.\n4 - Division.\n5 - Exit.");

Console.WriteLine("Enter your choice");

int choice = Convert.ToInt32(Console.ReadLine());

int result = 0; switch (choice)

{

case 1:

result = ninput1 + ninput2;

Console.WriteLine("The Addition of " + ninput1 + " and " + ninput2 + " is " + result);

break;

case 2:

result = ninput1 - ninput2;

Console.WriteLine("The Subtraction of " + ninput1 + " and " + ninput2 + " is " + result);

break;

case 3:

result = ninput1 \* ninput2;

Console.WriteLine("The Multiplication of " + ninput1 + " and " + ninput2 + " is " + result);

break;

case 4:

result = ninput1 / ninput2;

Console.WriteLine("The division of " + ninput1 + " and " + ninput2 + " is " + result);

break;

case 5:

Console.WriteLine("Exit");

break;

default:

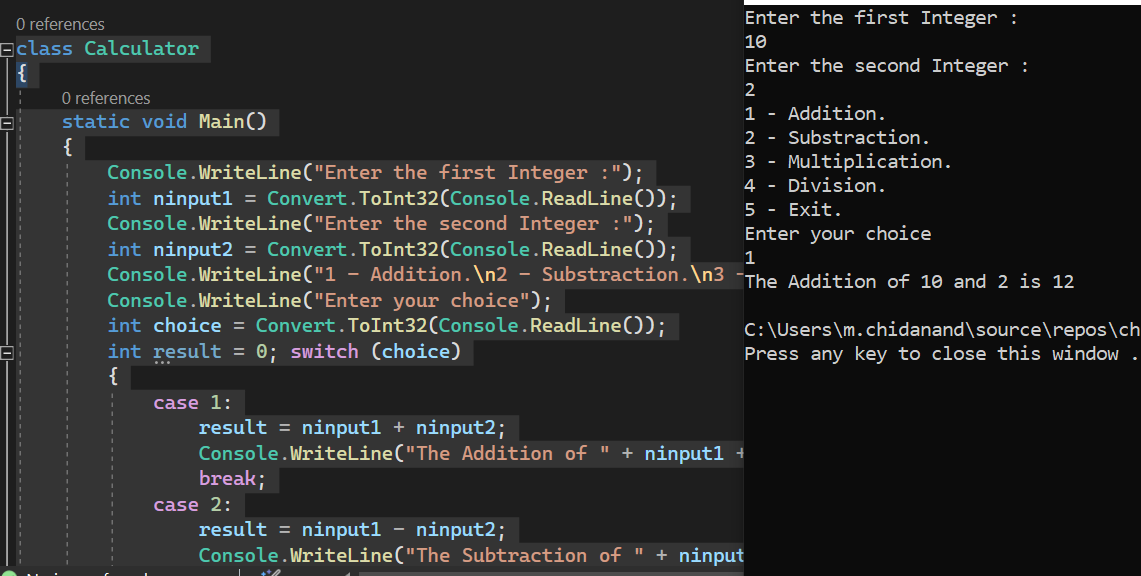
Console.WriteLine("Enter valid input");

break;

}

}

}



1. **Explain call by value with an example?**

Ans: value-type parameters are that pass a copy of original value to the function rather than reference. It does not modify the original value. A change made in passed value does not alter the actual value.

Eg:

class Program

{

public void Display(int a)

{

a += a;

Console.WriteLine("Value inside the function: " + a);

}

static void Main(string[] args)

{

int a = 100;

Program xyz = new Program();

