C# .NET

C# Introduction

- It Was developed by the Microsoft Corporation.
- It is high level programming language.
- It is an object oriented programming language.
- It is case sensitive programming language.
- It extension .cs.
- It is a very easy programming language.
- It belongs to C family.
- It is used to develop various types applications.
- Console,Windows,Web and Mobile applications.

C# Features

- Implicitly typed local variables
- Object and collection initializers
- Auto-Implemented properties
- Anonymous types
- Extension methods
- Query expressions
- Lambda expressions
- Expression trees
- Partial Methods

C# Program Structure

Predefined Namespace
User defined Namespace
    Class Declaration
        Data types
        Methods.

Structure Definitions

Namespace

It is set of classes,interfaces,enumerations,deligates etc….

Class

It is the collection of datatypes and methods.

C# Program Structure

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Test {
   class Program
   {
      static void Main(string[] args)
      {

      }
   }
}
```

Predefined Namespaces

User defined Namespace

Class Declaration

Method

Document Area

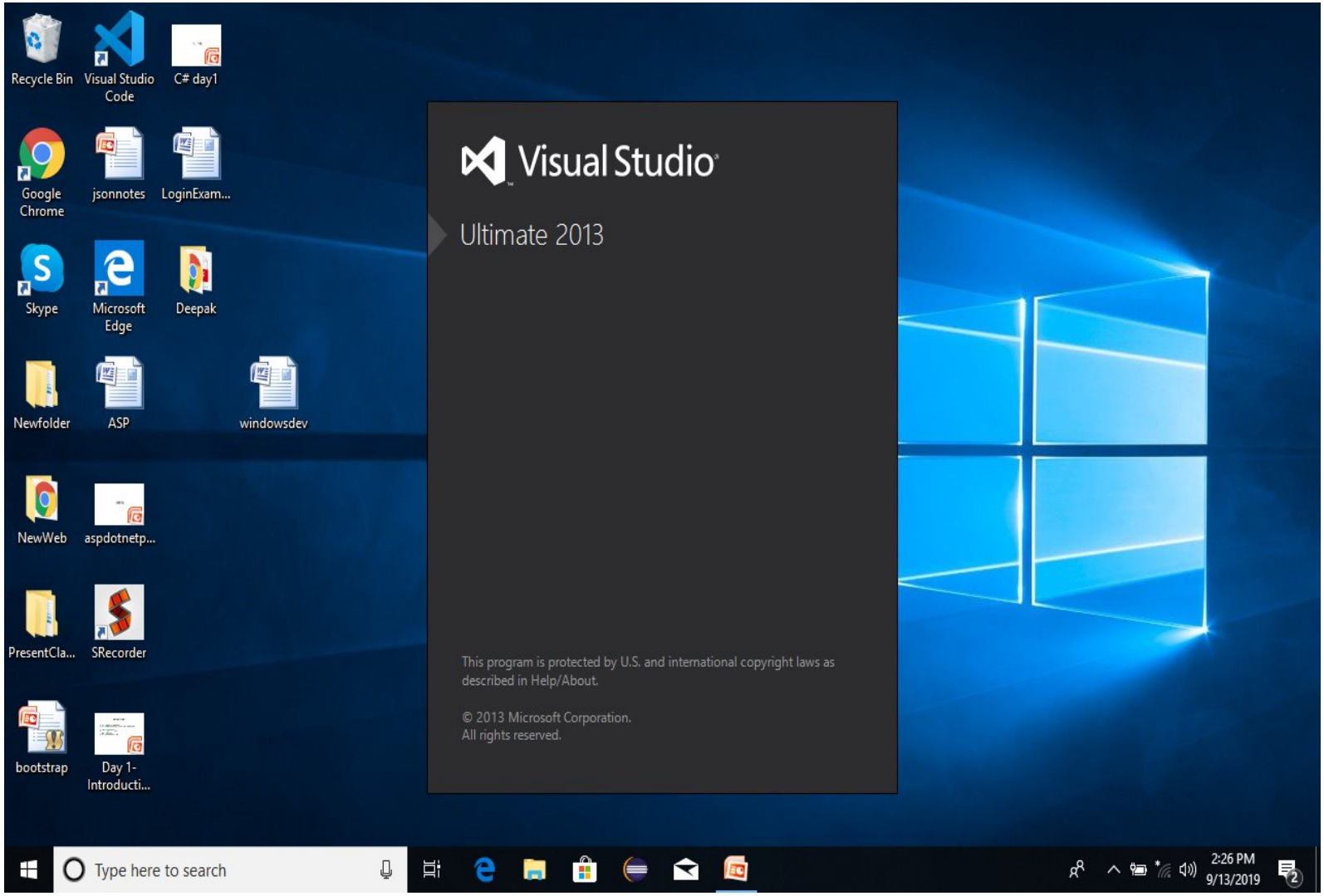
How to develop a program in C#?

1. Open the project
2. Code the program using C#.
