CSE 3100: Web Programming Laboratory

Lab 4: C# Fundamentals

**Kazi Saeed Alam**

**Assistant Professor,**

**Dept of CSE, KUET**

**Email: saeed.alam@cse.kuet.ac.bd**

**Subah Nawar**

**Lecturer,**

**Dept of CSE, KUET**

**Email: nawar@cse.kuet.ac.bd**

Contents

[Introduction 3](#_heading=h.27hpuapo1pp0)

[Creating a new project 3](#_heading=h.rlz5tj5v96g1)

[Data types 5](#_heading=h.bqrmy66ohil)

[Primitive Types 5](#_heading=h.4f05t9z1y5q7)

[Non-Primitive Types 5](#_heading=h.gyu4hcb6ydxn)

[Scope 6](#_heading=h.hllj3aypdm10)

[Using var to declare variable type 6](#_heading=h.8mfaol815z55)

[Type conversion 7](#_heading=h.ciubqruhx31j)

[Implicit type conversion 7](#_heading=h.d15pbmtqr700)

[Explicit type conversion 7](#_heading=h.igc722d78x88)

[Try-catch 7](#_heading=h.wb35b9c9if3d)

[C# operators 8](#_heading=h.uzlkm5gop0xj)

[Arithmetic operators 8](#_heading=h.30313mlertrw)

[Comparison operators 8](#_heading=h.h90lrsmcxjsj)

[Logical operators 9](#_heading=h.dbav7udy3i42)

[Bitwise operators 9](#_heading=h.4n2qx31qvyof)

[Array 10](#_heading=h.hrtrcbkebso0)

[Methods 10](#_heading=h.j2n2xe9fhemx)

[If-else statement 10](#_heading=h.9lxp607lvwz2)

[Switch statement 11](#_heading=h.hhzyyjhzww14)

[While loop 11](#_heading=h.h5f0raw3bwt6)

[For loop 11](#_heading=h.odyspflclqsr)

[Foreach loop 11](#_heading=h.sjneqlvdicg0)

[Class 12](#_heading=h.6c1y4of1o4rq)

[Constructor 14](#_heading=h.2caid11xtai)

[Getter setter 15](#_heading=h.icbuk04axgt0)

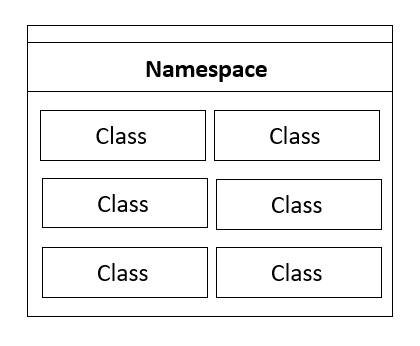
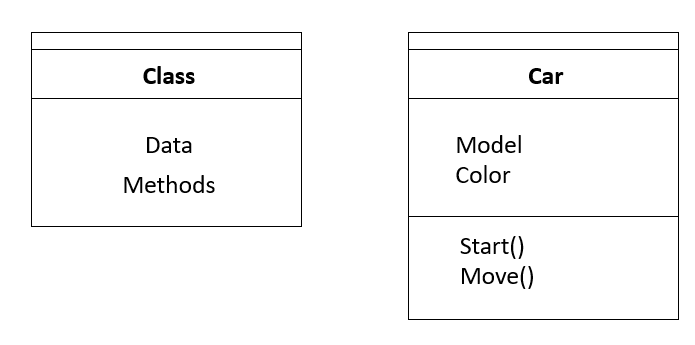
[Static class attributes 16](#_heading=h.zespm3rafkg7)

[Static methods & classes 17](#_heading=h.t7mossgxpky5)

[Inheritance 18](#_heading=h.x609s7vp7bsg)

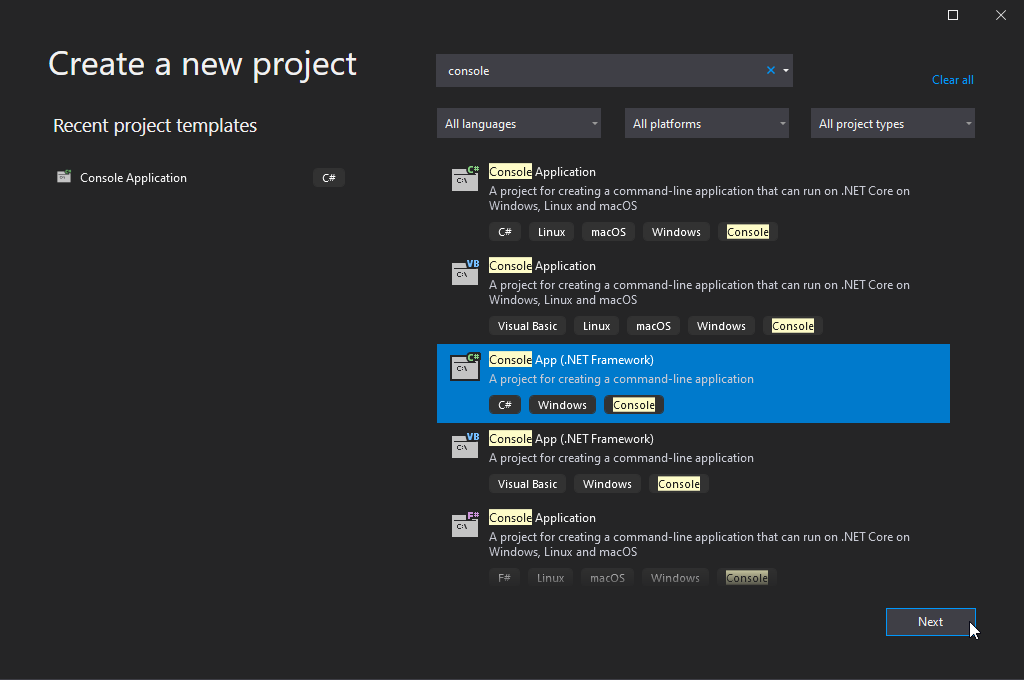
# Introduction

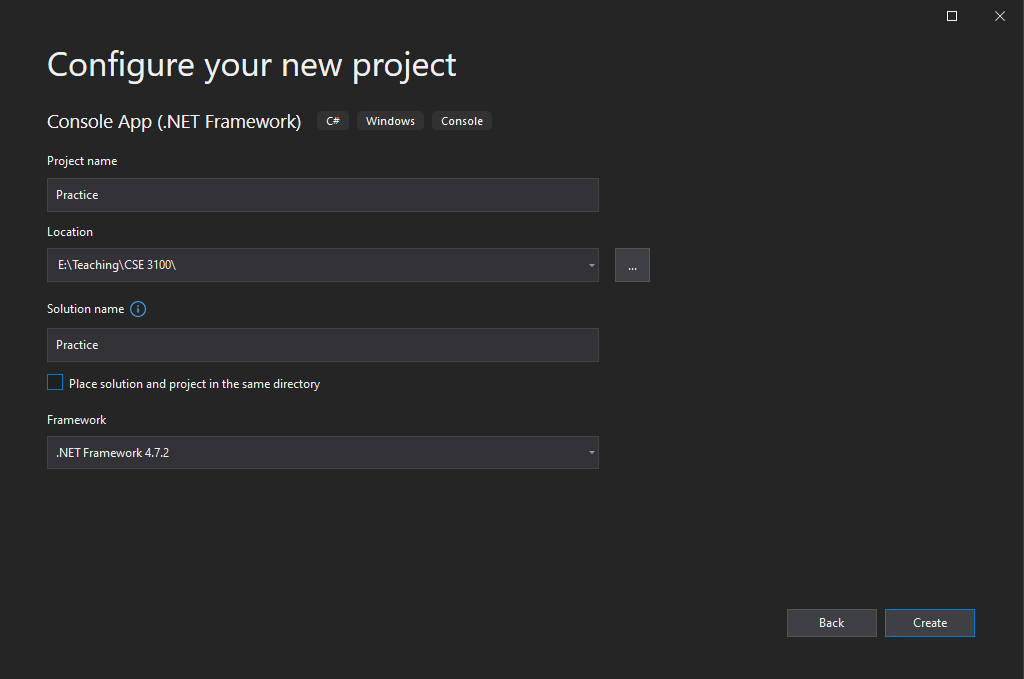
* C# is a programming language.
* .NET is a framework for building application on Windows.



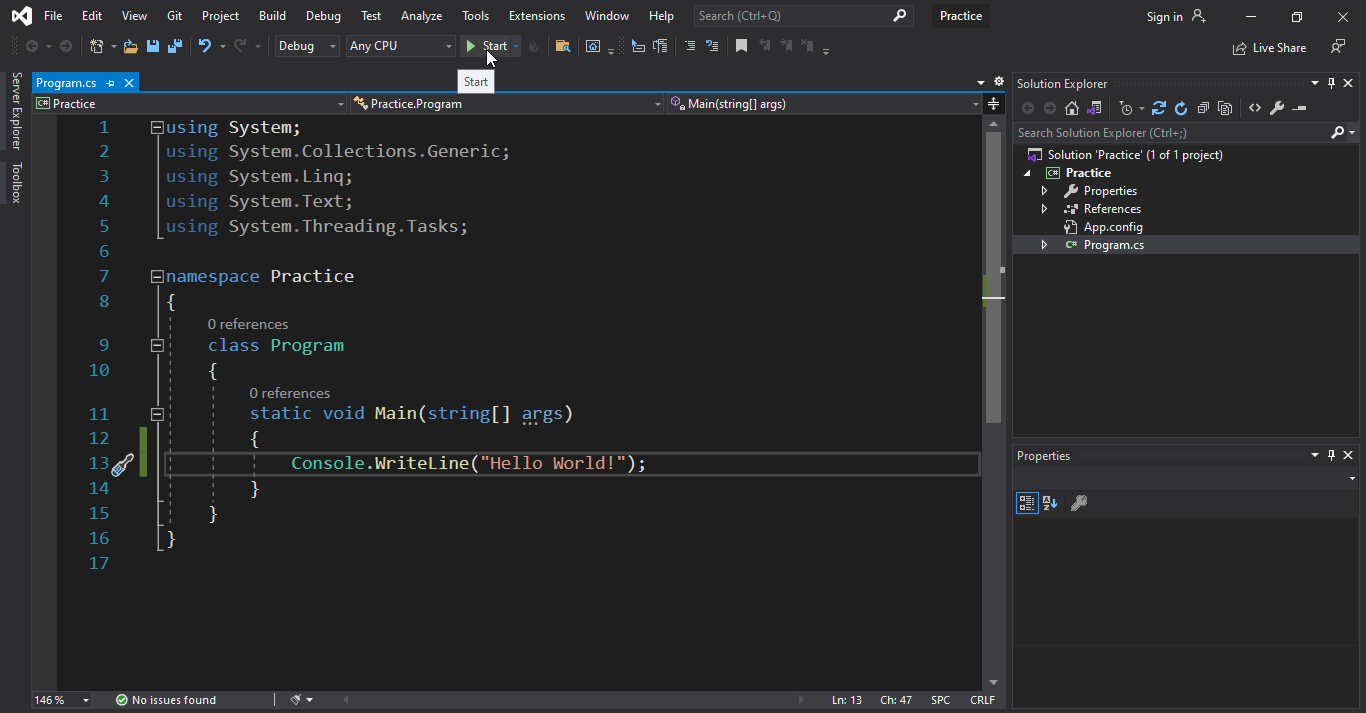
# Creating a new project

Create a new **Console App (.NET Framework)**



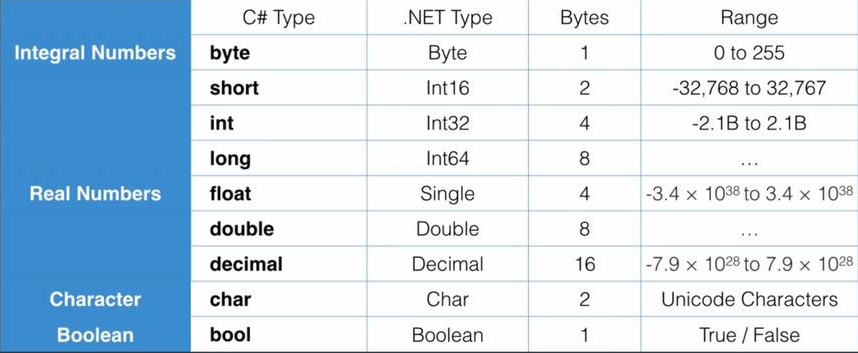


Let’s write our first program.

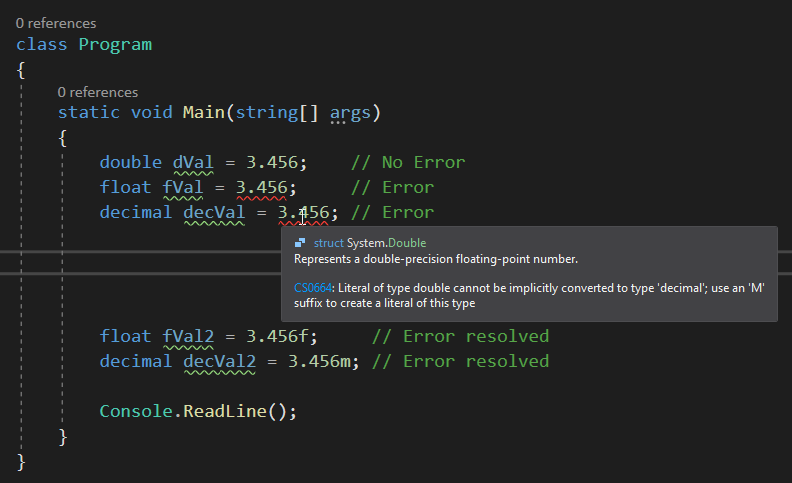


# Data types

## Primitive Types



Default **Real Number** type is **Double**. For example, if we directly write 3.456 to a float type variable, there will be an error. For that we have to write, **3.456f for float** and **3.456m for decimal.**



## 

## Non-Primitive Types

* String
* Array
* Enum
* Class

# Scope

byte a = 1;

Console.WriteLine(a);

{

Console.WriteLine(a);

byte b = 2;

Console.WriteLine(b);

{

Console.WriteLine(b);

char c = 'c';

Console.WriteLine(c);

}

Console.WriteLine(c); // Error: c is Out of scope

}

# Using var to declare variable type

class Program

{

static void Main(string[] args)

{

var number = 2;

var ch = 'a';

var str = "Hello World";

var fNumber = 45.6f;

Console.WriteLine(number);

Console.WriteLine(ch);

Console.WriteLine(str);

Console.WriteLine(fNumber);

Console.ReadLine();

}

}

# Type conversion

## Implicit type conversion

byte b = 1;

int i = b; // byte value will be converted to int value

Console.WriteLine("b = {0}, i = {1}", b, i);

## Explicit type conversion

int i = 1;

byte b = i; // Compliation error: Won't be converted implicitly

Console.WriteLine("b = {0}, i = {1}", b, i);

To solve this,

int i = 1;

byte b = Convert.ToByte(i);

Console.WriteLine("b = {0}, i = {1}", b, i);

Another example,

var number = "1234";

int i = Convert.ToInt32(number);

Console.WriteLine("number = {0}, i = {1}", number, i);

# Try-catch

try

{

var number = "1234";

byte b = Convert.ToByte(number);

Console.WriteLine("number = {0}, i = {1}", number, b);

}

catch (Exception e)

{

Console.WriteLine(e.Message);

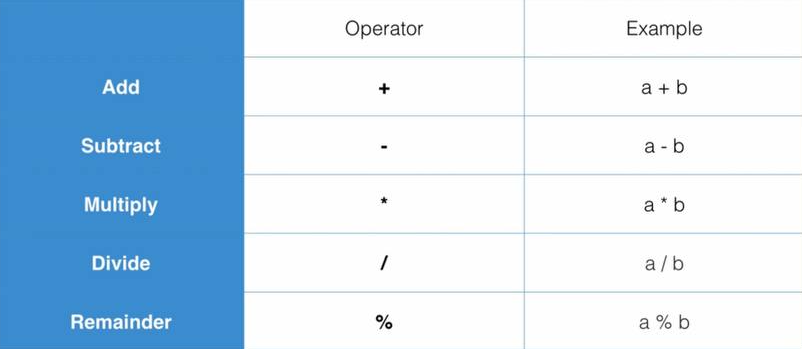
}

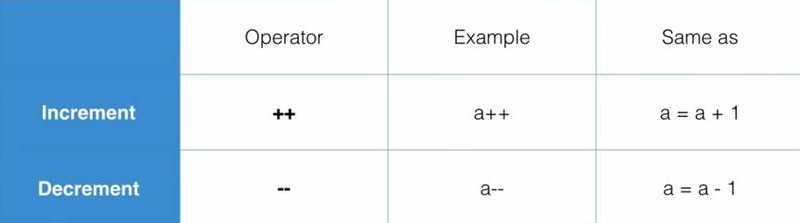
Output:

Value was either too large or too small for an unsigned byte.

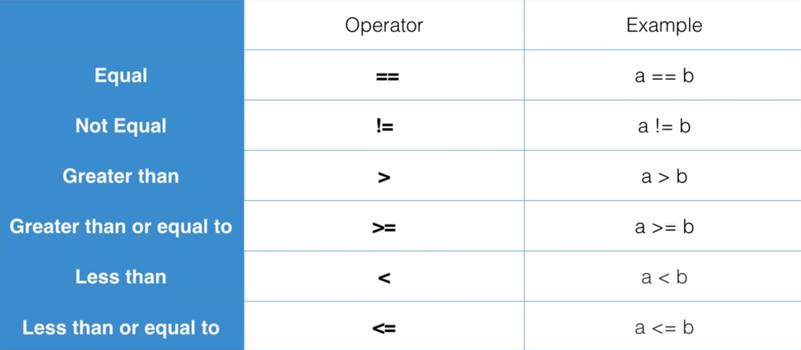
# C# operators

## Arithmetic operators

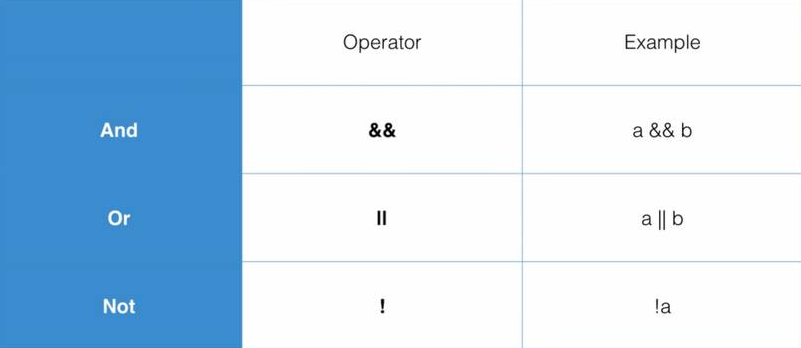




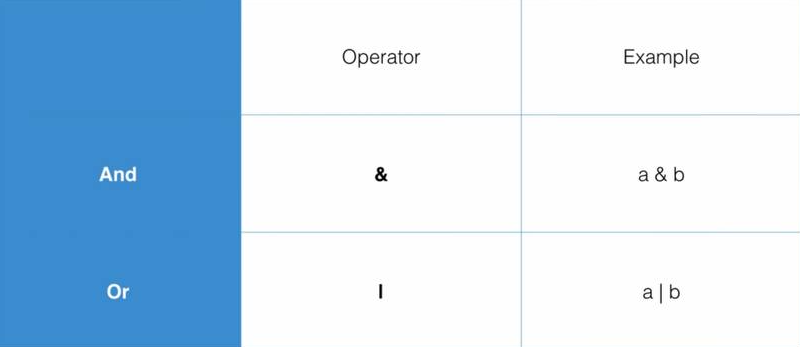
## Comparison operators



## Logical operators



## Bitwise operators



# Array

// 1D array

int[] numbers = { 1, 2, 3, 4, 5 };

string[] friends = new string[5];

friends[0] = "Jim";

friends[1] = "Kelly";

//2D array

int[,] numbers2d =

{

{ 1, 2, 3 },

{ 4, 5, 6 },

{ 7, 8, 9 }

};

Console.WriteLine(numbers2d[2, 1]);

# Methods

static void Main(string[] args)

{

SayHi("Jack");

Console.WriteLine(AddNumbers(1, 3));

Console.ReadLine();

}

static void SayHi(string name)

{

Console.WriteLine("Hello " + name);

}

static double AddNumbers(double a, double b)

{

return a + b;

}

# If-else statement

static int GetMax(int a, int b)

{

int result;

if (a > b)

{

result = a;

}

else

{

result = b;

}

return result;

}

# Switch statement

static string GetDay(int dayIndex)

{

string dayName;

switch (dayIndex)

{

case 0:

dayName = "Sunday";

break;

case 1:

dayName = "Monday";

break;

case 2:

dayName = "Tuesday";

break;

case 3:

dayName = "Wednesday";

break;

case 4:

dayName = "Thursday";

break;

case 5:

dayName = "Friday";

break;

case 6:

dayName = "Saturday";

break;

default:

dayName = "None";

break;

}

return dayName;

}

# While loop

int i = 0;

while (i < 5)

{

Console.WriteLine(i);

i++;

}

# For loop

for (int i = 0; i < 5; i++)

{

Console.WriteLine(i);

}

# Foreach loop

int[] numbers = { 1, 2, 3, 4, 5 };

foreach (int value in numbers)

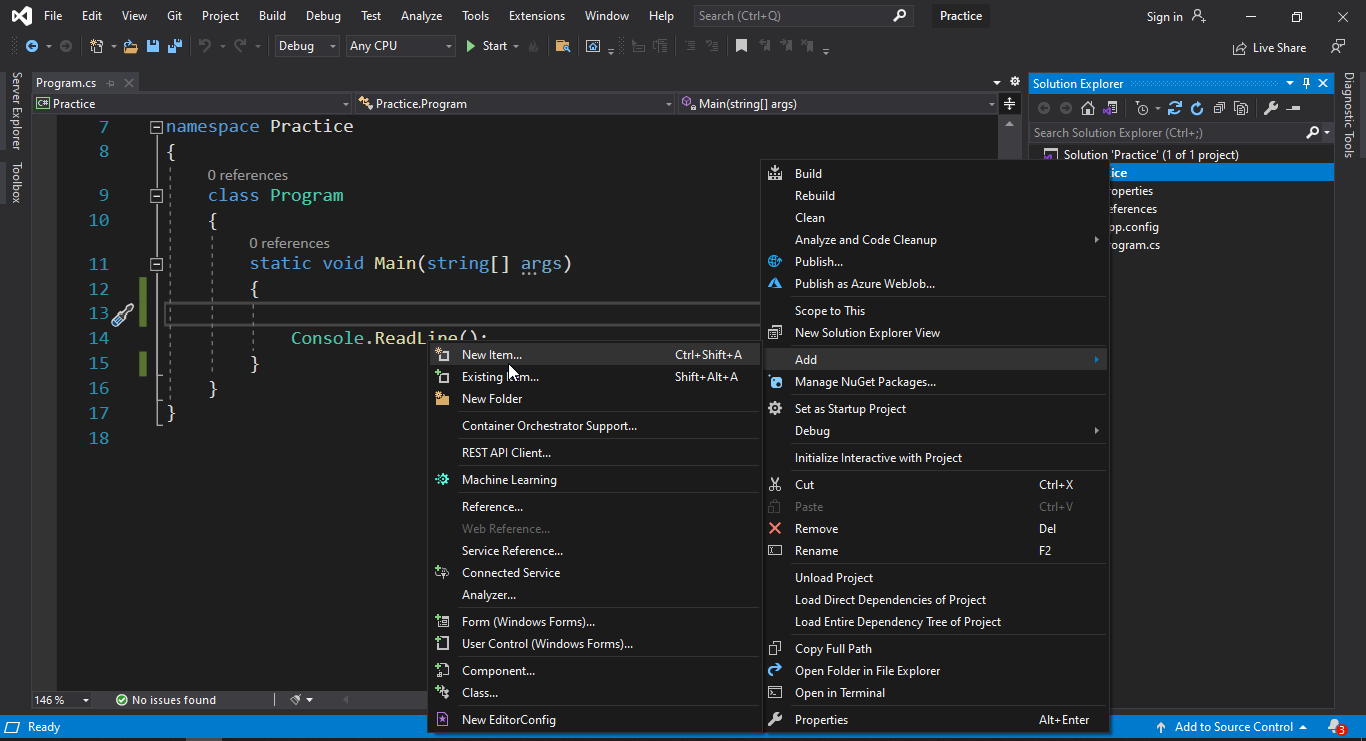
{

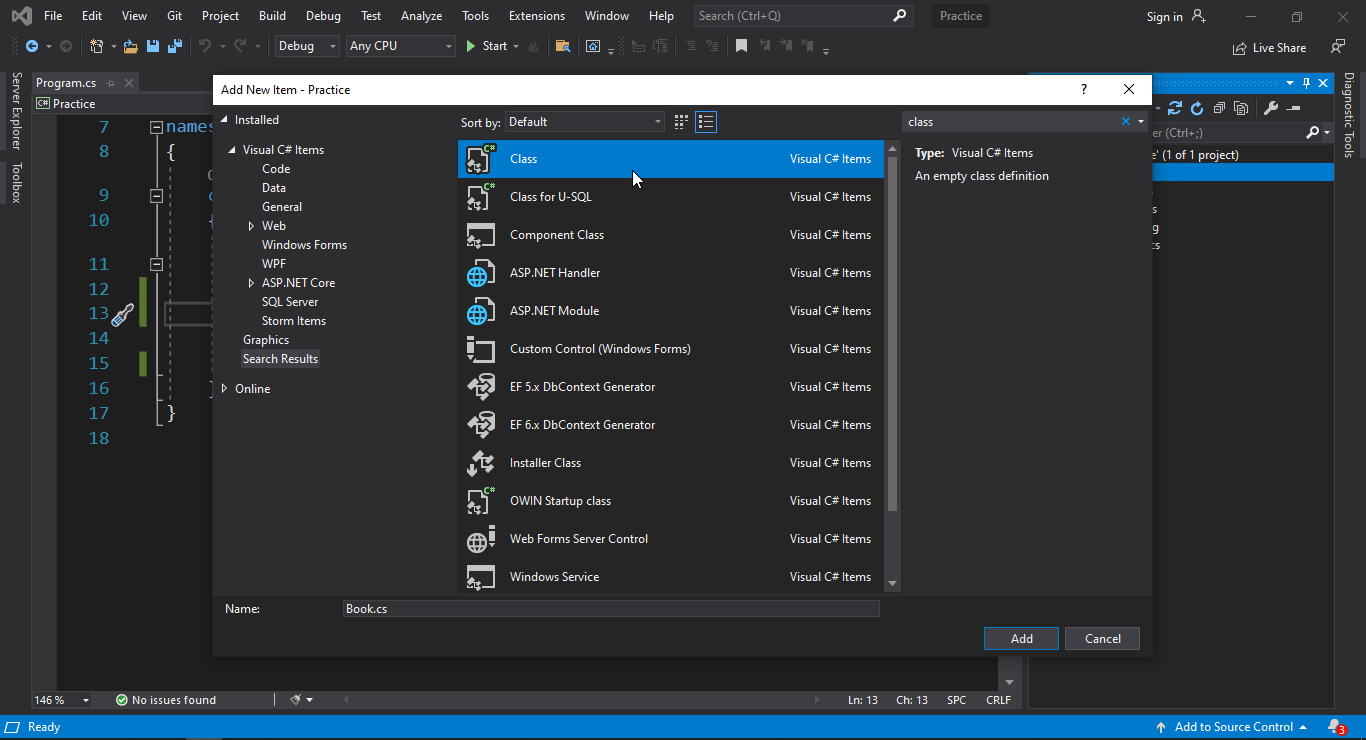
Console.WriteLine(value);

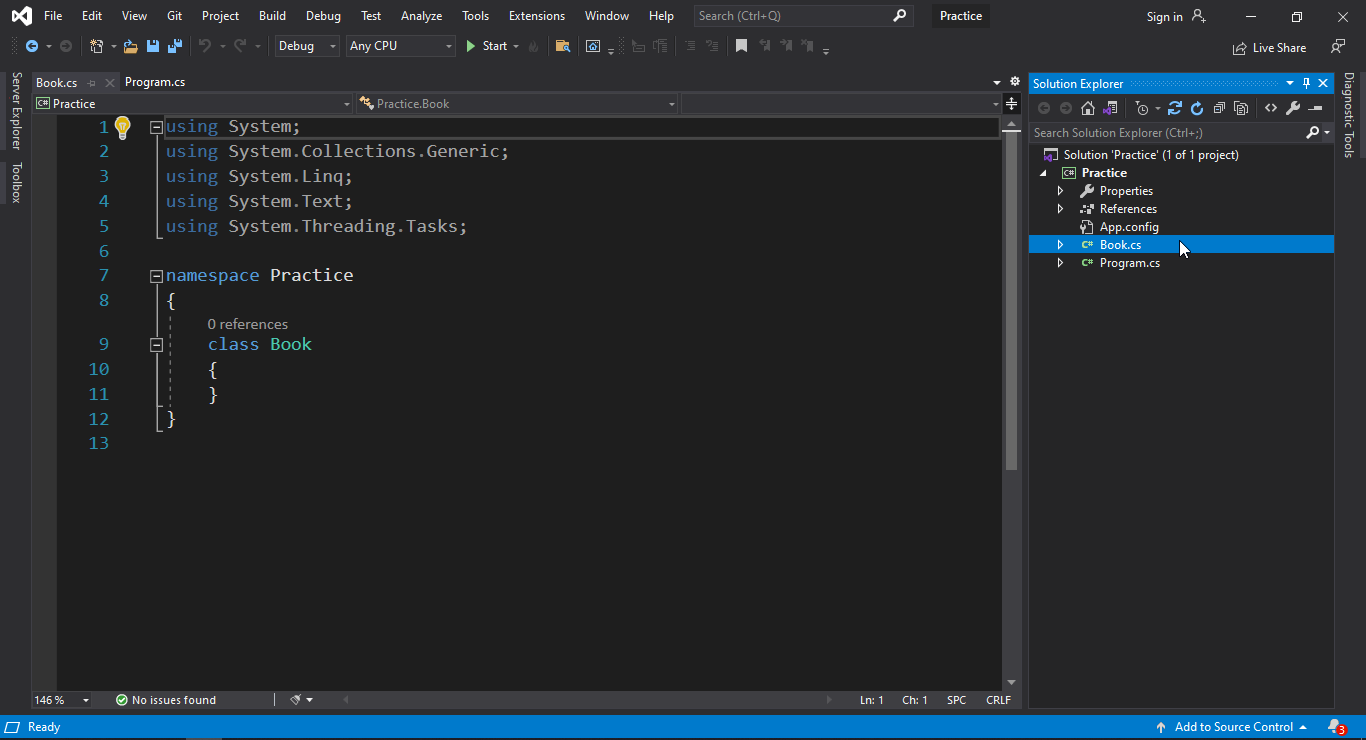
}

# Class

Creating a new class in the project,







New class **Book** will be created on same namespace.

**Book** class:

class Book

{

public string title;

public string author;

public int pages;

public string GetDetails() // Method

{

return "Name: "+ title +"; Author "+ author +"; "+ Convert.ToString(pages) +" pages.";

}

}

Main **Program** class:

class Program

{

static void Main(string[] args)

{

Book book1 = new Book();

book1.title = "Harry Potter";

book1.author = "JK Rowling";

book1.pages = 400;

Console.WriteLine(book1.GetDetails());

Console.ReadLine();

}

}

# Constructor

**Book** class:

class Book

{

public string title;

public string author;

public int pages;

public Book() // Constructor

{

}

public Book(string aTitle, string aAuthor, int aPages) // Constructor

{

title = aTitle;

author = aAuthor;

pages = aPages;

}

public string GetDetails()

{

return "Name: "+ title +"; Author "+ author +"; "+ Convert.ToString(pages) +" pages.";

}

}

At **Program** class:

class Program

{

static void Main(string[] args)

{

Book book1 = new Book();

book1.title = "Harry Potter";

book1.author = "JK Rowling";

book1.pages = 400;

Console.WriteLine(book1.GetDetails());

Book book2 = new Book("The Falling of Love", "Marisa Oldham", 678);

Console.WriteLine(book2.GetDetails());

Console.ReadLine();

}

}

# Getter setter

**Book** class:

class Book

{

public string title;

public string author;

public int pages;

private int rating; // Can't access beyond the scope of this class

public int Rating

{

get

{

return rating;

}

set

{

if (value < 0) rating = 0;

else if (value > 5) rating = 5;

}

}

public Book(string aTitle, string aAuthor, int aPages, int aRating)

{

title = aTitle;

author = aAuthor;

pages = aPages;

Rating = aRating;

}

}

**Program** class:

class Program

{

static void Main(string[] args)

{

Book book1 = new Book("Harry Potter", "JK Rolling", 400, 5);

Book book2 = new Book("The Falling of Love", "Marisa Oldham", 678, 6);

Console.WriteLine(book2.rating); // Error

Console.WriteLine(book2.Rating);

Console.ReadLine();

}

}

# Static class attributes

**Book** class:

class Book

{

public string title;

public string author;

public int pages;

public static int bookCount;

public Book(string aTitle, string aAuthor, int aPages, int aRating)

{

title = aTitle;

author = aAuthor;

pages = aPages;

bookCount++;

}

public int GetBookCount()

{

return bookCount;

}

}

**Program** class:

class Program

{

static void Main(string[] args)

{

Book book1 = new Book("Harry Potter", "JK Rolling", 400, 5);

Console.WriteLine(Book.bookCount); // 1

Book book2 = new Book("The Falling of Love", "Marisa Oldham", 678, 6);

Console.WriteLine(book2.GetBookCount()); // 2

Console.ReadLine();

}

}

# Static methods & classes

**UsefulTools** class:

static class UsefulTools

{

public static void SayHi(string name)

{

Console.WriteLine("Hello " + name);

}

}

**Program** class:

class Program

{

static void Main(string[] args)

{

UsefulTools.SayHi("Jack");

Console.ReadLine();

}

}

# Inheritance

**Chef** class:

class Chef

{

public void MakeChicken()

{

Console.WriteLine("The chef makes chicken");

}

public void MakeSalad()

{

Console.WriteLine("The chef makes salad");

}

public virtual void MakeSpecialDish()

{

Console.WriteLine("The chef makes bbq ribs");

}

}

**ItalianChef** class inherits **Chef** class, also overrides method **MakeSpecialDish()**:

class ItalianChef : Chef

{

public override void MakeSpecialDish()

{

Console.WriteLine("The italian chef makes pasta");

}

public void MakePizza()

{

Console.WriteLine("The italian chef makes pizza");

}

}

**Program** class:

class Program

{

static void Main(string[] args)

{

Chef chef = new Chef();

chef.MakeChicken();

chef.MakeSalad();

chef.MakeSpecialDish();

ItalianChef italian = new ItalianChef();

italian.MakeChicken();

italian.MakeSalad();

italian.MakeSpecialDish();

italian.MakePizza();

Console.ReadLine();

}

}