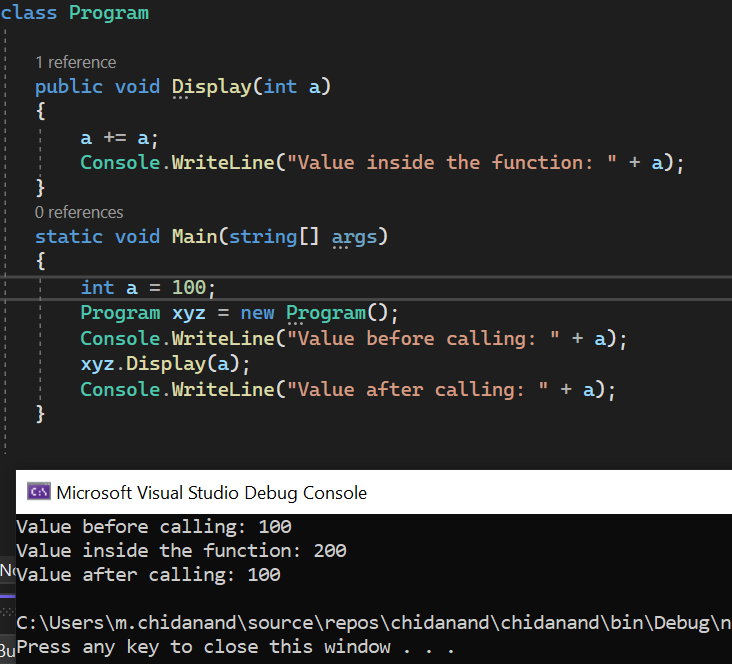
Console.WriteLine("Value before calling: " + a);

xyz.Display(a);

Console.WriteLine("Value after calling: " + a);

}

}



1. **Explain call by reference with an example?**

Ans: C# provides a **ref** keyword to pass argument as reference-type. It passes reference of arguments to the function rather than copy of original value. The changes in passed values are permanent and **modify** the original variable value.

Eg:

using System;

namespace callByReference

{

class Program{

public void Show(ref int val)

{

val += val;

Console.WriteLine("Value inside the function: " + val);

}

static void Main(string[] args)

{

int val = 100;

Program program = new Program();

Console.WriteLine("Value before calling: " + val);

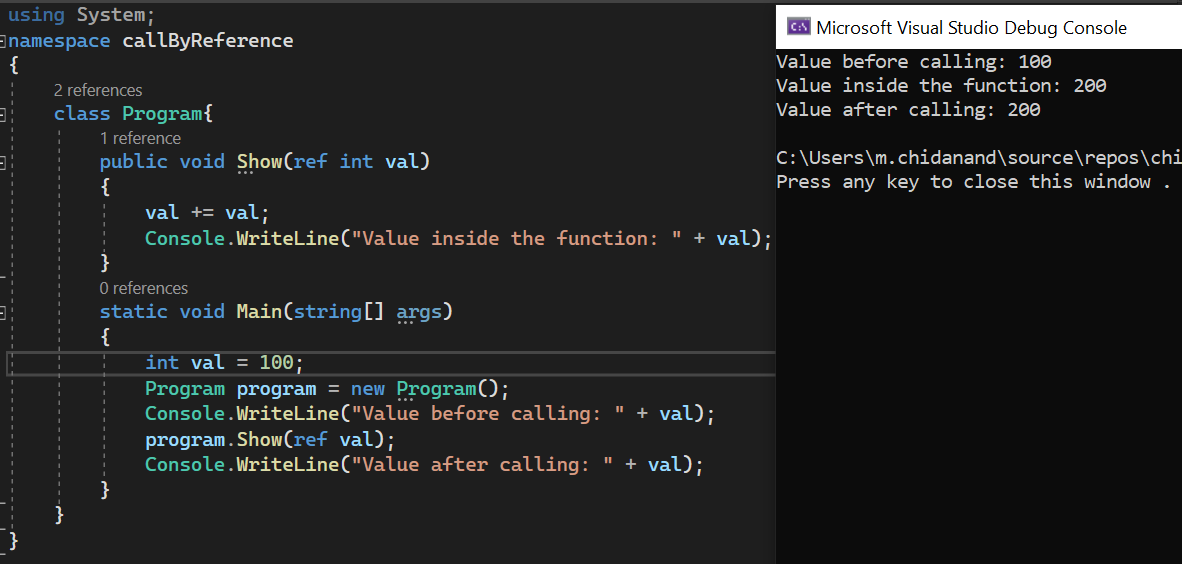
program.Show(ref val);

Console.WriteLine("Value after calling: " + val);

}

}

}



1. **What is Out parameter and when it is used? Give an example?**

Ans:a C# provides **out** keyword to pass arguments as out-type. It is like reference-type, except that it does not require variable to initialize before passing. We must use **out** keyword to pass argument as out-type. It is useful when we want a function to return multiple values.