3. Execute using Ctrl+F5.
4. These Steps will be repeated for all the programs.

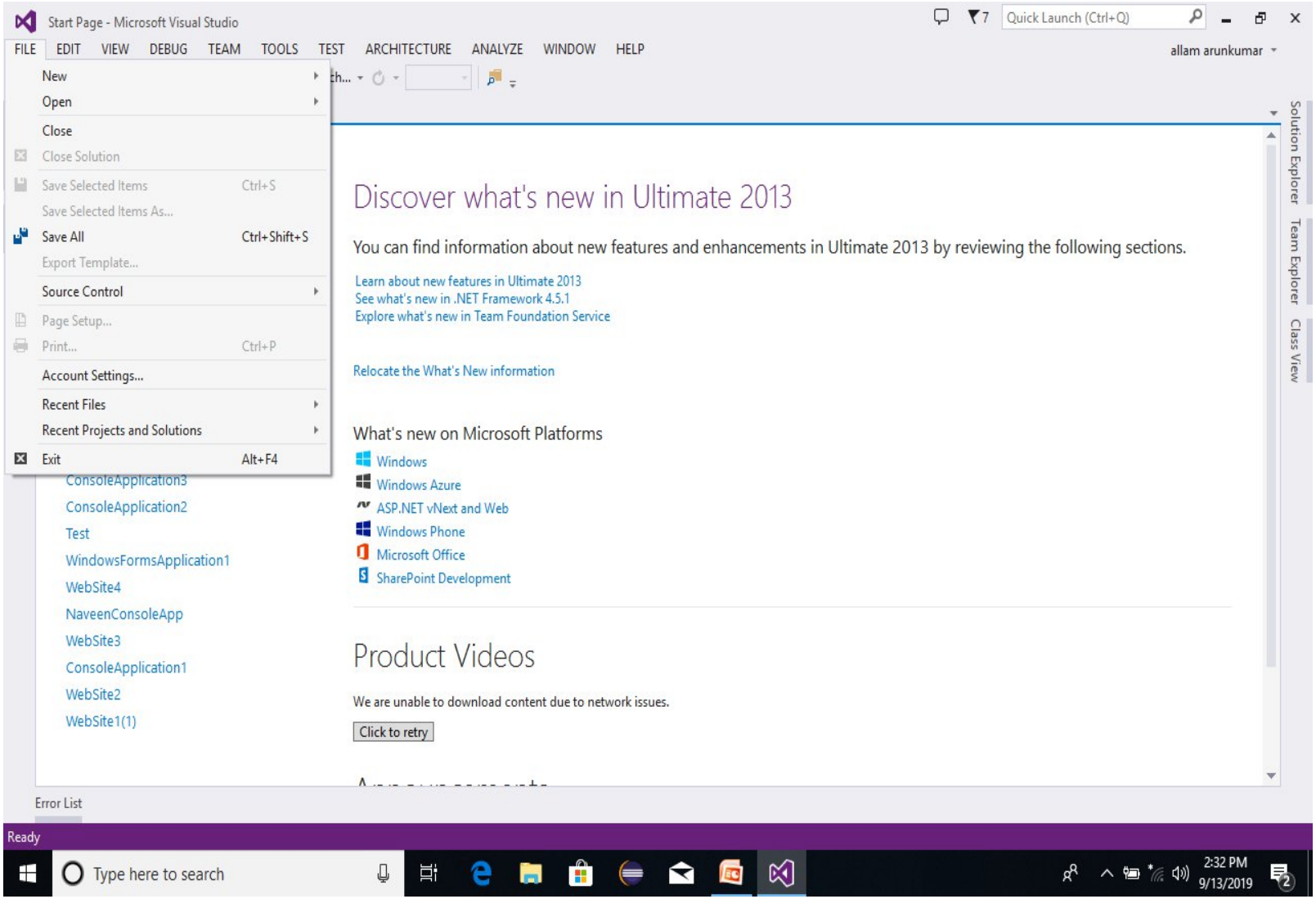
Visual Studio Introduction

1. It is an IDE(Integrated Development Environment).
2. It is used develop software , web and Mobile applications.
3. It is Introduced by Microsoft.
4. In order to work with any project in Visual Studio tool we need know about Solution Explorer and Server Explorer .
5. Solution Explorer in which project and its respect files can be added.
6. Sever Explorer in which Databases can deal the with the DataBases.

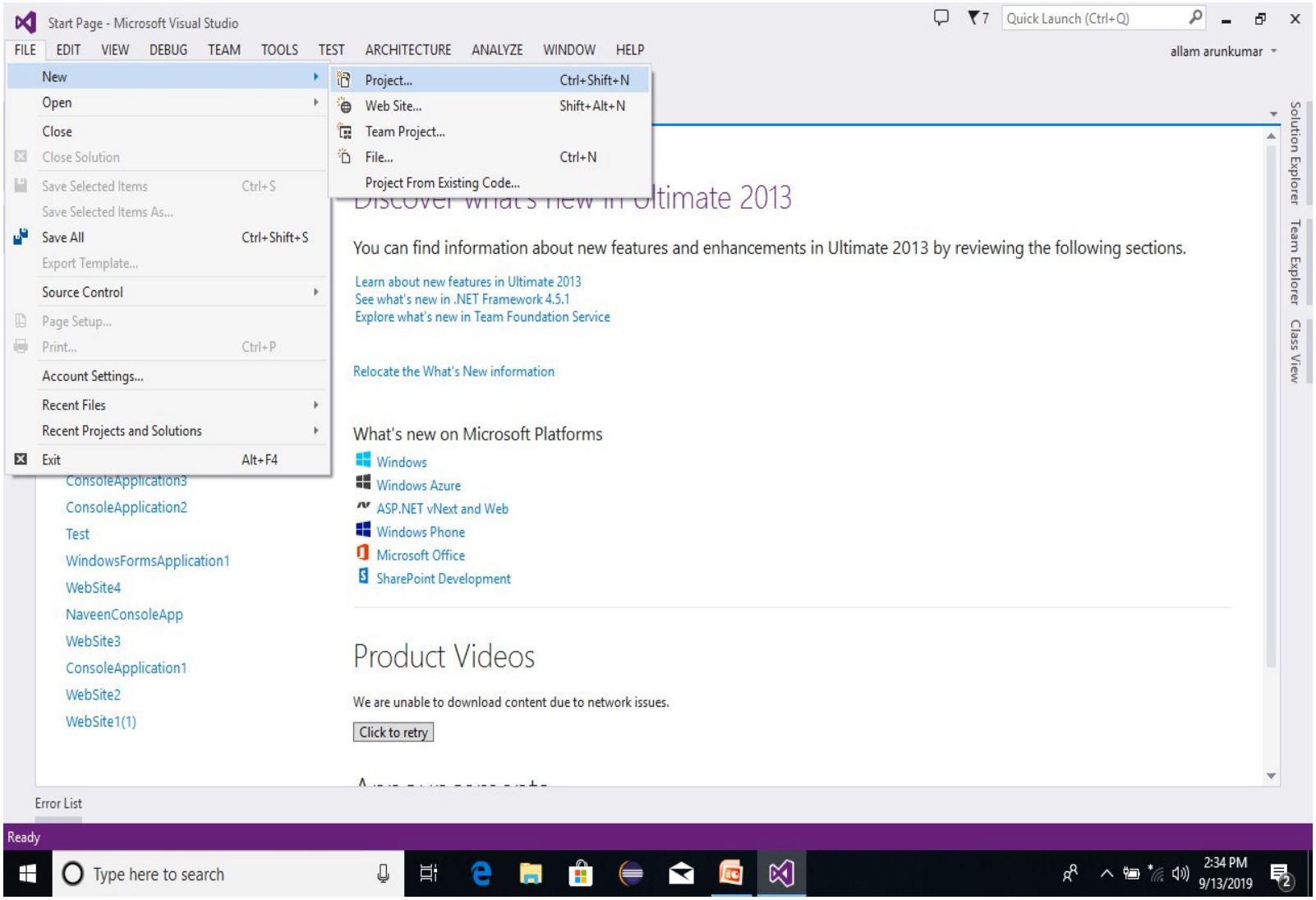
Creating new project in VS



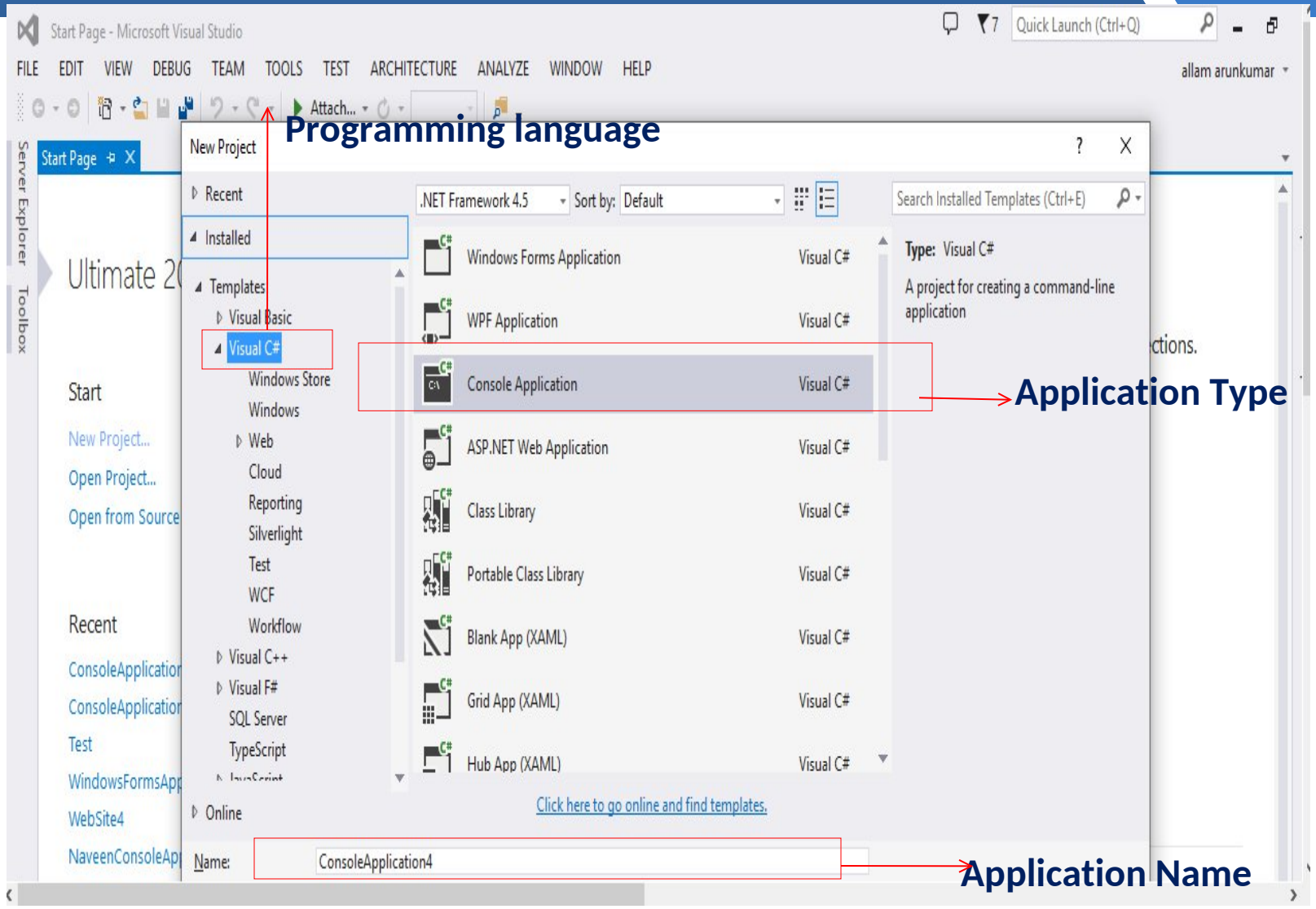
