```
using System;

//systems library gives you useful tools like writeing to console etc.


/*-Methods are indented inside the definition of the class (move to the right by one
or more [Tab] characters);
-Method contents are indented inside the definition of the method;
-The opening curly bracket { must be on its own line and placed exactly under the
method or class it refers to;
-The closing curly bracket } must be on its own line, placed exactly vertically
under the respective opening
 bracket (with the same indentation);

-All class names must start with a capital letter;
-Variable names must begin with a lower-case letter;
-Method names must start with a capital letter;

Code indentation follows a very simple rule: when some piece of code is logically
inside another piece of code,
    it is indented (moved) on the right with a single [Tab]. For example if a method
is defined inside a class,
    it is indented(moved to the right). In the same way if a method body is inside a
method, it is indented.
    To simplify this, we can assume that when we have the character “{“, all the
code after it until its closing “}”
    should be indented on the right.*/

// can comment selecting text and hitting ctrl+shift+/
// or just double forward slash to comment in line... for example... behind your
code

/**********************************Part 1***********************************/

class HelloCSharp
{
    static void Main(string[] args)
    {
        Console.WriteLine("Hello, World!");
    }

}

// F5 Quickly lets you debug

/*A class is a data structure in C# that combines data variables and functions into
a single unit.*/
/*"Main" is your method here... also known as a function... this way you can call it
over and over again*/
/*static means that the method belongs to the Program class and not an object of the
Program class.*/
/*void means this method does not have a return value*/

/*Main is however meant for the main body of your code... this is where you should
call methods, Not write them.*/

/**********************************Part 2***********************************/
```

```
namespace PracticeGreetings
//namespace is for organization
{
    class HelloCSharp
    {
        static void Hello()
        {
            Console.WriteLine("Hello, World!");
        }
        static void Main(string[] args)
        {
            Hello();
        }

    }

}

/**********************************Part 3**********************************/

namespace PracticeGreetings
{
    class HelloCSharp
    {
        static void Hello()
        {
            Console.WriteLine("Hello, World!");
            Console.WriteLine("I'm  Computer Programmer");
            Console.WriteLine("Look at me go!!!!");

            Console.WriteLine("I can even do math!!!!");
            Console.WriteLine(3+3);

            Console.Write("No more new line");
            Console.Write("All just mashed together");
        }
        static void Main(string[] args)
        {
            Hello();
        }

    }

}

/**********************************Part 4**********************************/

/*int -stores integers(whole numbers), without decimals, such as 123 or -123
double - stores floating point numbers, with decimals, such as 19.99 or -19.99
char - stores single characters, such as 'a' or 'B'. Char values are surrounded by
single quotes
string - stores text, such as "Hello World". String values are surrounded by double
quotes
bool - stores values with two states: true or false*/

namespace Variables
{
```

```
    class IntroToVariables
    {
        static void Variables()
        {
            //Variables need to be defined using type and then value
            string yourName = "Schaub";
            //int FavNum = 4;
            //long can be used for very very big numbers
            //float can be used for small doubles.... just use double probably
easier... but using more memory
            double favDecimal = 3.14D;
            char favLetter = 'Z';
            bool trueOrFalse = true;
            Console.WriteLine(yourName);
            //Console.WriteLine(FavNum);
            Console.WriteLine(favDecimal);
            //Console.WriteLine(FavNum+FavDecimal);
            Console.WriteLine(favLetter);
            Console.WriteLine(trueOrFalse);

            //careful as these can just be overwritten

            //FavNum = 8;
            //Console.WriteLine(FavNum);

            // Can declare as constant

            const int favNum = 4;
            Console.WriteLine(favNum);

            // can combine display variables
            string greeting = "Hello my name is " + yourName + " My favorite number
is " + favNum;
            Console.WriteLine(greeting);

            //do math as well
            int x = 5;
            int y = 6;
            Console.WriteLine("Five + Six = " + (x+y));

            //casting is changing a variable from one type to another... can
automatically go int -> double...
            //but need to explicitly cast the other way around

            double testDouble = 5.678D;
            int castDouble = (int)testDouble;
            Console.WriteLine(testDouble);
            Console.WriteLine(castDouble);
            Console.WriteLine(Convert.ToString(castDouble));
        }
        static void Main(string[] args)
        {
            Variables();
        }

    }

}
```

```
/**********************************Part 5**********************************/

namespace UserInputs
{
    class UserFriendly
    {
        static void Inputs()
        {
            Console.WriteLine("Hello Stranger, what is your name?");
            String strangersName = Console.ReadLine();
            Console.WriteLine("Nice to meet you " + strangersName);

            Console.WriteLine("How old are you?");
            int strangersAge = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Wow " + strangersAge + "You don't look a day under
105");



        }
        static void Main(string[] args)
        {
            Inputs();
        }

    }

}

/**********************************Part 6**********************************/

namespace ArithmaticOperators
//aka math
{
    class LetsDoMath
    {
        static void MathTime()
        {

/*            Operator Name            Description Example
                        +Addition        Adds together two values
x + y
                    -       Subtraction     Subtracts one value from another x -y
              *       Multiplication  Multiplies two values x *y
/       Division        Divides one value by another x / y
%       Modulus Returns the division remainder                  x % y
++      Increment Increases the value of a variable by 1        x++
--      Decrement Decreases the value of a variable by 1        x--*/

            int num1 = 8;
            int num2 = 12;
            int sum1 = num1+num2;

            Console.WriteLine("The sum is " + sum1);

            for (int i = 0; i < 5; i++)
            {
                Console.WriteLine(i);
```

```
            }

            //can also assign variables and do variable math

/*            Operator Example     Same As
            = x = 5       x = 5
 +=           x += 3      x = x + 3
 -=           x -= 3      x = x - 3
 *=           x *= 3      x = x * 3
 /=           x /= 3      x = x / 3*/

            int x = 5;

            // want to add 8?
            Console.WriteLine("I was 5.... see: " + x);
            x += 8;
            Console.WriteLine("I was 5.... then we added 8: " + x);

            // logic operators... true/false
            //can do comparisons... this are critical for loops

/*            Operator Name            Example
            ==          Equal to x == y
    !=          Not equal x != y
>           Greater than x > y
<           Less than x<y
                        >=  Greater than or equal to    x >= y
                        <=  Less than or equal to       x <= y

            num1 = 3;
            num2 = 4;*/

            bool compareNum = num1 == num2;

            Console.WriteLine("I compared the numers "+ num1+" and "+num2+" and they
are equal: "+ compareNum);

            //can due multiple logic operators as well... also handy for loops

/*            Operator Name            Description Example
            &&          Logical and Returns True if both statements are true
x < 5 &&  x < 10
        ||          Logical or Returns True if one of the statements is true
x < 5 || x < 4
   !          Logical not     Reverse the result, returns False if the result is
true!(x < 5 && x < 10)*/


            bool doubleComparison = (num1 < 5 && num2 < 5);

            Console.WriteLine("Both the numers "+ num1+" and "+num2+" are less than
5: "+ doubleComparison);

            //few more fancy math things

            int winner = Math.Max(num1, num2);
            Console.WriteLine("Of the numers "+ num1+" and "+num2+", "+ winner+" is
bigger");
```

```
            Math.Min(num1, num2);
            Console.WriteLine(Math.Sqrt(num2));
            Math.Sqrt(num1);


        }
        static void Main(string[] args)
        {
            MathTime();
        }

    }

}
```