Eg:

using System;

namespace OutParameter

{

class Program

{

// User defined function

public void Show(out int val) // Out parameter

{

int square = 5;

val = square;

val \*= val;

}

// Main function

static void Main(string[] args)

{

int val = 50;

Program program = new Program(); // Creating Object

Console.WriteLine("Value before passing out variable " + val);

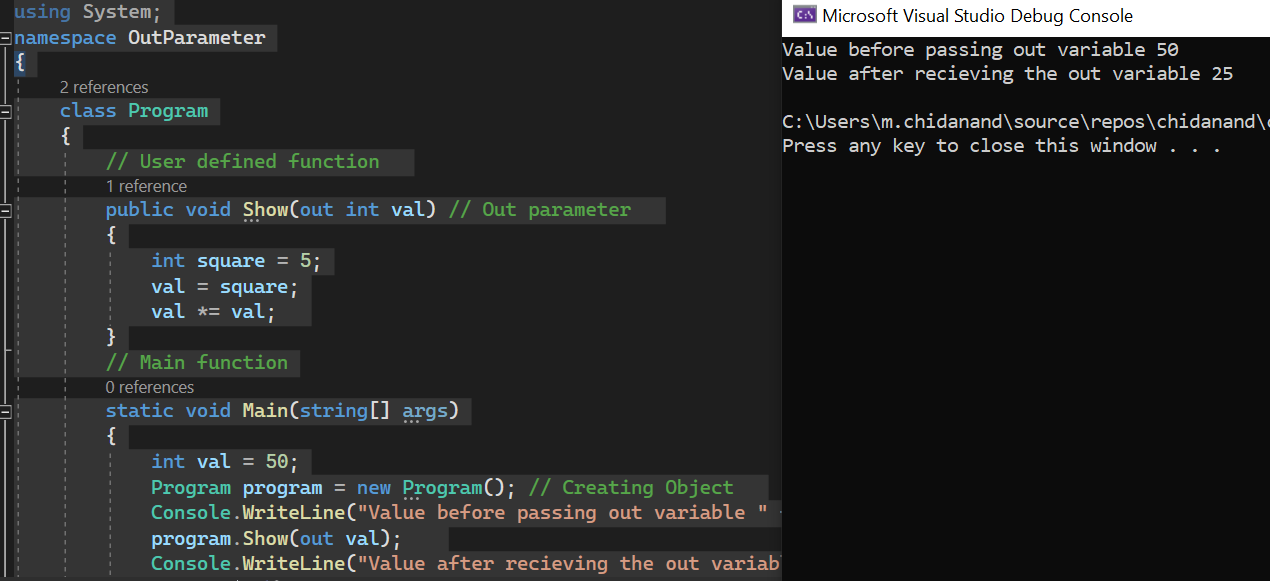
program.Show(out val);

Console.WriteLine("Value after recieving the out variable " + val);

}

}

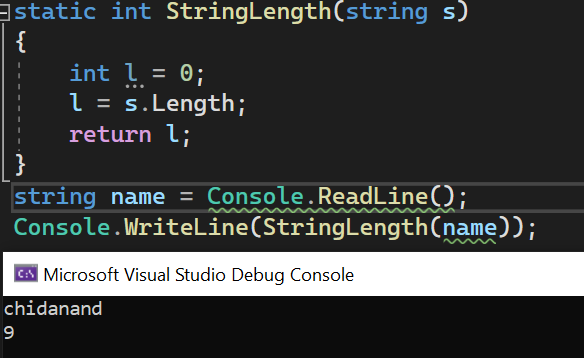
}



1. **Write a function which accepts a string as input and returns the number of characters in the string?**

Ans:

static int StringLength(string s)  
{  
    int l = 0;  
    l = s.Length; return l;  
}  
string name = Console.ReadLine();  
Console.WriteLine(StringLength(name));