Creating new project in VS



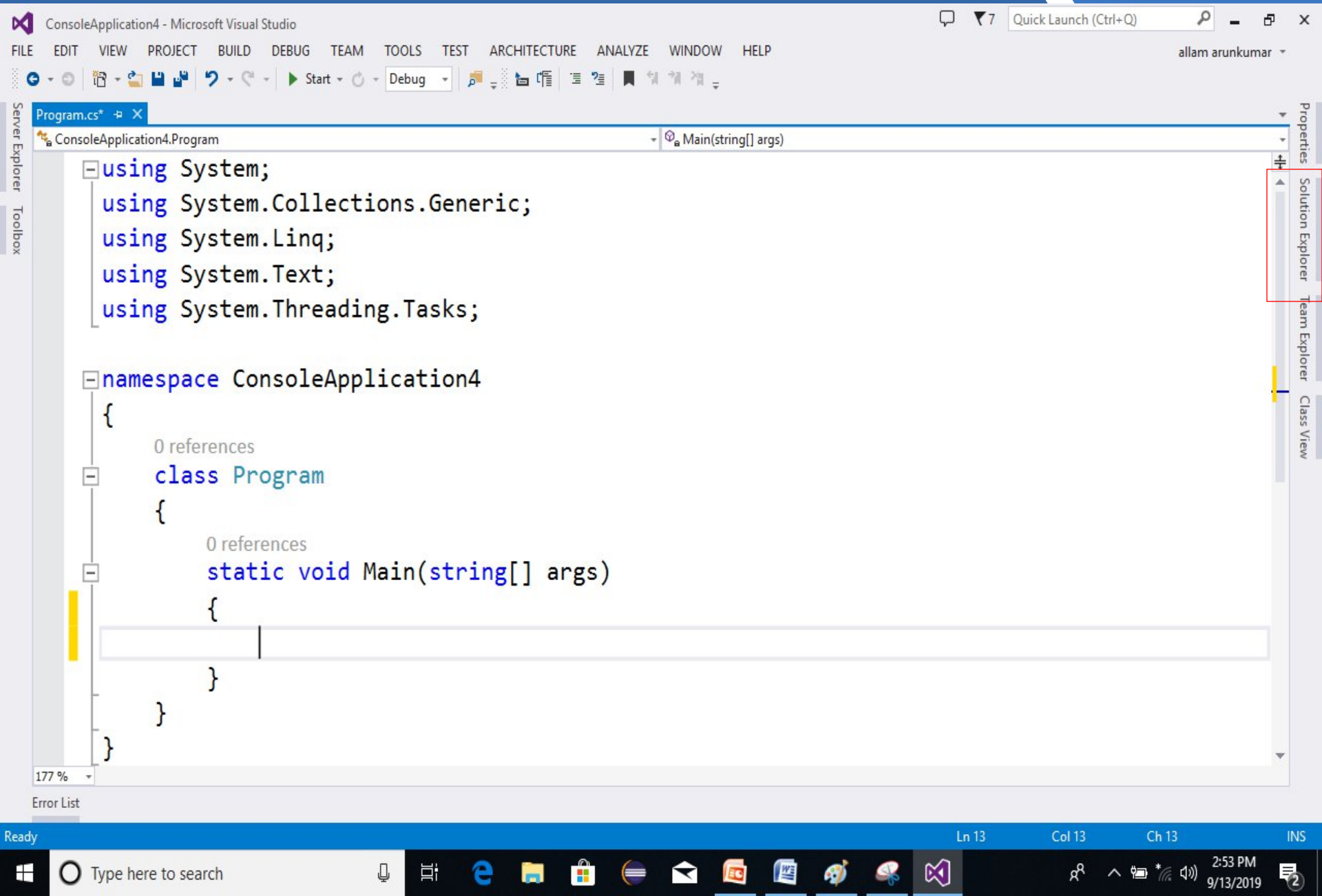
Creating new project in VS



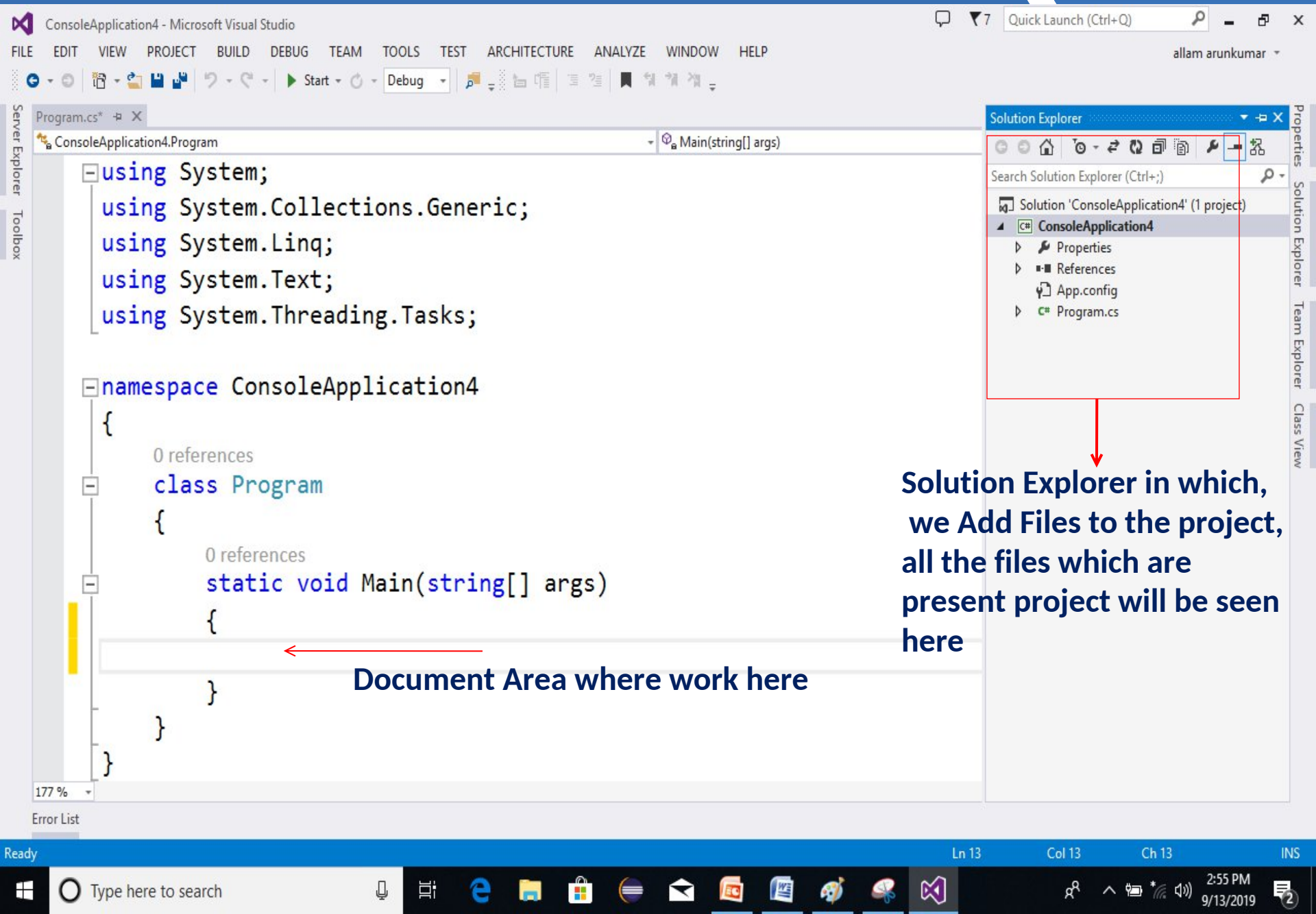
Creating new project in VS



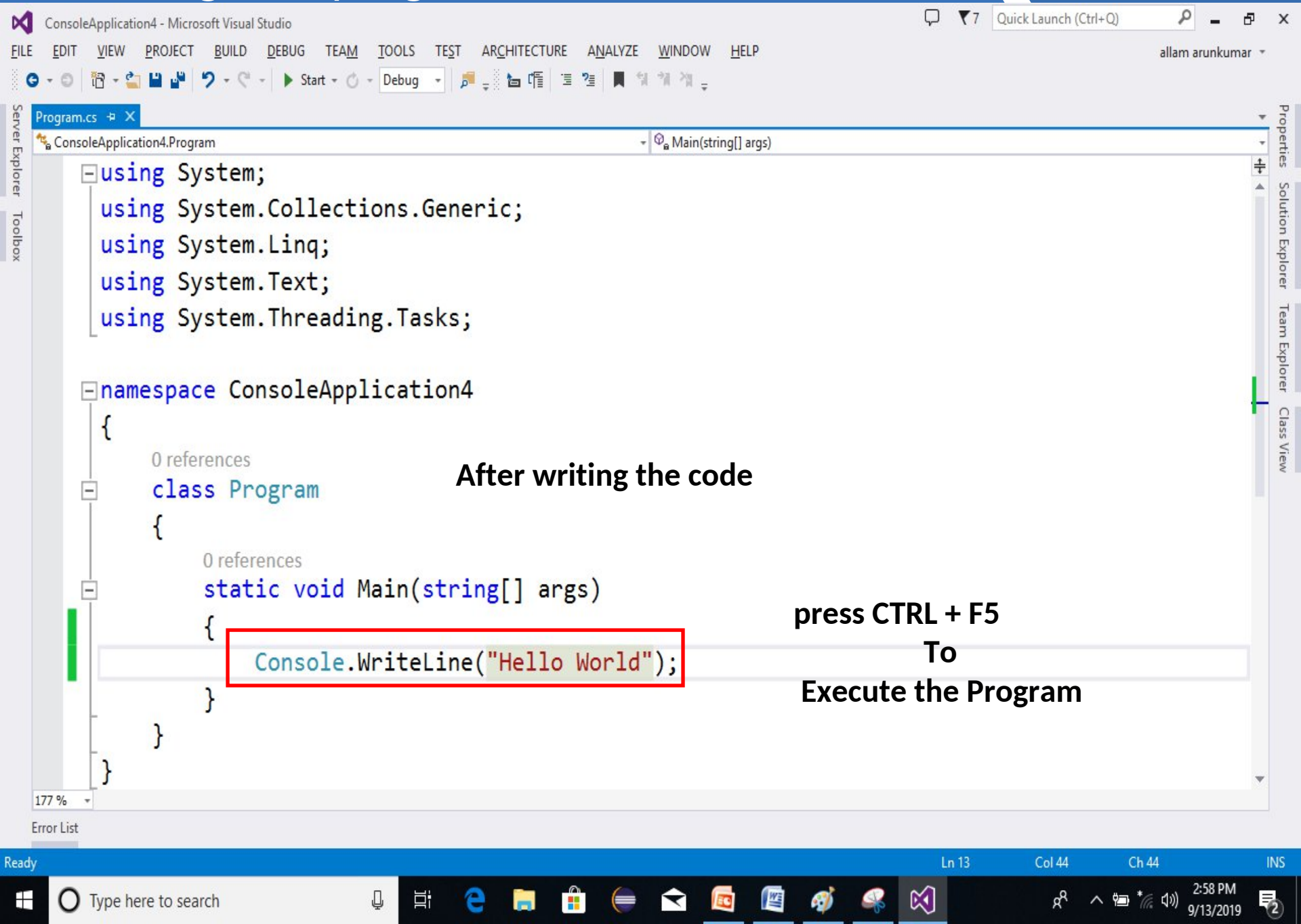
Coding new program in VS



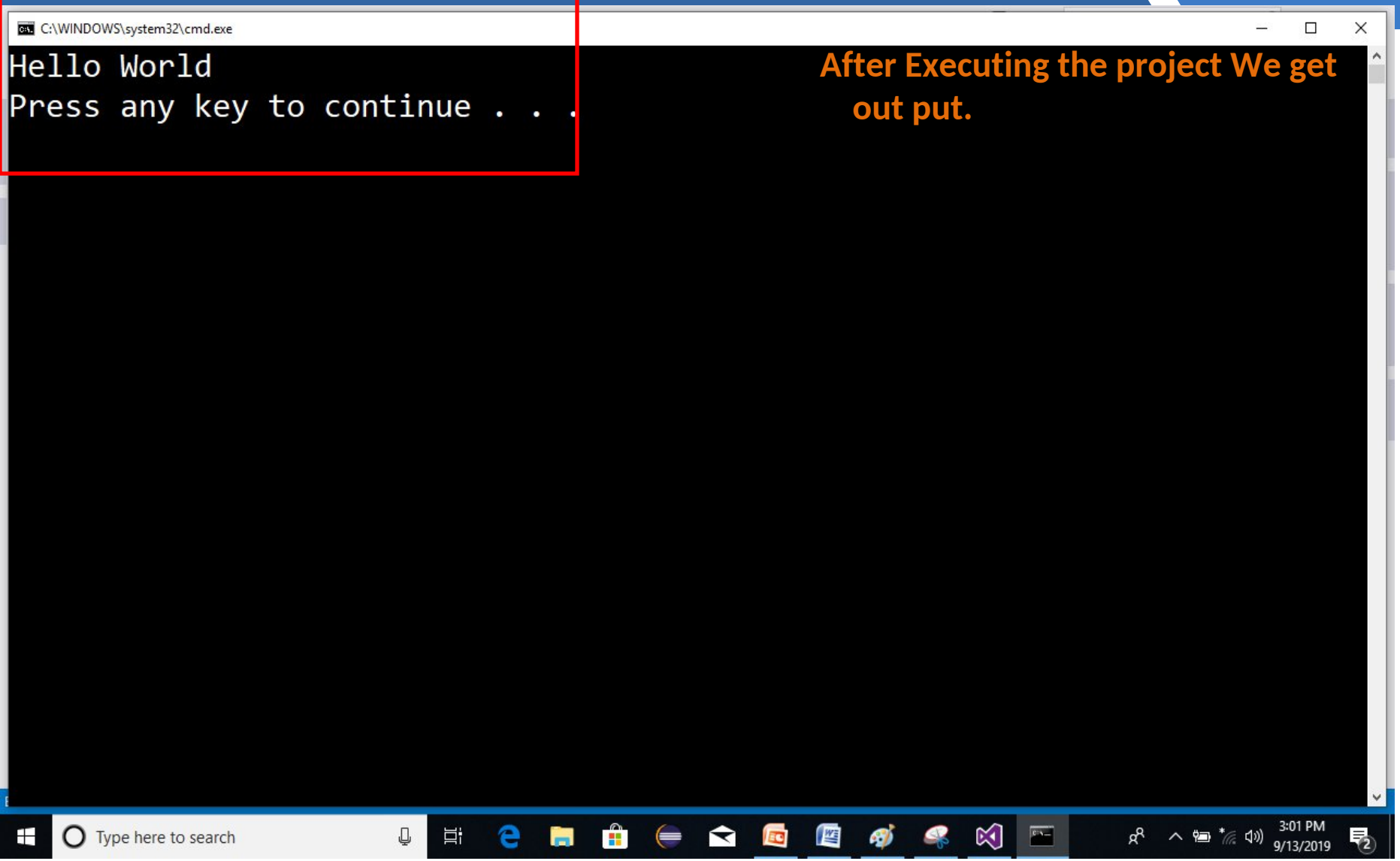
Coding new program in VS



Coding new program in VS



Coding new program in VS



Data Types in C#

- char 1 byte
- int 4 byte
- float 4 byte
- double 8 byte
- decimal 16 byte
- String
- class

Input and Out Put Methods

Out put Methods

These methods can be used to display the statements and values inside the console App.

They are

