# CITP 155 – Programming I

## Comments Lab

## Graded Activity (15 points)

Provide any screen shots using the Snipping Tool and selecting only the relevant portion of the screen (instead of the entire screen). Provide any answers using a blue font. Please note that part of being a good programmer is being precise. If you have typos in your code, such as the words that are supposed to be displayed, you will not receive full credit.

KEEP ALL OF YOUR SOLUTION AND PROJECT FILES THROUGHOUT THE DURATION OF THE CLASS!

Concepts taken from the online C# tutorial <https://csharp.net-tutorials.com/basics/code-comments/>

**Open a Visual Studio Solution**

Open your Classes solution

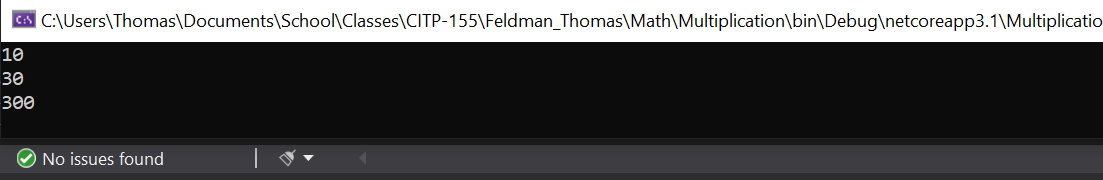
Location: c:\users\student\documents\visual studio 2015\Projects\LastName\_FirstName\Classes\Classes.sln

**Open and Test Your C# Program (2 points)**

Open the Math program.

Before making any changes to the program, run the debugger to ensure that your program works properly.

Provide a screen shot of your testing. The screen shot should include the title bar of the console with the full path of the program and three lines of white text (the output of the program).



**Add Comments to Your C# Program**

**As you add the following comments to your program, run the debugger from time to time to make sure you have not impacted the functionality of the program.**

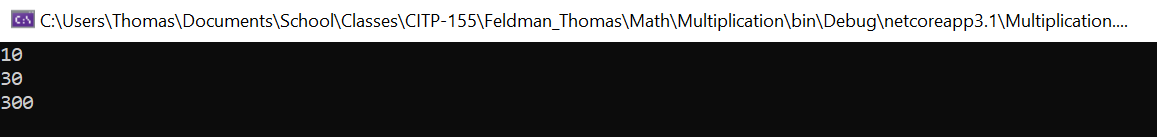
Using the coding examples found in the <https://csharp.net-tutorials.com/basics/code-comments/> as a guide, add the following comments to your program. Please use the basic Single-line, and Multi-line comments.

* After the Using Statement(s), add the following as a Multi-line comment  
  **Initial Developer: Xxx Xxx  
  Initial Development Date: xx/xx/xxxx  
  Modification: xx/xx/xxxx**replacing the x’s with appropriate information
* In the Main method, add an inline Single-line comment (on the same line as the code itself) on the ReadLine line of code. The comment should say:  
  **Stops the Console awaiting input**
* In the Main method, add a Single-line comment above the lines of code that call the static Plus method. The comment should say:  
  **Calling the Plus method without instantiating the Class**
* In the Main method, add a Single-line comment above Console.ReadLine(); The comment should be a TODO token and include the following reminder:  
  **Come up with a way to list out the individual numbers which were added together to get this answer**
* In the SillyMath class, before the first method is defined, add the following Multi-line comment:  
  **These methods call one another based on the number of integers passed in.  
  The real work is done in the method receiving four integers.**
* In the method designed to receive two integers, before the return line of code, add a comment which will be a TODO token and include the following reminder:  
  **This logic could be revised to use different variable names for both variables**
* In the method designed to receive three integers, before the return line of code, add a comment which will be a TODO token and include the following reminder:  
  **This logic could be revised to use different variable names for all three variables**
* In the method designed to receive four integers, before the return line of code, add the following Single-line comment.  
  **The real work is done here**
* At the end of the program, after the last curly brace, add the following Multi-line comment:  
  **The math done in this program is utilized by calls from many applications.  
  Any modifications should include a full round of testing.**

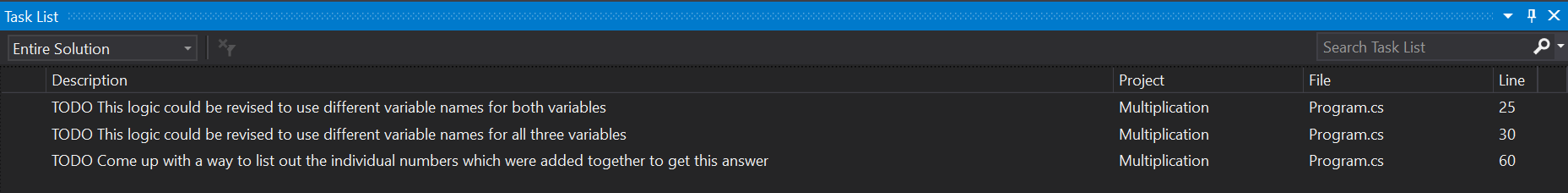
**Provide Results (13 points)**

Test the program (run the debugger).

Provide a screen shot of your testing. The screen shot should include the title bar of the console with the full path of the program and three lines of white text (the output of the program).



Provide a screen shot of the Task List which should include your TODO items from this application as well as the ones we created in class together.



Copy and paste the lines of code from your program here. This is not a screen shot. This is code I can copy and paste to run on my own.

using System;

/\*

Initial Developer: Thomas Feldman

Initial Development Date: 02/22/2021

Modification: 02/27/2021

\*/

namespace Multiplication

{

class Multiply

{

/\*

These methods call one another based on the number of integers passed in.

The real work is done in the method receiving four integers.

\*/

//The real work is done here

public static int Calculate(int numberOne, int numberTwo, int numberThree, int numberFour)

{

return numberOne \* numberTwo \* numberThree \* numberFour;

}

//TODO This logic could be revised to use different variable names for both variables

public static int Calculate(int numberOne, int numberTwo)

{

return Multiply.Calculate(numberOne, numberTwo, 1);

}

//TODO This logic could be revised to use different variable names for all three variables

public static int Calculate(int numberOne, int numberTwo, int numberThree)

{

return Multiply.Calculate(numberOne, numberTwo, numberThree, 1);

}

}

class Program

{

static void Main(string[] args)

{

int returnedValue = 1;

int numOne = 2;

int numTwo = 5;

int numThree = 3;

int numFour = 10;

//Calling the Plus method without instantiating the Clas

returnedValue = Multiply.Calculate(numOne, numTwo);

Console.WriteLine(returnedValue);

returnedValue = Multiply.Calculate(numOne, numTwo, numThree);

Console.WriteLine(returnedValue);

returnedValue = Multiply.Calculate(numOne, numTwo, numThree, numFour);

Console.WriteLine(returnedValue);

//TODO Come up with a way to list out the individual numbers which were added together to get this answer

Console.ReadLine(); //Stops the Console awaiting input

}

}

}

/\*

The math done in this program is utilized by calls from many applications.

Any modifications should include a full round of testing

\*/