1. **Explain Named Arguments in C# methods with an example?**

**Ans:***Named arguments* enable you to specify an argument for a parameter by matching the argument with its name rather than with its position in the parameter list.

public class NamedArgumentsExample

{  
        static string GetFullName(string firstName, string lastName)  
        {  
            return firstName + " " + lastName;  
        }  
        public static void Main(string[] args)  
        {  
            string fullName1 = GetFullName("Rahul", "Kumar");  
            string fullName2 = GetFullName(firstName: "Rahul", lastName: "Kumar");   
            string fullName3 = GetFullName(lastName: "Rahul", firstName: "Kumar");   
            Console.WriteLine(fullName1);  
            Console.WriteLine(fullName2);  
            Console.WriteLine(fullName3);  
        }  
    }  
}

Text

Description automatically generated

1. **How do you declare multiples variables of same type with comma-separated?**

**Ans:**To declare more than one variable of the **same type**, use a comma-separated list.

Eg:

int x=4, y=5,z=50;

Console.writeline(x+y+z);

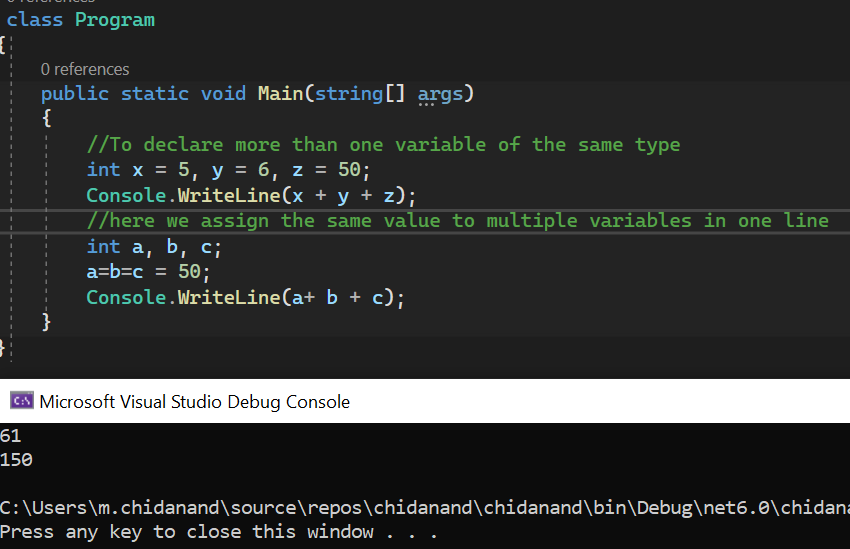
You can also assign the **same value** to multiple variables in one line

Eg:

int a,b,c;

a=b=c=50;

Console.writeline(a+b+c);



1. **Explain Implicit and Explicit casting with an example?**

**Ans:**

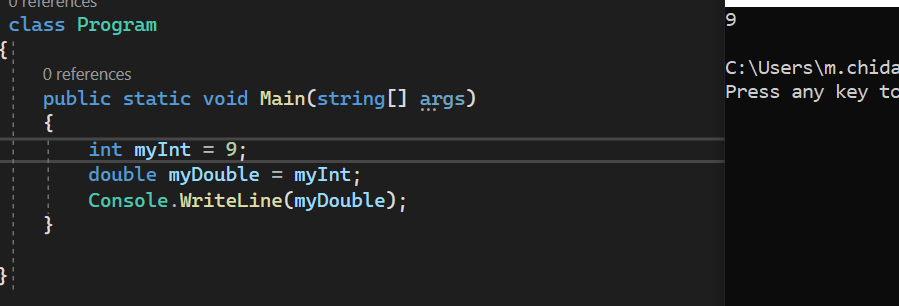
**implicit casting** - conversion of smaller data type to the larger type size.char -> int -> long -> float -> double.

Eg:

int myInt = 9;

double myDouble =myInt;

Console.WriteLine(myDouble);



**Explicit casting** - conversion of larger data type to the smaller type size.

Eg:

double myDouble = 9.78;

int myInt = (int) myDouble;

Console.WriteLine(myInt)