Console.WriteLine()

Console.Write()

Input and Out Put Methods

Input Methods

These are used to read values from the console.

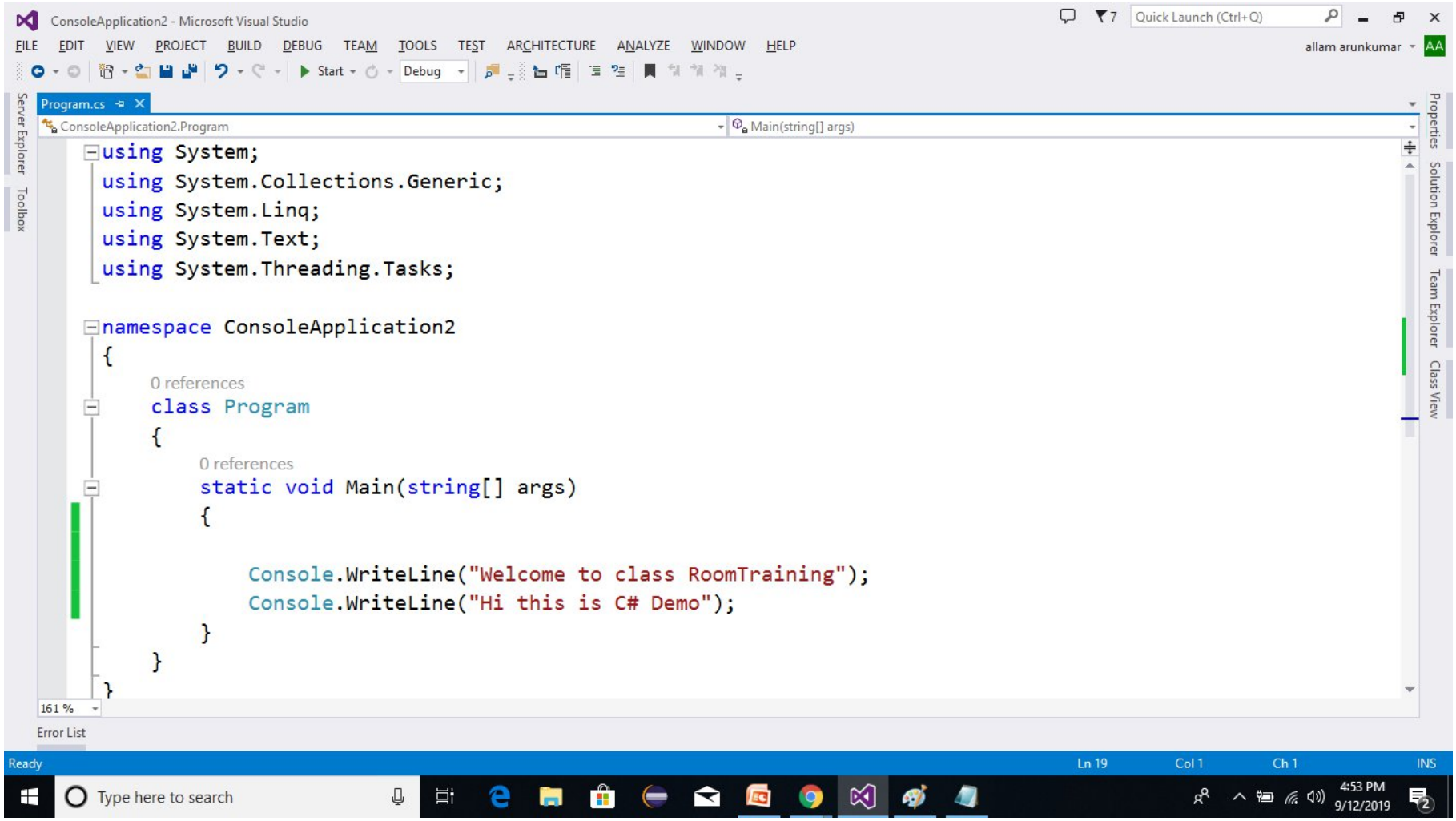
Tyey are

Console.ReadLine()

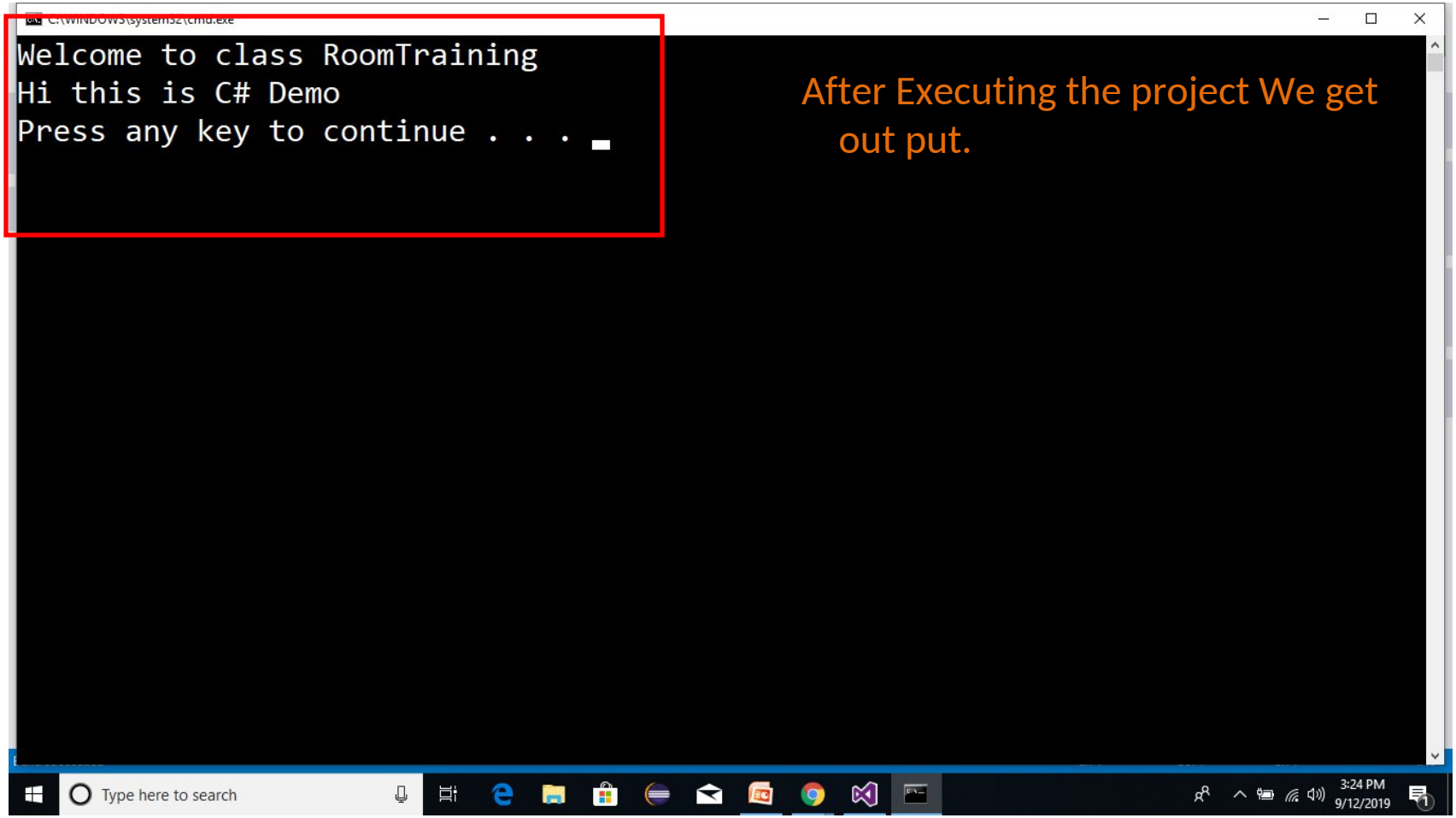
Console.Read()

Console.ReadKey()

Example on out put methods



Coding new program in VS



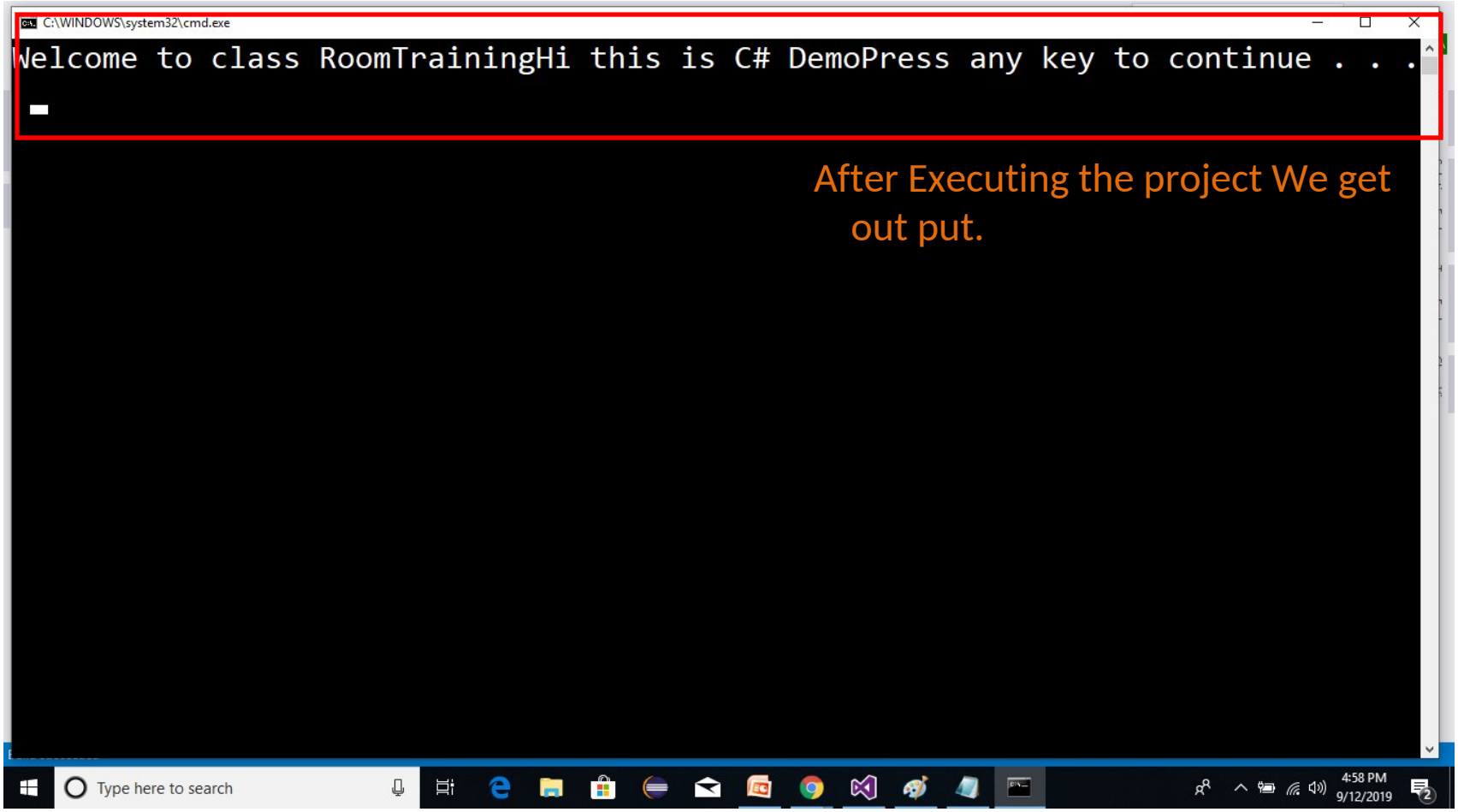
Example on out put methods

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApplication2
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {

            Console.Write("Welcome to class RoomTraining");
            Console.Write("Hi this is C# Demo");
        }
    }
}
```

After Executing VS



Example on Data Types with Out put methods

```
static void Main(string[] args)
      {
         int   a  = 10;
         float  x  =  4.5f;
         double  d  =  7.8;
         string s  =  "Welcome";
         decimal  m  =  5.6m;
         Console.WriteLine("{0}  {1}    {2}    {3}  {4}",a,x,d,s,m);
      }
```

Input and Out Put Methods

- Input Methods
  - These are used to read values from the console.
  - Tyey are
    - Console.ReadLine()
    - Console.Read()
    - Console.ReadKey()

Input and Out Put Methods

- Console.ReadLine()
  It is used to read a string
- Int.Parse(Console.ReadLine())
  It is used to Read Integer Value
- Float.Parse(Console.ReadLine())
  It is used to Read Float value.
- Double.Parse(Console.ReadLine())
  It is used to Read double value.
- Bool.Parse(Console.ReadLine())
  It is used to boolean value
- Console.Read()
  It used to Read ASCII Value of the given Character.
- Console.ReadKey()
  It is used to read a key from the console

Reading values

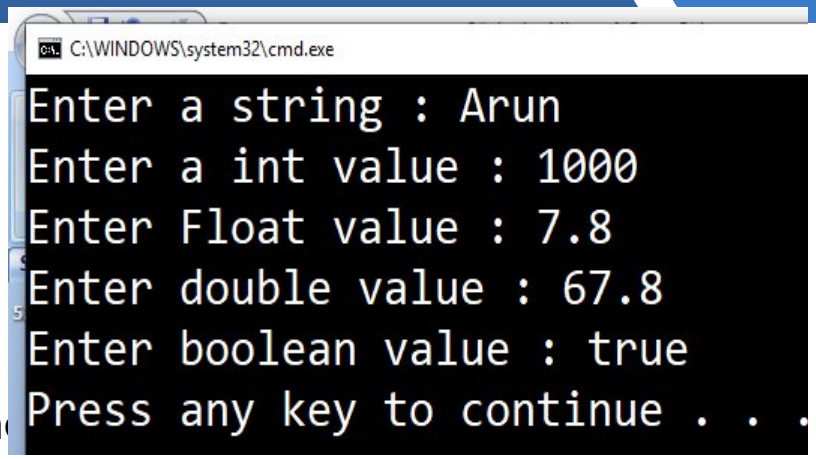
```
static void Main(string[] args)
    {
        Console.Write("Enter a string : ");
        string s = Console.ReadLine();

        Console.Write("Enter a int value :
        int a = int.Parase(Console.ReadLin

        Console.Write("Enter Float value : ");
        float f =float.Parse(Console.ReadLine());

        Console.Write("Enter double value : ");
        double d =double.parse(Console.ReadLine());

        Console.Write("Enter boolean value : ");
        bool b =bool.Parse( Console.ReadLine());
    }
```



Operators
In
C#

Operators in C#

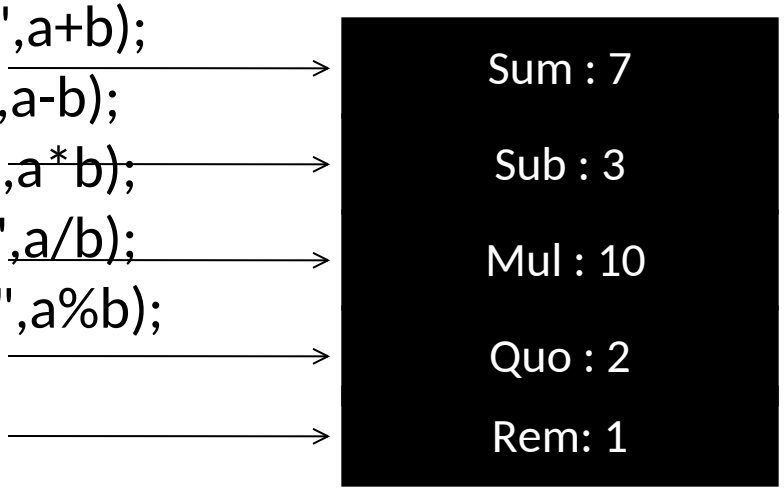
- Arithmetic operators .
- Relational operators.
- Logical Operators.
- Assignment operators.
- conditional operators.
- Increment and Decrement operators.
- Bitwise Operators.// home work

Arithmetic operators .

- + -> add
- - ->sub
- * ->mul
- / ->div-->quo 5/2 ---> 2
- % ->mode->Rem 5%2 --->1

Arithmetic operators .

```
static void Main(string[] args)
    {
        int a = 5;
        int b = 2;
        Console.WriteLine("Sum:{0}",a+b);
        Console.WriteLine("Sub:{0}",a-b);
        Console.WriteLine("Mul:{0}",a*b);
        Console.WriteLine("Quo:{0}",a/b);
        Console.WriteLine("Rem:{0}",a%b);
    }
```

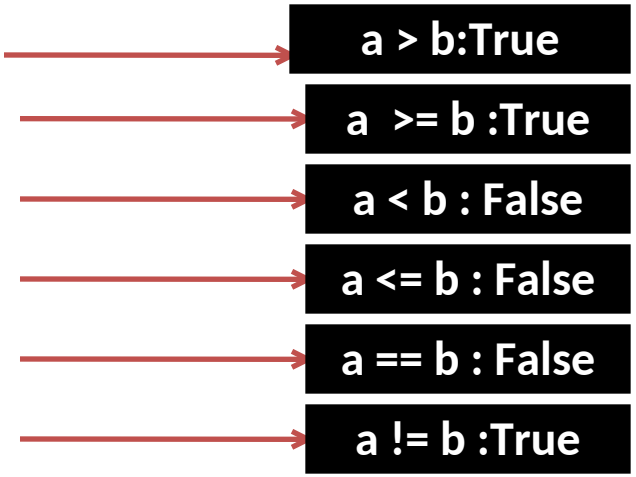


Relational operators:

> ->Greter
< ->less
>= ->Greaterthan or equal
<= ->Lesserthan or equal
== ->(comparision)
!= ->(Not Equal)

Relational operators

```
static void Main(string[] args)
    {
        int a = 5;
        int b = 2;
        Console.WriteLine("a >  b : {0}", a > b);
        Console.WriteLine("a >= b : {0}", a >= b);
        Console.WriteLine("a <  b : {0}", a < b);
        Console.WriteLine("a <= b : {0}", a <= b);
        Console.WriteLine("a == b : {0}", a == b);
        Console.WriteLine("a != b : {0}", a != b);
    }
```



Logical Operators

&&- --> and
|| ---> or
! ---> not

con1	con2	con1 && con2
true	true	true
true	false	false
false	true	false
false	false	false

con1	con2	con1 || con2
true	true	true
true	false	true
false	true	true
false	false	false

con	!con
true	false
false	true

Assignment operators

=	Assignment operator
+=	Additive Assignment
-=	Subtractive Assignment
*=	Multiplicative Assignment
/=	Division Assignment

Assignment Operators program

```
static void Main(string[] args)
        {
            int a = 10;
            int b = 20;
            Console.WriteLine("a = {0}",a);
            a += b;           // a = a+b
            Console.WriteLine("a = {0}", a);
            a -= b;
            Console.WriteLine("a = {0}", a);
            a *= b;
            Console.WriteLine("a = {0}", a);
            a /= b;
            Console.WriteLine("a = {0}", a);
        }
```

a = 10

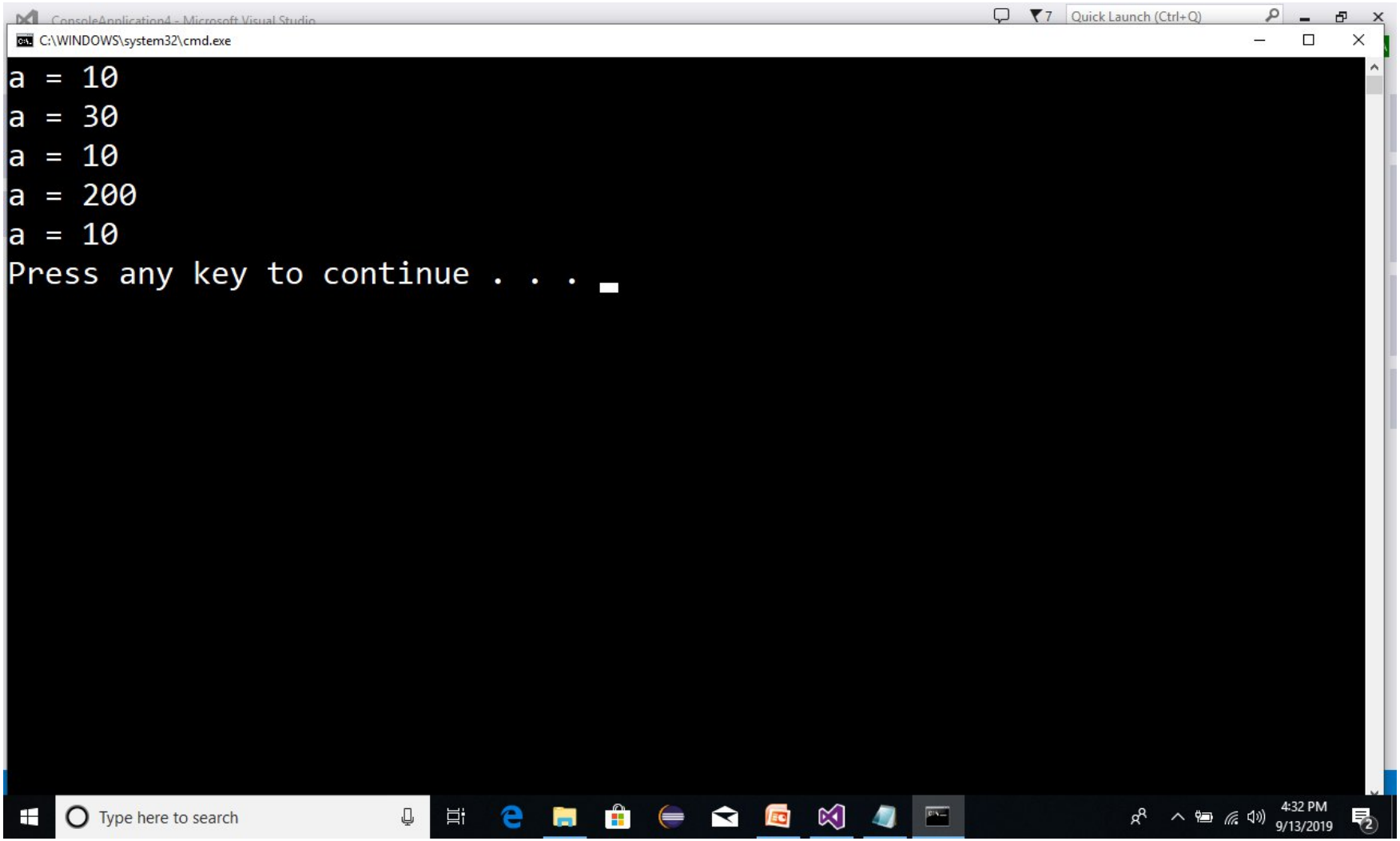
a = 30

a = 10

a = 200

a = 10

Assignment operators:



Conditional operators

? : (Ternary Operator)

con?stmt1:stmt2;

con-->True--->stmt1 can be executed

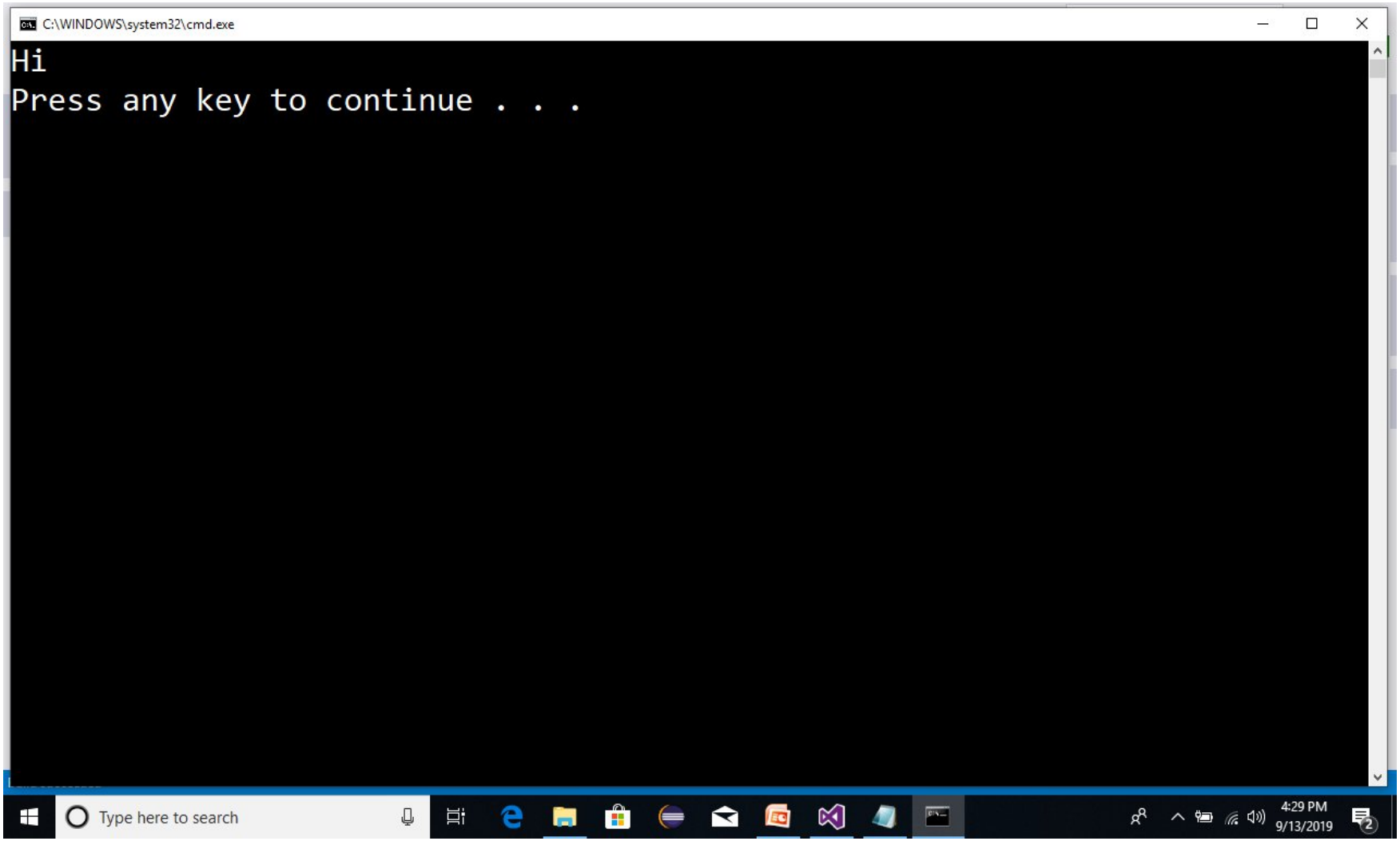
con--->False-->stmt2 can be executed

Conditional Operators program

```
static void Main(string[] args)
    {
        bool con = true;
        Console.WriteLine(con?"Hi":"Bye");
    }
```

Hi

Conditional operators:

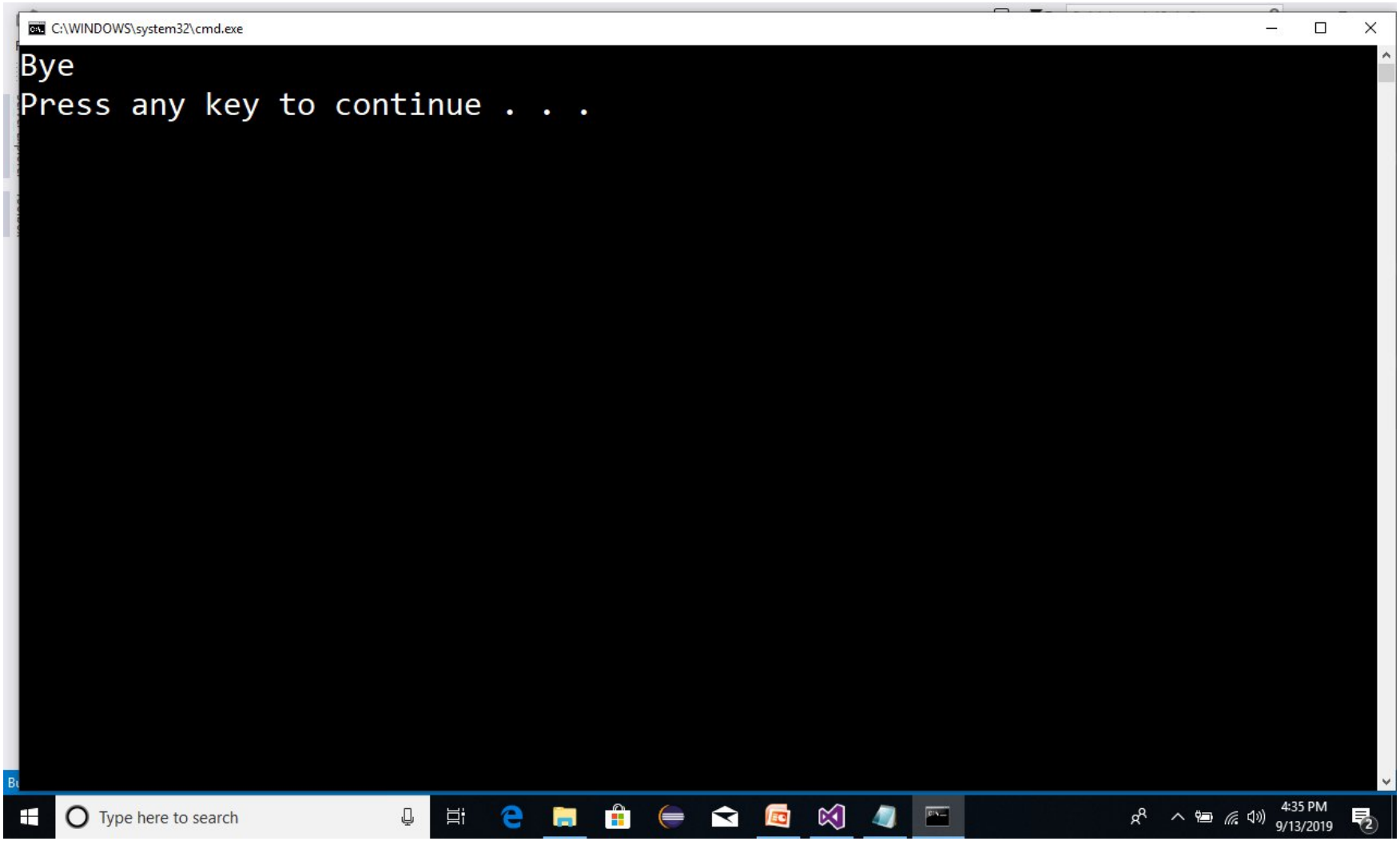


Conditional Operators program

```
static void Main(string[] args)
    {
        bool con = false;
        Console.WriteLine(con?"Hi":"Bye");
    }
```

Bye

Conditional operators:



Increment and Decrement operators

++ Increment Operator

-- Decrement Operator

Increment and Decrement Operators program

```
static void Main(string[] args)
      {
         int a = 10;
         int b = 20;
        Console.WriteLine("a = {0}",a++);  // Post Increment
        Console.WriteLine("a = {0}", a);
       Console.WriteLine(“b = {0}", ++b);
       Console.WriteLine(“b = {0}", b);
   }
```

a = 10

a = 11

a = 21

a = 21

Increment decrement operators: