CO453 Application Programming

Repetition (Loops) and Arrays

Session Summary

Conditional Loops

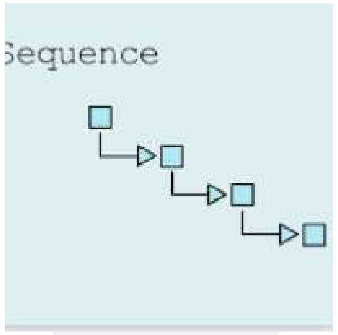
Fixed Loops

Try..Catch()

Arrays

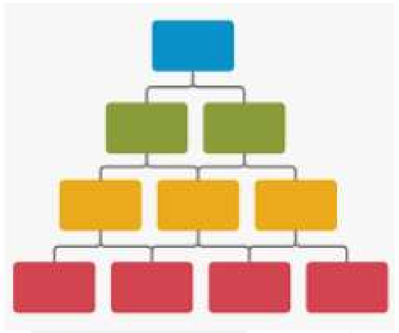
Unit Tests

5 Fundamental Programming Concepts



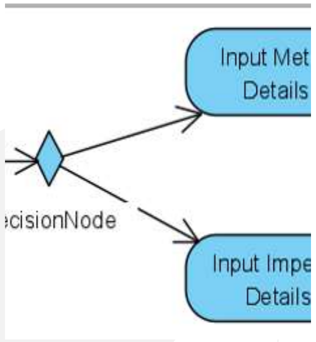
Sequence

The first statement is executed before the second statement



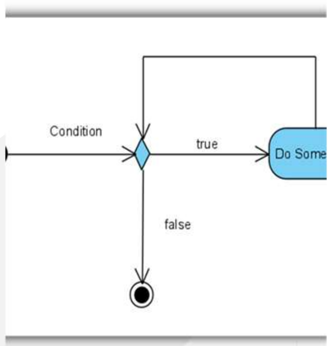
Hierarchy

Programs contain classes, which contain methods which contain statements



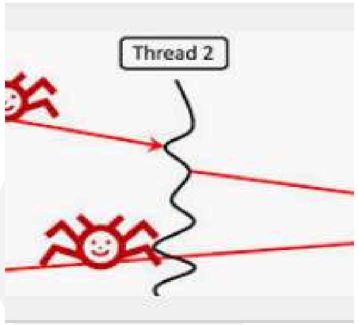
Selection

If the condition is true one thing happens otherwise another thing happens



Repetition

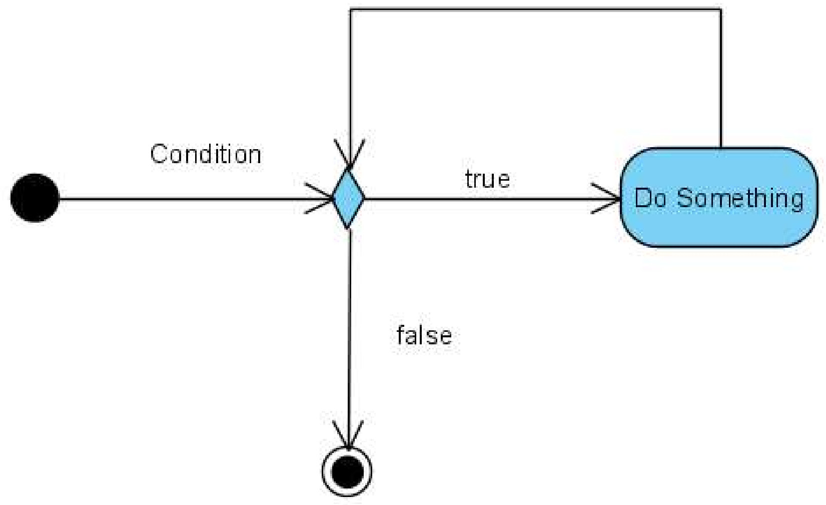
A series of statements is executed one or more times



Concurrency

One or more program threads are executed at the same time

C# Conditional while loop (check before)



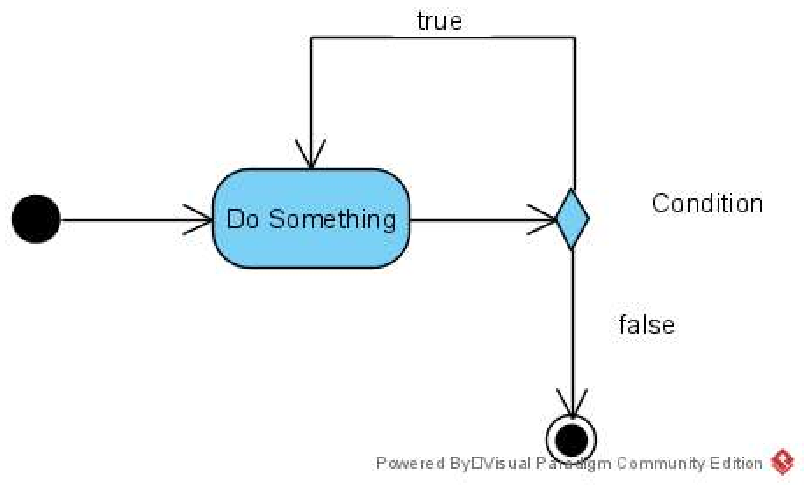
```
bool carryOn = false;

while (carryOn)
{
    // do something
}
```

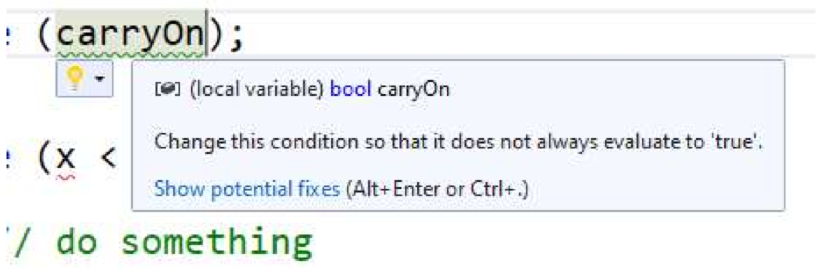
How many?

```
int x = 1;

while (x < 10)
{
    // do something
    x++;
}
```

How many?

C# Conditional do…while loop (check after)





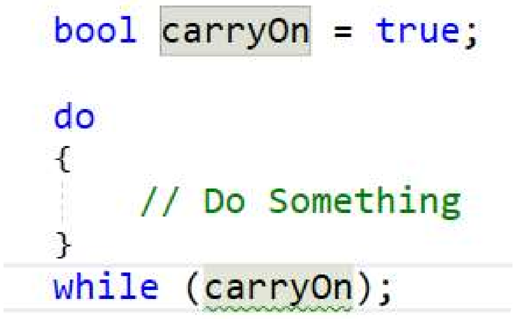
```
: (carryOn);
```

(local variable) bool carryOn

Change this condition so that it does not always evaluate to 'true'.

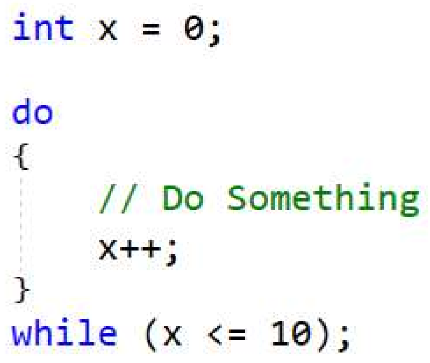
Show potential fixes (Alt+Enter or Ctrl+.)

```
: (x <
'/ do something
```



```
bool carryOn = true;

do
{
    // Do Something
}
while (carryOn);
```

How many?



```
int x = 0;

do
{
    // Do Something
    x++;
}
while (x <= 10);
```

How many?

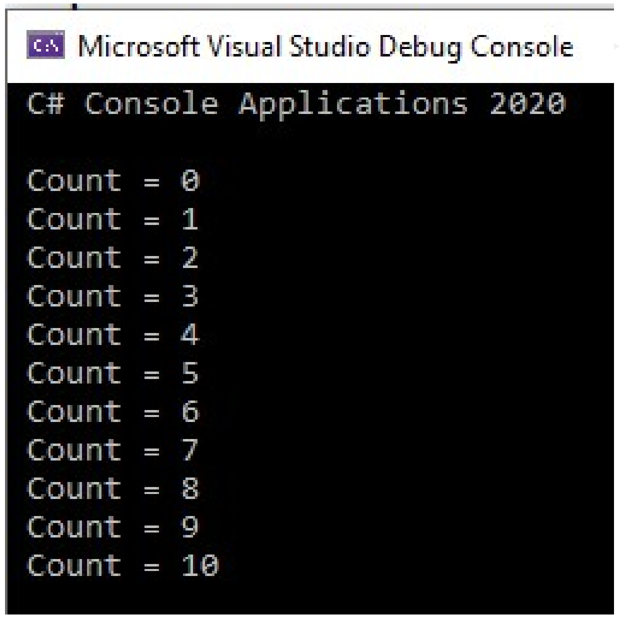
Fixed Loops

```
for(int count = 0; count <= 10; count++)
{
    // Do Something
}
```

How many?

```
public const int MAXN_PEOPLE = 10;
```

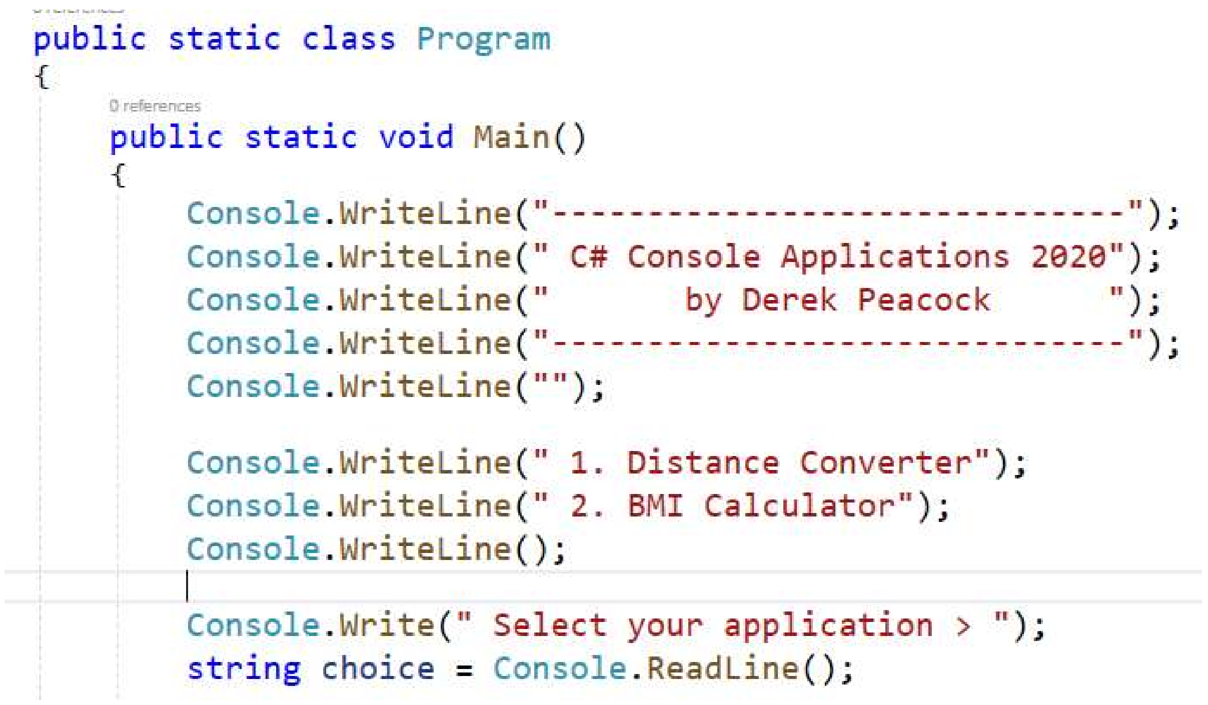
```
for(int count = 0; count < MAXN_PEOPLE; count++)
{
    Console.WriteLine($" Count = {count}");
}
```



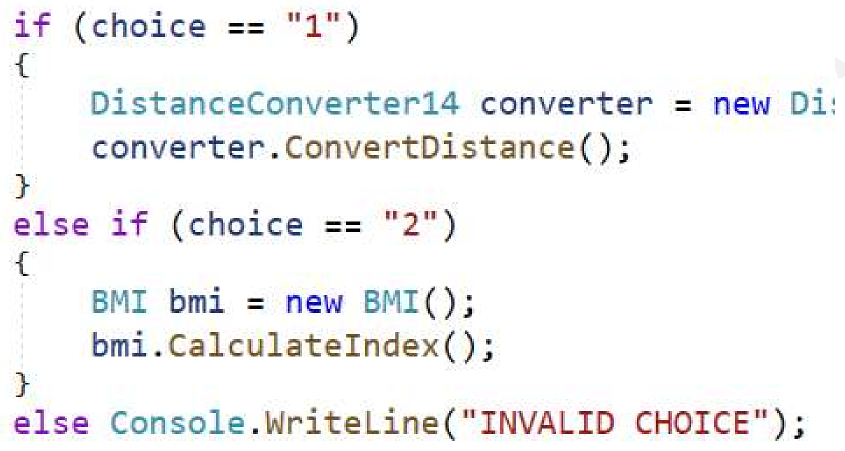
Microsoft Visual Studio Debug Console

```
C# Console Applications 2020

Count = 0
Count = 1
Count = 2
Count = 3
Count = 4
Count = 5
Count = 6
Count = 7
Count = 8
Count = 9
Count = 10
```

Program.cs



```
public static class Program
{
    0 references
    public static void Main()
    {
        Console.WriteLine("------------------------------");
        Console.WriteLine(" C# Console Applications 2020");
        Console.WriteLine("       by Derek Peacock       ");
        Console.WriteLine("------------------------------");
        Console.WriteLine("");

        Console.WriteLine(" 1. Distance Converter");
        Console.WriteLine(" 2. BMI Calculator");
        Console.WriteLine();

        Console.Write(" Select your application > ");
        string choice = Console.ReadLine();
```



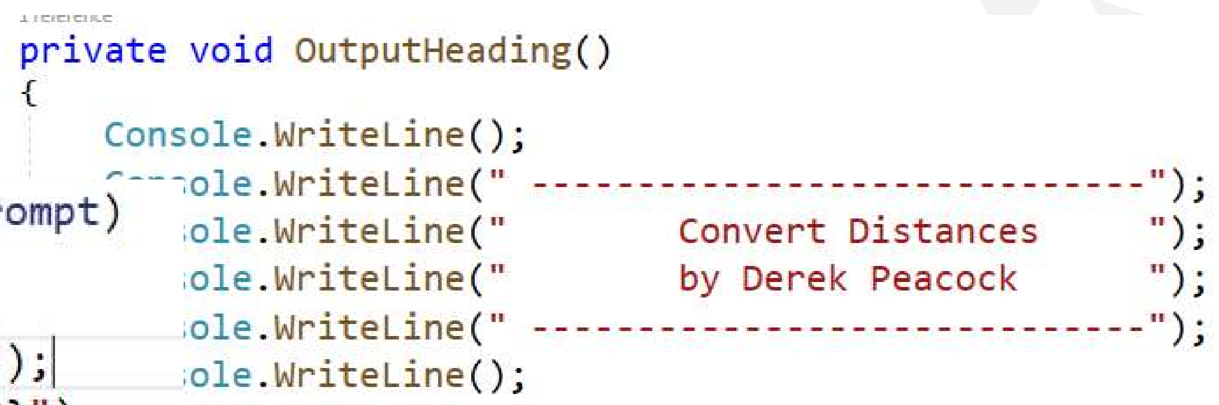
```
if (choice == "1")
{
    DistanceConverter14 converter = new Di
    converter.ConvertDistance();
}
else if (choice == "2")
{
    BMI bmi = new BMI();
    bmi.CalculateIndex();
}
else Console.WriteLine("INVALID CHOICE");
```

If the user makes an invalid choice they are not given a chance to choose again

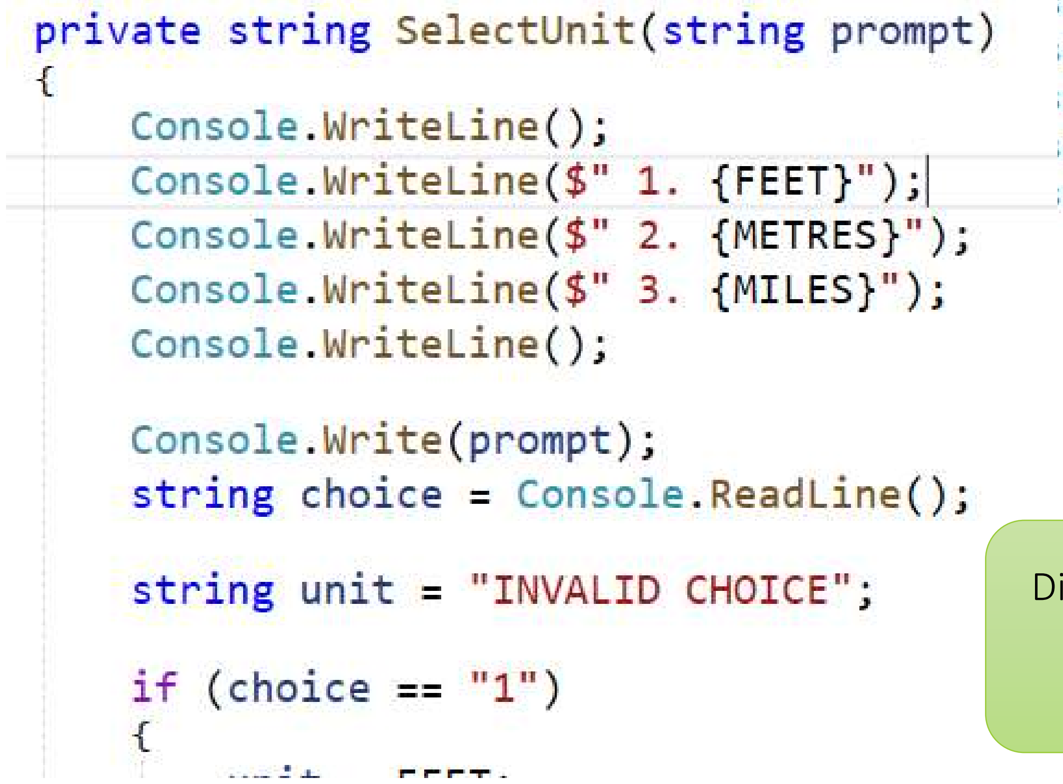
Does this remind you of other code?

DRY!!!

Select Unit & Output Heading



```
private void OutputHeading()
{
    Console.WriteLine();
    ole.WriteLine("  ----------------------------");
    ole.WriteLine("        Convert Distances     ");
    ole.WriteLine("        by Derek Peacock      ");
    ole.WriteLine("  ----------------------------");
    ole.WriteLine();
```



```
private string SelectUnit(string prompt)
{
    Console.WriteLine();
    Console.WriteLine($" 1. {FEET}");
    Console.WriteLine($" 2. {METRES}");
    Console.WriteLine($" 3. {MILES}");
    Console.WriteLine();

    Console.Write(prompt);
    string choice = Console.ReadLine();

    string unit = "INVALID CHOICE";

    if (choice == "1")
    {
        unit = FEET;
```

Displaying a list of choices and asking the user to select one

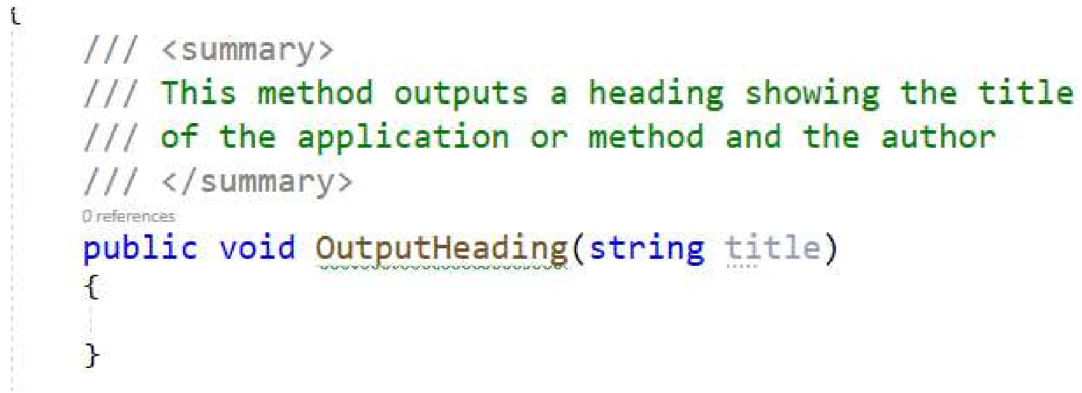
Creating UserLib Class

Can create UserLib from scratch following these slides

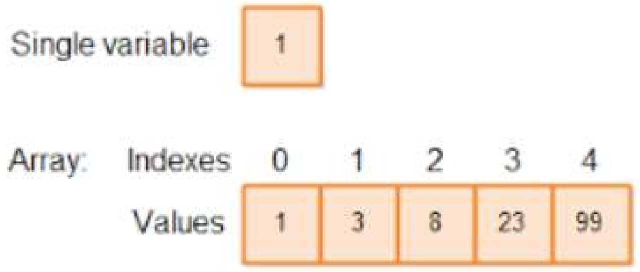
Can import the code from GitHub

https://github.com/BNU-CO453/ConsoleHelpers.git

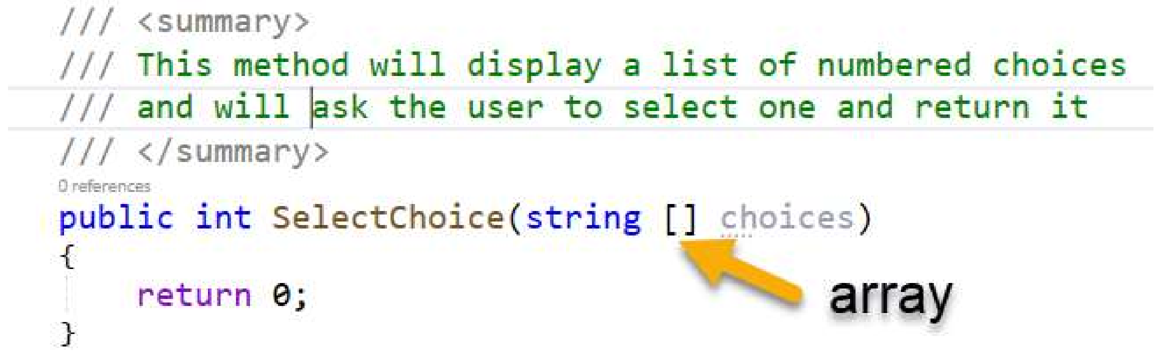
User Interface Component Methods



```
{
    /// <summary>
    /// This method outputs a heading showing the title
    /// of the application or method and the author
    /// </summary>
    0 references
    public void OutputHeading(string title)
    {

    }
```

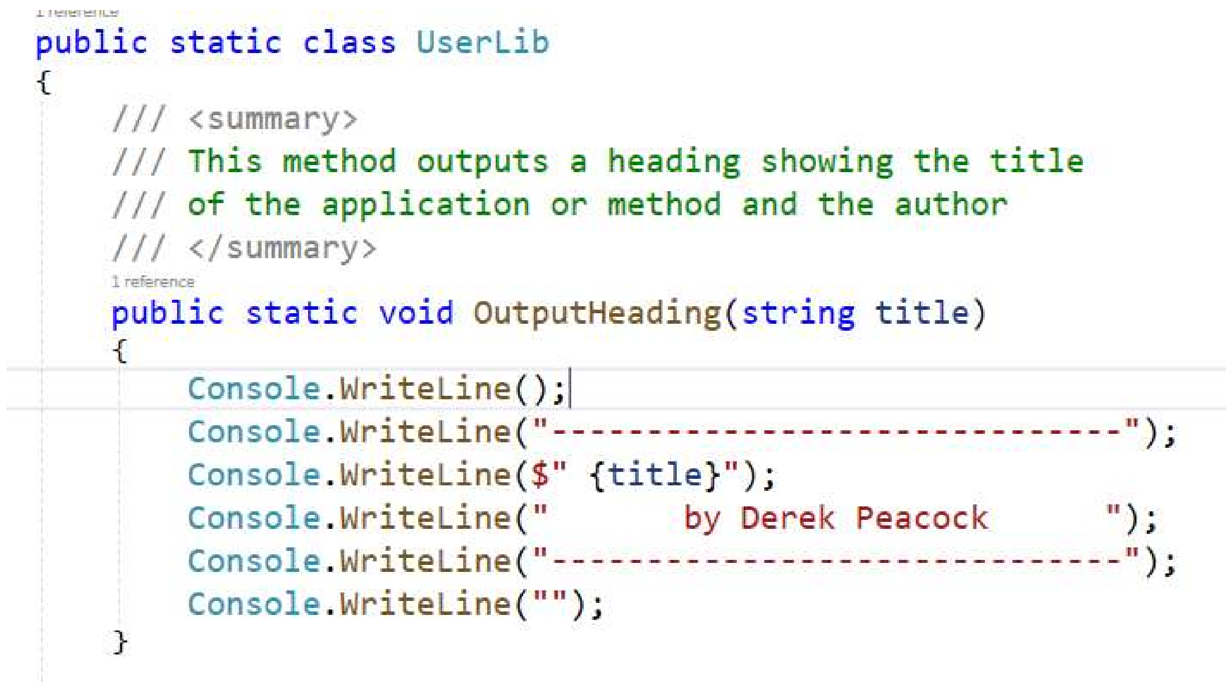


Visual Studio offers Class Library projects

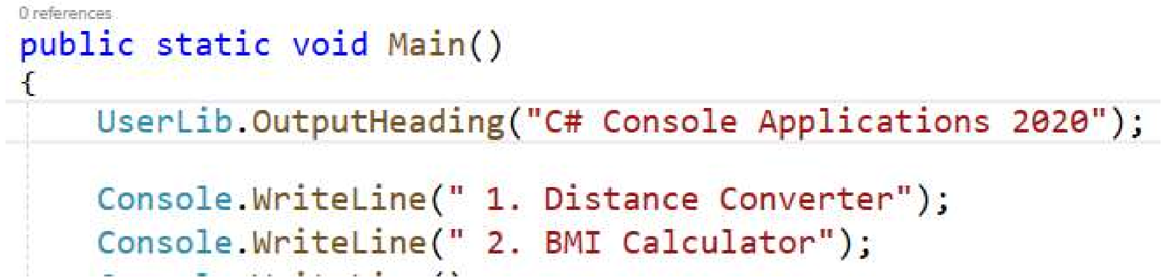


```
/// <summary>
/// This method will display a list of numbered choices
/// and will ask the user to select one and return it
/// </summary>
0 references
public int SelectChoice(string [] choices)
{
    return 0;
}
```

Static Classes – OutputHeading()



```
public static class UserLib
{
    /// <summary>
    /// This method outputs a heading showing the title
    /// of the application or method and the author
    /// </summary>
    public static void OutputHeading(string title)
    {
        Console.WriteLine();
        Console.WriteLine("-----------------------------");
        Console.WriteLine($" {title}");
        Console.WriteLine("       by Derek Peacock      ");
        Console.WriteLine("-----------------------------");
        Console.WriteLine("");
    }
```

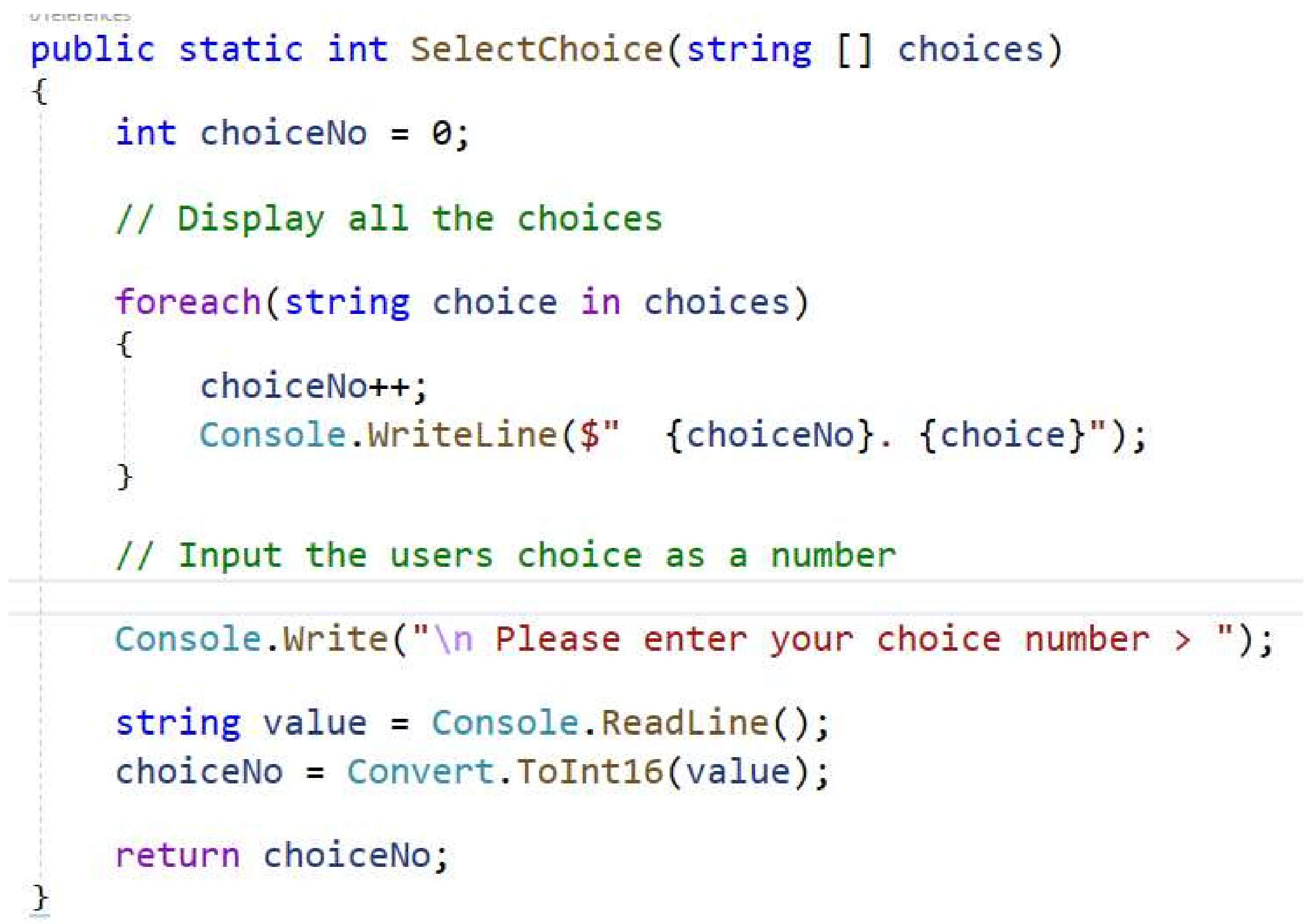


```
public static void Main()
{
    UserLib.OutputHeading("C# Console Applications 2020");

    Console.WriteLine(" 1. Distance Converter");
    Console.WriteLine(" 2. BMI Calculator");
```

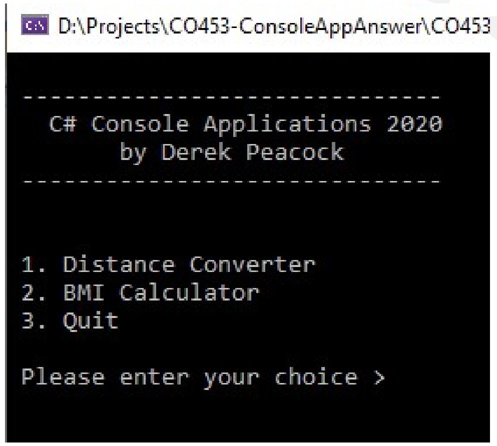
OutputHeading can be used by BMI and DistanceConverter and Program classes

Static classes can be used without creating an object

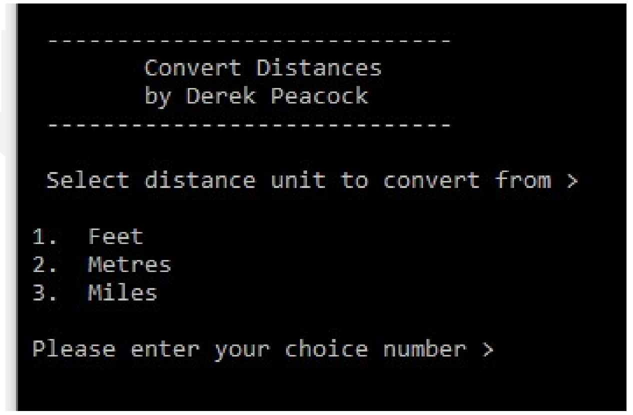
SelectChoice()



```
public static int SelectChoice(string [] choices)
{
    int choiceNo = 0;

    // Display all the choices

    foreach(string choice in choices)
    {
        choiceNo++;
        Console.WriteLine($"  {choiceNo}. {choice}");
    }

    // Input the users choice as a number

    Console.Write("\n Please enter your choice number > ");

    string value = Console.ReadLine();
    choiceNo = Convert.ToInt16(value);

    return choiceNo;
}
```

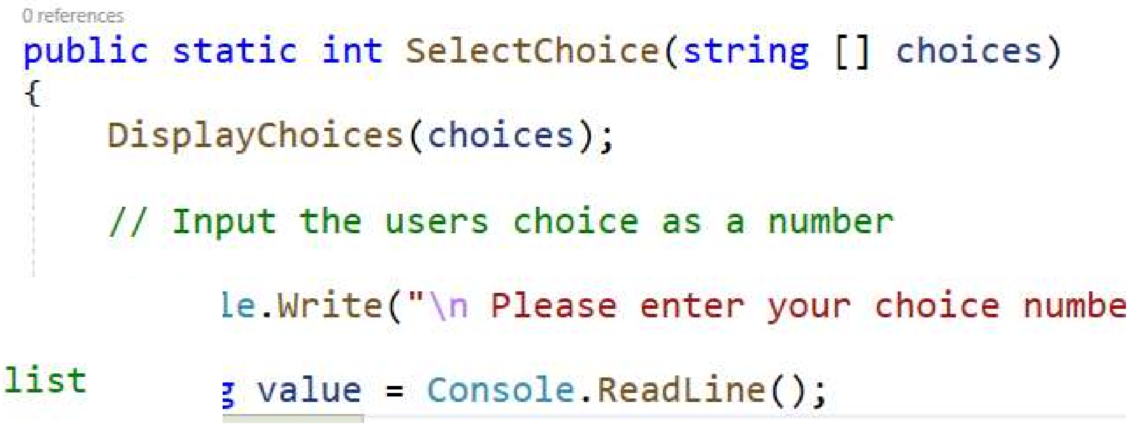


```
D:\Projects\CO453-ConsoleAppAnswer\CO453

 -----------------------------
  C# Console Applications 2020
       by Derek Peacock
 -----------------------------


 1. Distance Converter
 2. BMI Calculator
 3. Quit

 Please enter your choice >
```

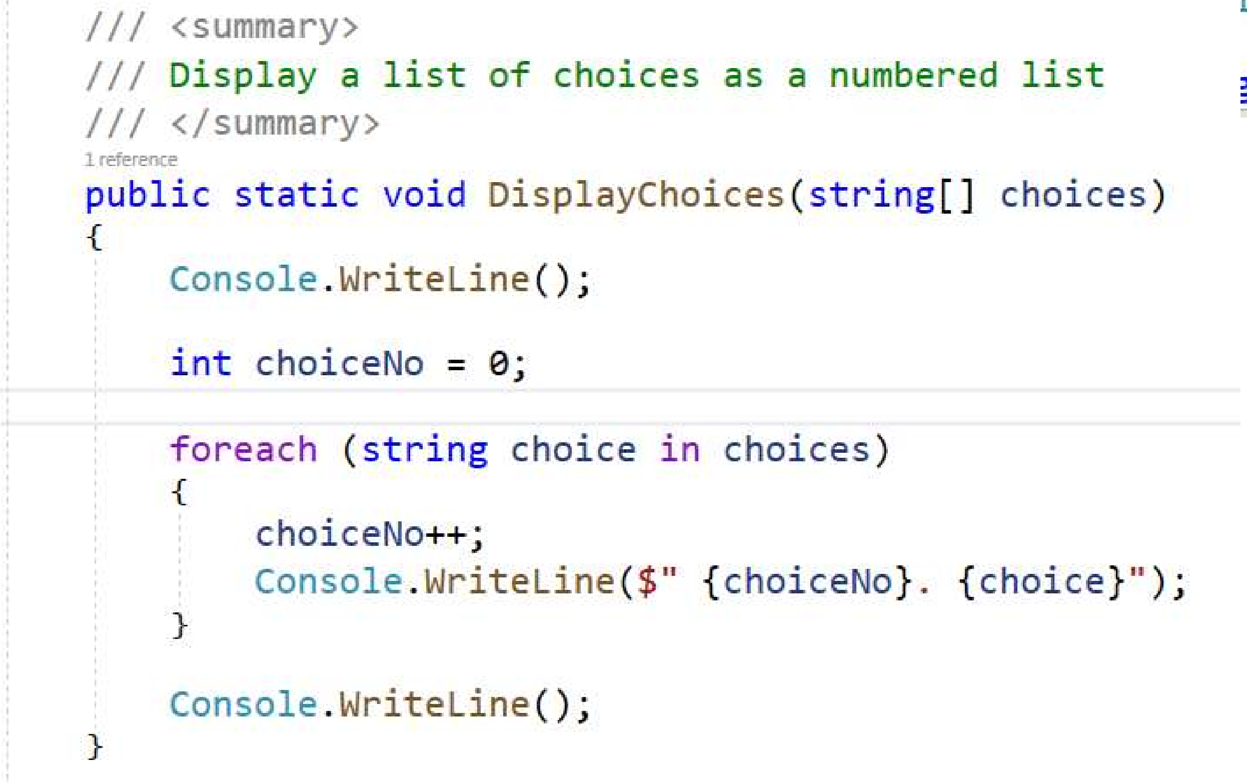


```
 -----------------------------
       Convert Distances
       by Derek Peacock
 -----------------------------

 Select distance unit to convert from >

1.   Feet
2.   Metres
3.   Miles

Please enter your choice number >
```

Separate DisplayChoices



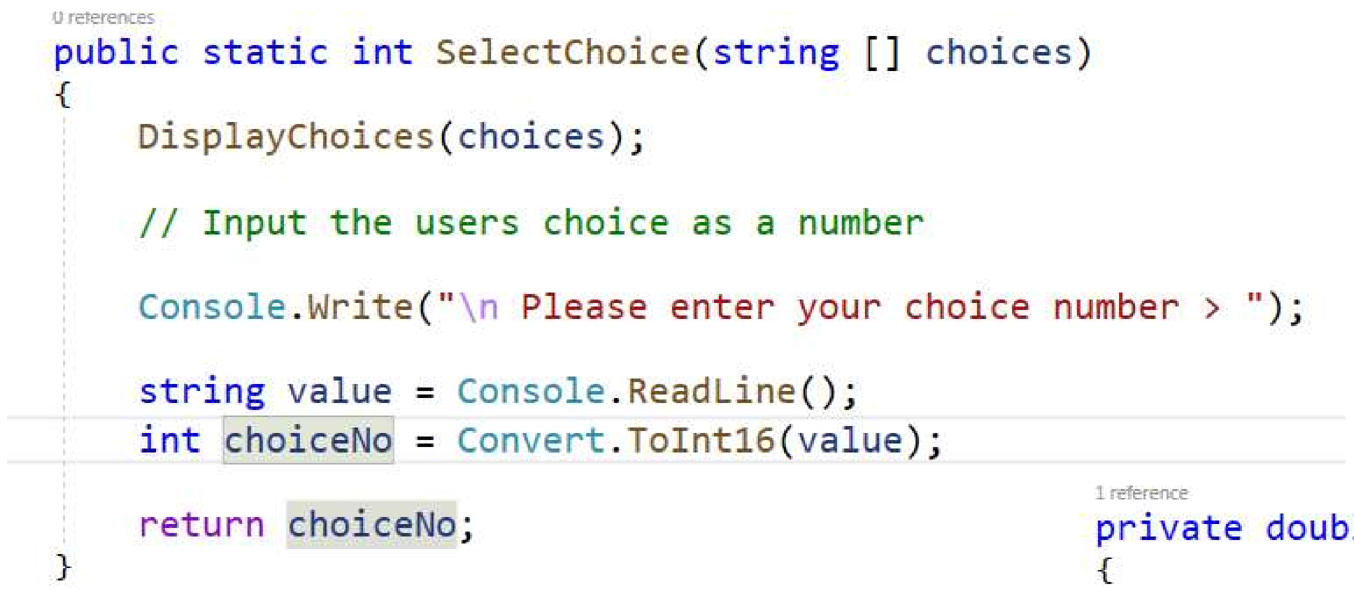
```
0 references
public static int SelectChoice(string [] choices)
{
    DisplayChoices(choices);

    // Input the users choice as a number

 le.Write("\n Please enter your choice numbe

 g value = Console.ReadLine();
```



```
/// <summary>
/// Display a list of choices as a numbered list
/// </summary>
1 reference
public static void DisplayChoices(string[] choices)
{
    Console.WriteLine();

    int choiceNo = 0;

    foreach (string choice in choices)
    {
        choiceNo++;
        Console.WriteLine($" {choiceNo}. {choice}");
    }

    Console.WriteLine();
}
```

Keep methods small and performing a single action

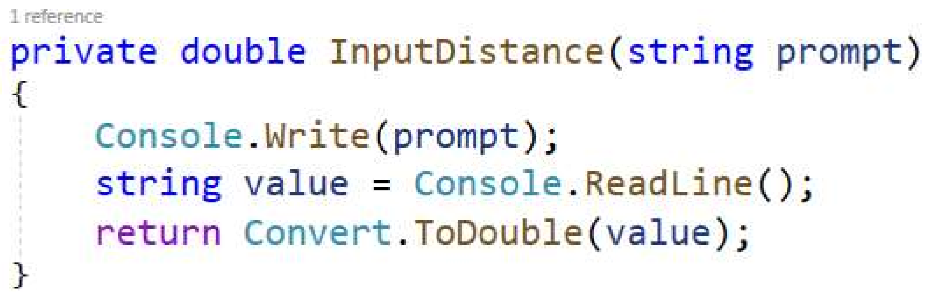
Need a Separate InputNumber()



```
0 references
public static int SelectChoice(string [] choices)
{
    DisplayChoices(choices);

    // Input the users choice as a number

    Console.Write("\n Please enter your choice number > ");

    string value = Console.ReadLine();
    int choiceNo = Convert.ToInt16(value);

    return choiceNo;
}
```

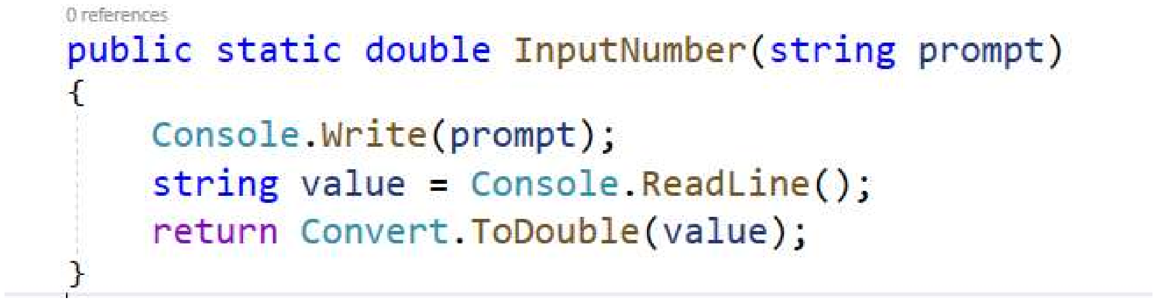
DRY: CODE DUPLICATION!!!

Integer is a subset of double!



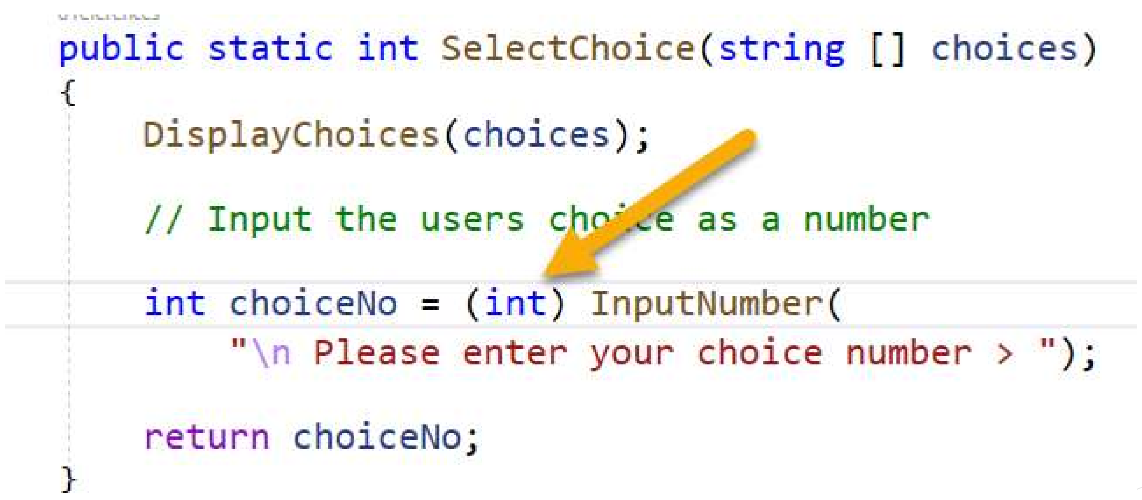
```
1 reference
private double InputDistance(string prompt)
{
    Console.Write(prompt);
    string value = Console.ReadLine();
    return Convert.ToDouble(value);
}
```

InputNumber() Version 1.0



```
0 references
public static double InputNumber(string prompt)
{
    Console.Write(prompt);
    string value = Console.ReadLine();
    return Convert.ToDouble(value);
}
```

InputNumber returns a double which is cast to an integer.



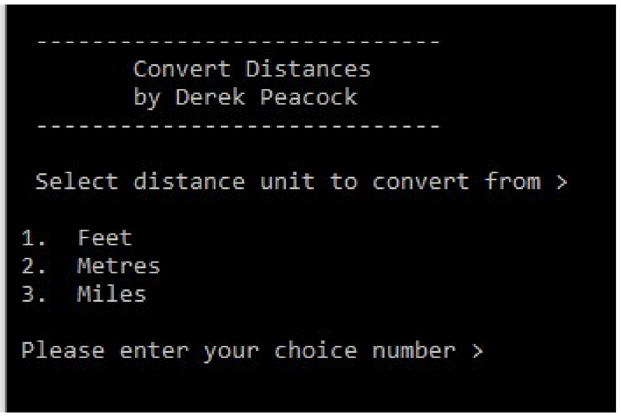
```
public static int SelectChoice(string [] choices)
{
    DisplayChoices(choices);

    // Input the users choice as a number

    int choiceNo = (int) InputNumber(
        "\n Please enter your choice number > ");

    return choiceNo;
}
```

DistanceConverter

SelectUnit now uses SelectChoice but issues remain. What are they??

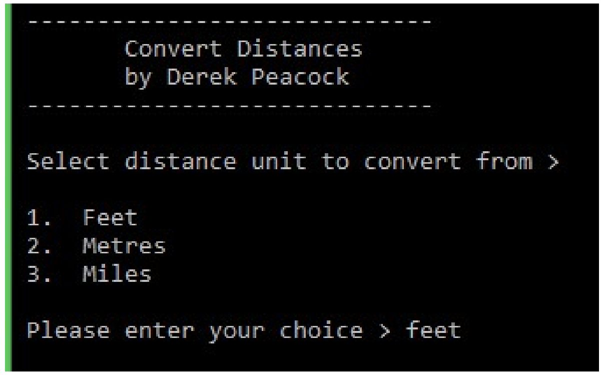
```
private DistanceUnit SelectUnit(string prompt)
{
    Console.WriteLine(prompt);

    string[] choices = { $" {DistanceUnit.Feet}",
                         $" {DistanceUnit.Metres}",
                         $" {DistanceUnit.Miles}"};

    int choice = UserLib.SelectChoice(choices);
```

```
private DistanceUnit SelectUnit(string prompt)
{
    Console.WriteLine($" 1. {DistanceUnit.Feet}");
    Console.WriteLine($" 2. {DistanceUnit.Metres}");
    Console.WriteLine($" 3. {DistanceUnit.Miles}");

    Console.Write(prompt);
    string choice = Console.ReadLine();
```



```
 ----------------------------
       Convert Distances
       by Derek Peacock
 ----------------------------

 Select distance unit to convert from >

1.   Feet
2.   Metres
3.   Miles

Please enter your choice number >
```

Remaining Issues with InputNumber()



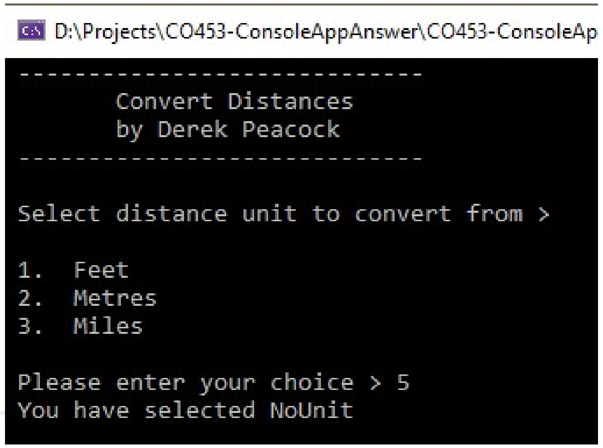
```
 ----------------------------
       Convert Distances
       by Derek Peacock
 ----------------------------

 Select distance unit to convert from >

 1.   Feet
 2.   Metres
 3.   Miles

 Please enter your choice > feet
```

Exception 1

If the user enters an invalid integer

Exception 2

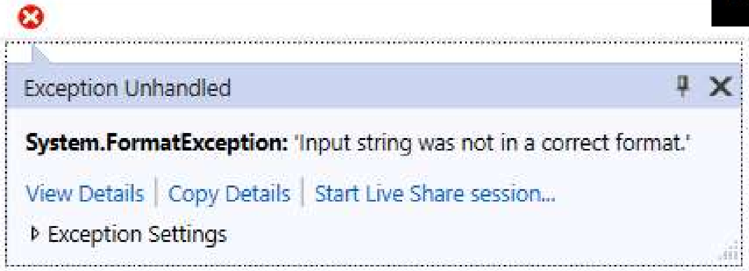
If the user enters an integer outside the range



D:\Projects\CO453-ConsoleAppAnswer\CO453-ConsoleAp

```
 ----------------------------
       Convert Distances
       by Derek Peacock
 ----------------------------

 Select distance unit to convert from >

 1.   Feet
 2.   Metres
 3.   Miles

 Please enter your choice > 5
 You have selected NoUnit
```

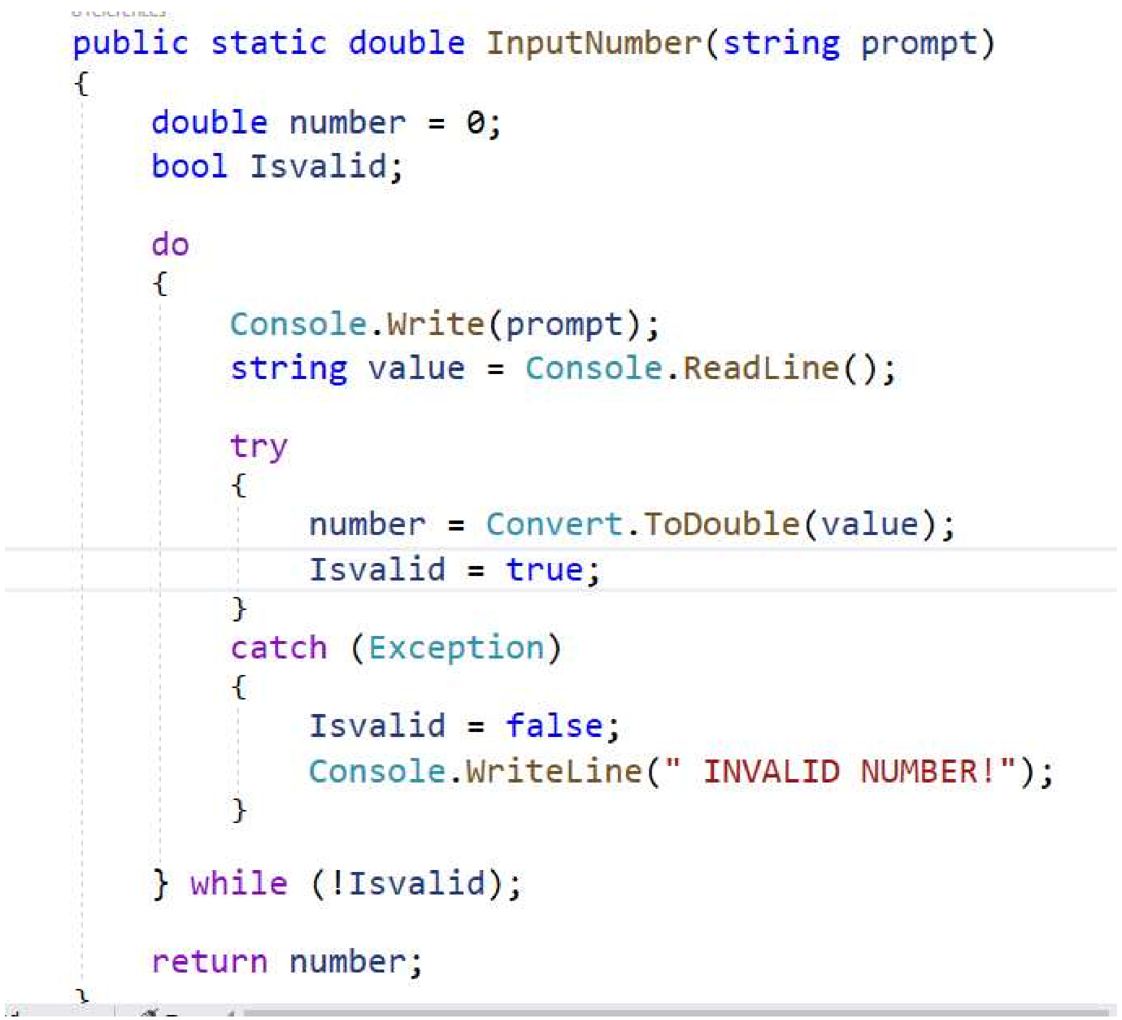
```
Console.Write(prompt);
string value = Console.ReadLine();
number = Convert.ToDouble(value);
```



Exception Unhandled

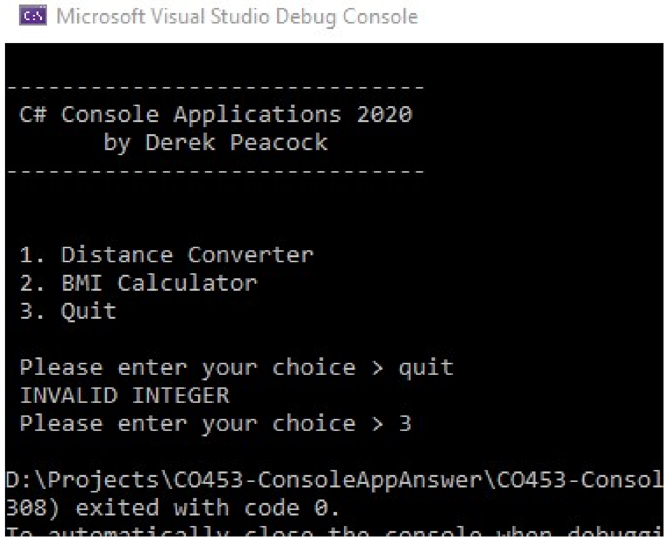
System.FormatException: 'Input string was not in a correct format.'

UserLib.InputNumber()



```
public static double InputNumber(string prompt)
{
    double number = 0;
    bool Isvalid;

    do
    {
        Console.Write(prompt);
        string value = Console.ReadLine();

        try
        {
            number = Convert.ToDouble(value);
            Isvalid = true;
        }
        catch (Exception)
        {
            Isvalid = false;
            Console.WriteLine(" INVALID NUMBER!");
        }

    } while (!Isvalid);

    return number;
}
```

This method will always return a valid double

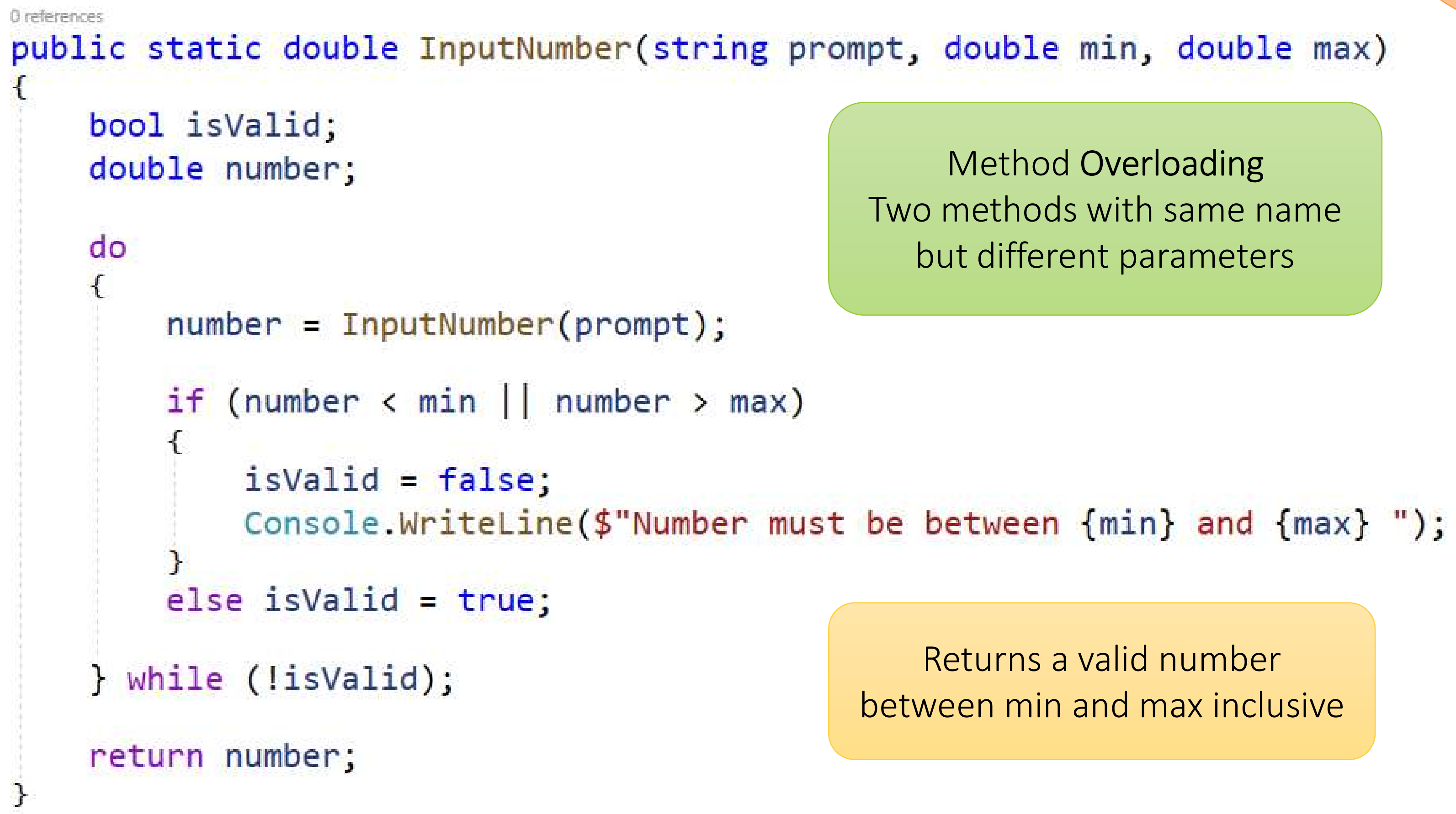


Microsoft Visual Studio Debug Console

```
------------------------------
 C# Console Applications 2020
       by Derek Peacock
------------------------------


 1. Distance Converter
 2. BMI Calculator
 3. Quit

 Please enter your choice > quit
 INVALID INTEGER
 Please enter your choice > 3

D:\Projects\CO453-ConsoleAppAnswer\CO453-Consol
308) exited with code 0.
To automatically close the console when debuggi
```

2nd InputNumber with a range check

This InputNumber uses the other InputNumber

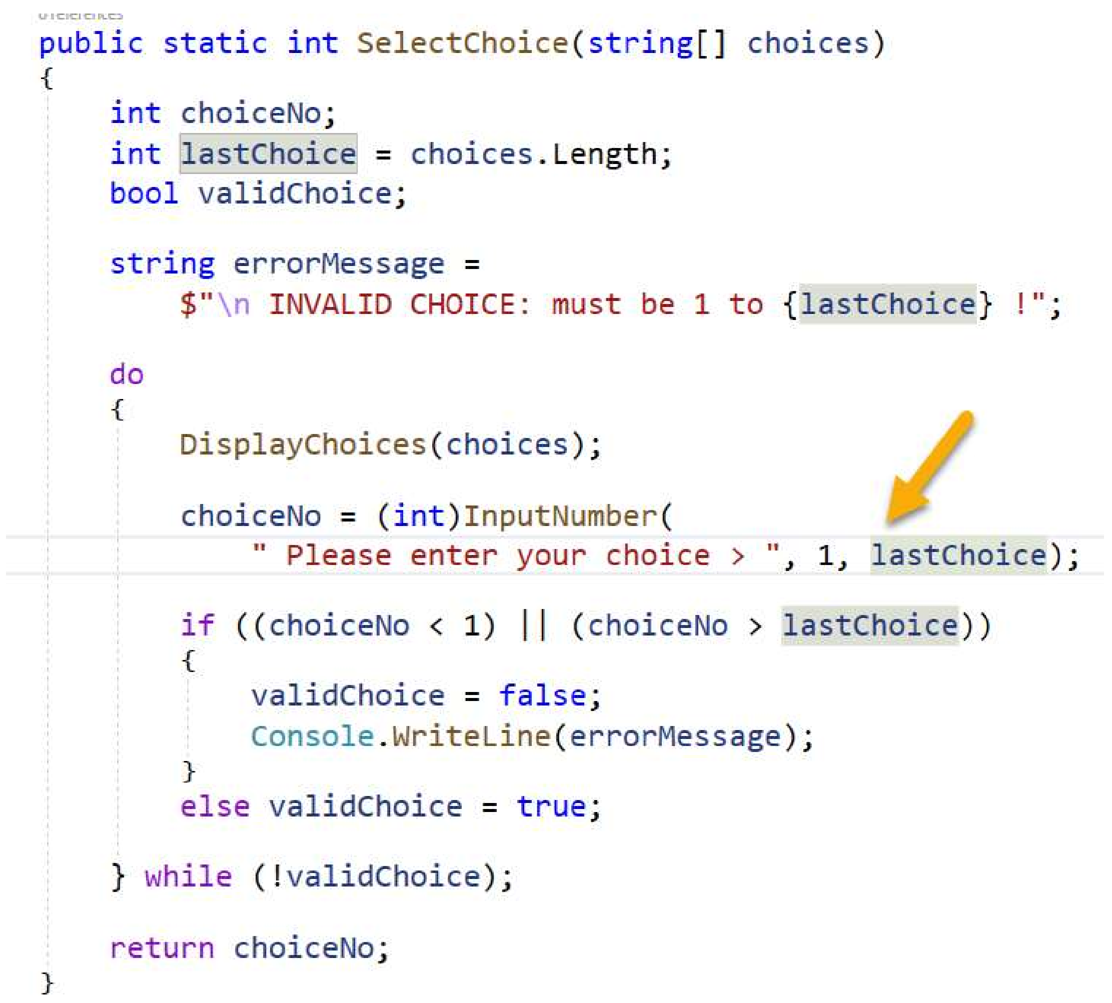


```
0 references
public static double InputNumber(string prompt, double min, double max)
{
    bool isValid;
    double number;

    do
    {
        number = InputNumber(prompt);

        if (number < min || number > max)
        {
            isValid = false;
            Console.WriteLine($"Number must be between {min} and {max} ");
        }
        else isValid = true;

    } while (!isValid);

    return number;
}
```

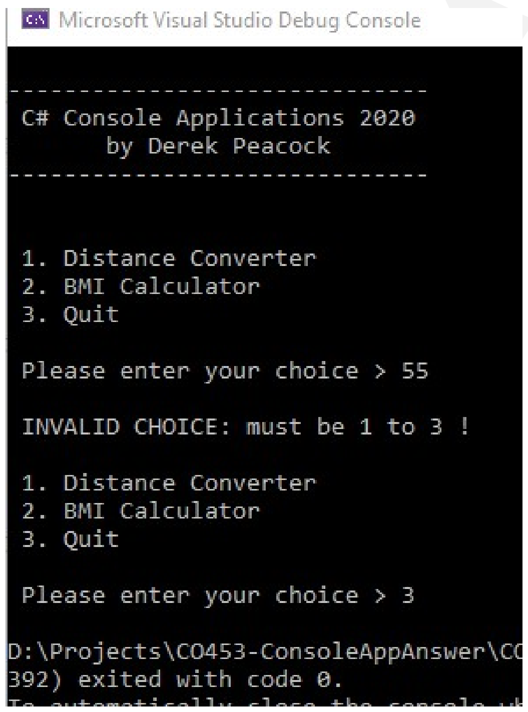
Method **Overloading**
Two methods with same name but different parameters

Returns a valid number between min and max inclusive

Ensuring SelectChoice() is always Valid



```
public static int SelectChoice(string[] choices)
{
    int choiceNo;
    int lastChoice = choices.Length;
    bool validChoice;

    string errorMessage =
        $"\n INVALID CHOICE: must be 1 to {lastChoice} !";

    do
    {
        DisplayChoices(choices);

        choiceNo = (int)InputNumber(
            " Please enter your choice > ", 1, lastChoice);

        if ((choiceNo < 1) || (choiceNo > lastChoice))
        {
            validChoice = false;
            Console.WriteLine(errorMessage);
        }
        else validChoice = true;

    } while (!validChoice);

    return choiceNo;
}
```



Microsoft Visual Studio Debug Console

```
------------------------------
C# Console Applications 2020
      by Derek Peacock
------------------------------


1. Distance Converter
2. BMI Calculator
3. Quit

Please enter your choice > 55

INVALID CHOICE: must be 1 to 3 !

1. Distance Converter
2. BMI Calculator
3. Quit

Please enter your choice > 3

D:\Projects\CO453-ConsoleAppAnswer\C
392) exited with code 0.
```

choiceNo returned is now always valid!

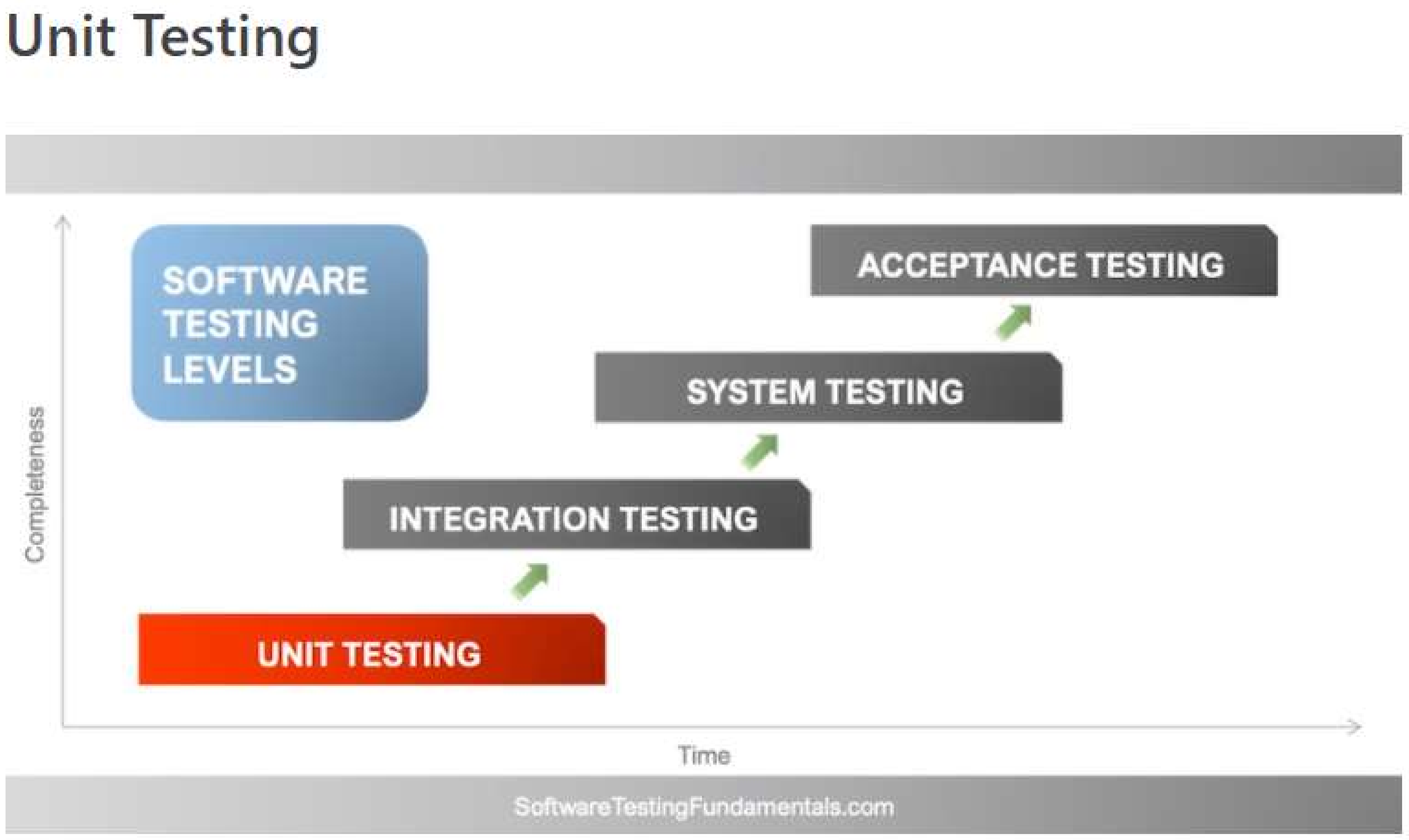
Refactoring Summary

Added a Library Class

Contains a fool proof method for entering valid numbers

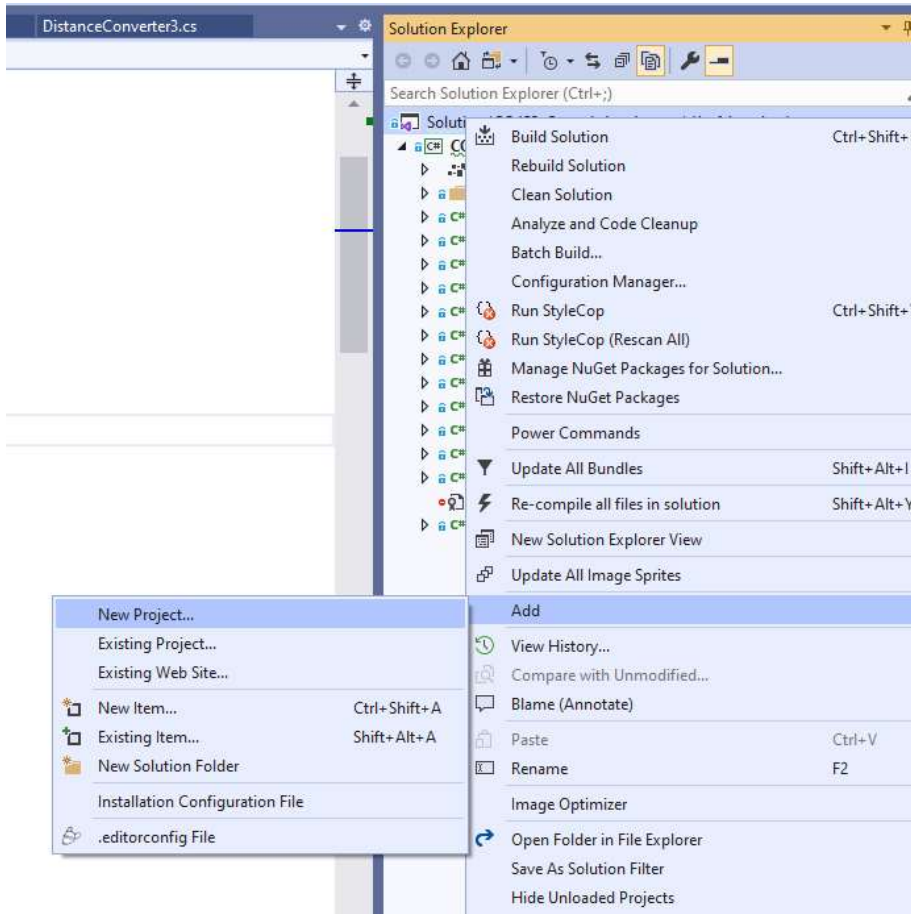
Contains a method for selecting one of an array of choices

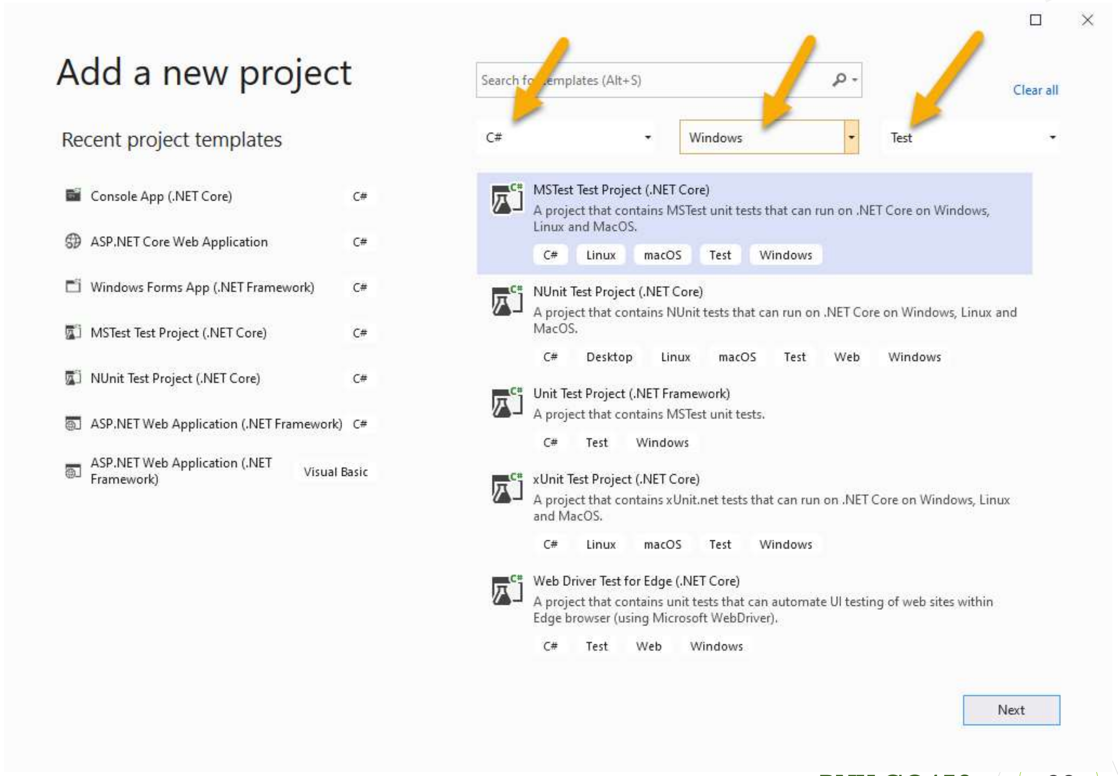
Unit Testing (Automated Testing)



Automated Testing: MS Unit Tests

- Right click on Solution and Add -> New Project
- Select C#, Windows, Test
- Select MSTest (.NET Core)





Add a new project

Recent project templates

Console App (.NET Core)	C#
ASP.NET Core Web Application	C#
Windows Forms App (.NET Framework)	C#
MSTest Test Project (.NET Core)	C#
NUnit Test Project (.NET Core)	C#
ASP.NET Web Application (.NET Framework)	C#
ASP.NET Web Application (.NET Framework)	Visual Basic

Search for templates (Alt+S)

Clear all

C# · Windows · Test

MSTest Test Project (.NET Core)
A project that contains MSTest unit tests that can run on .NET Core on Windows, Linux and MacOS.
C# Linux macOS Test Windows

NUnit Test Project (.NET Core)
A project that contains NUnit tests that can run on .NET Core on Windows, Linux and MacOS.
C# Desktop Linux macOS Test Web Windows

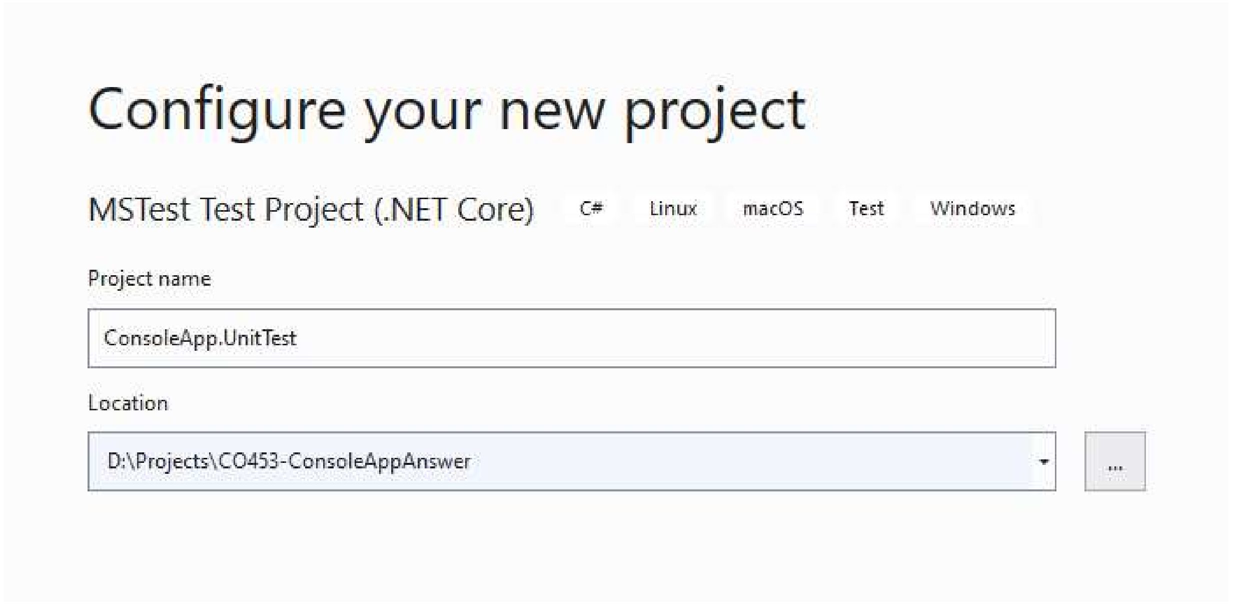
Unit Test Project (.NET Framework)
A project that contains MSTest unit tests.
C# Test Windows

xUnit Test Project (.NET Core)
A project that contains xUnit.net tests that can run on .NET Core on Windows, Linux and MacOS.
C# Linux macOS Test Windows

Web Driver Test for Edge (.NET Core)
A project that contains unit tests that can automate UI testing of web sites within Edge browser (using Microsoft WebDriver).
C# Test Web Windows

Next

Adding MS Unit Test Project



Configure your new project

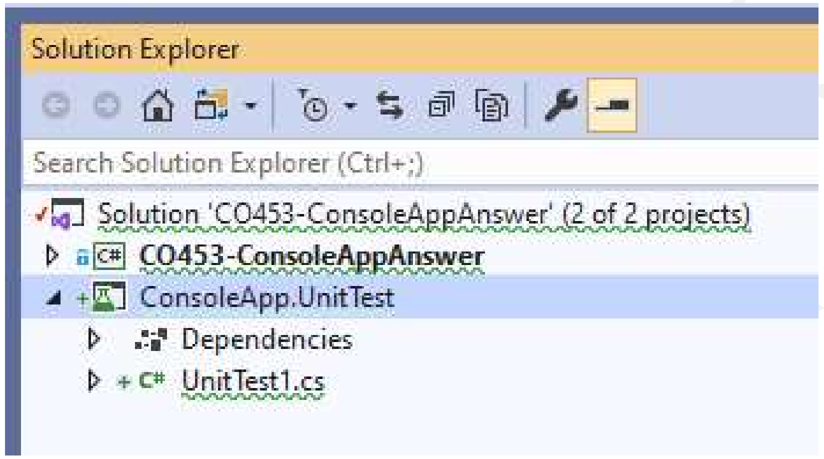
MSTest Test Project (.NET Core) C# Linux macOS Test Windows

Project name

ConsoleApp.UnitTest

Location

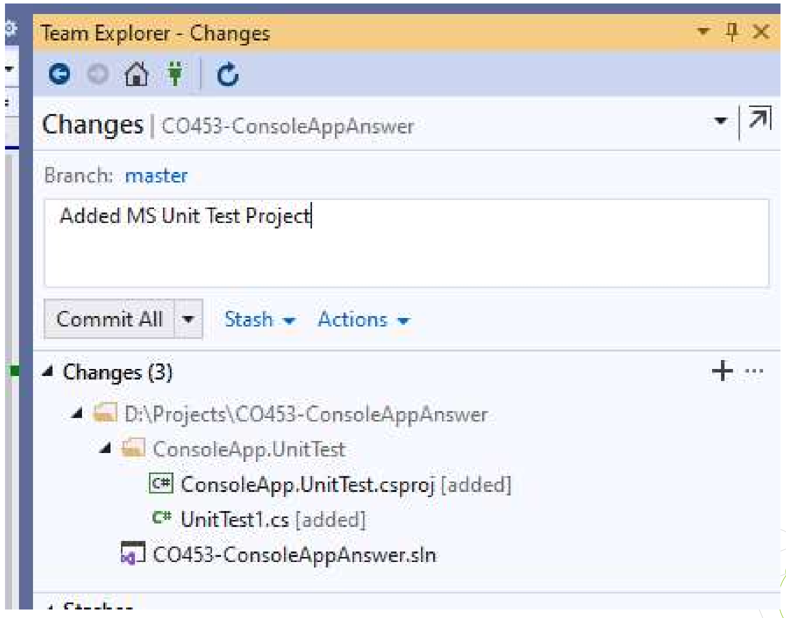
D:\Projects\CO453-ConsoleAppAnswer



Solution Explorer

Search Solution Explorer (Ctrl+;)

```
Solution 'CO453-ConsoleAppAnswer' (2 of 2 projects)
  CO453-ConsoleAppAnswer
  ConsoleApp.UnitTest
    Dependencies
    UnitTest1.cs
```



Team Explorer - Changes

Changes | CO453-ConsoleAppAnswer

Branch: master

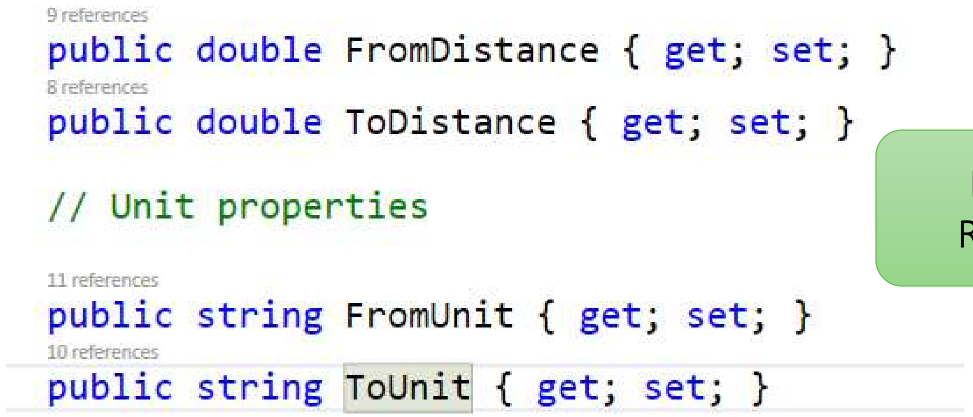
Added MS Unit Test Project

Commit All   Stash   Actions

```
Changes (3)
  D:\Projects\CO453-ConsoleAppAnswer
    ConsoleApp.UnitTest
      ConsoleApp.UnitTest.csproj [added]
      UnitTest1.cs [added]
    CO453-ConsoleAppAnswer.sln
```

```
[TestClass]
0 references
public class UnitTest1
{
    [TestMethod]
    0 references
    public void TestMethod1()
    {
    }
}
```

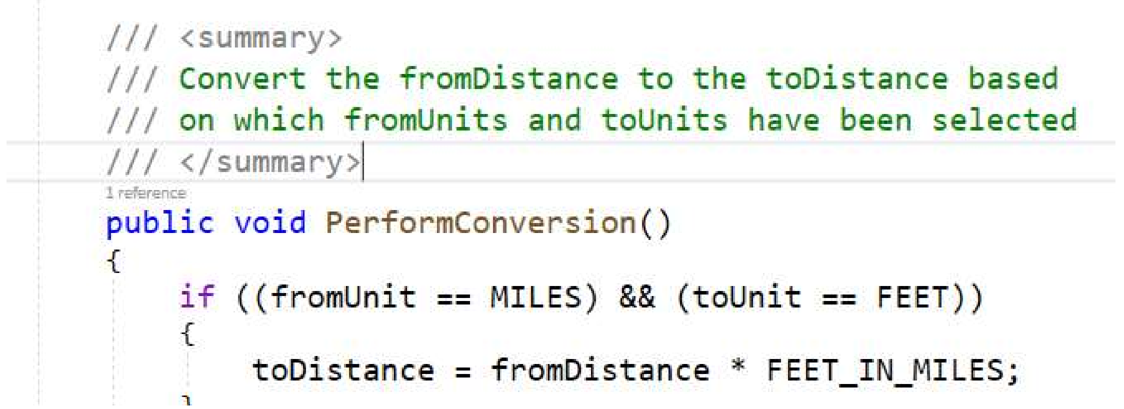
Change private to public access



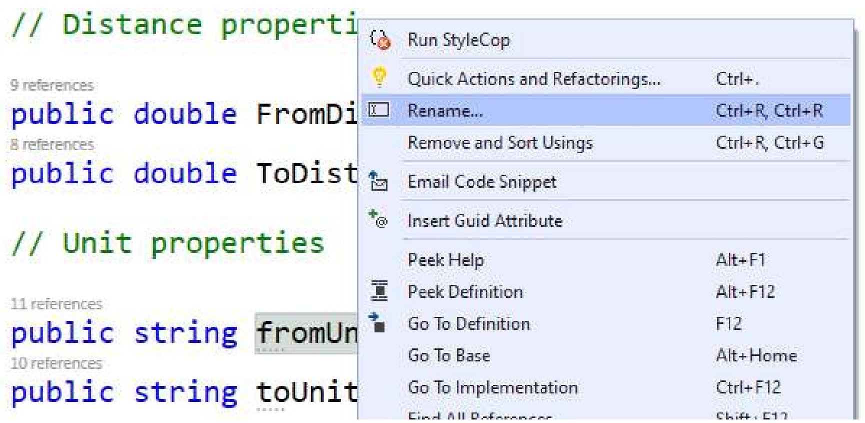
```
9 references
public double FromDistance { get; set; }
8 references
public double ToDistance { get; set; }

// Unit properties

11 references
public string FromUnit { get; set; }
10 references
public string ToUnit { get; set; }
```

Must use Rename!!!

private attributes become public properties



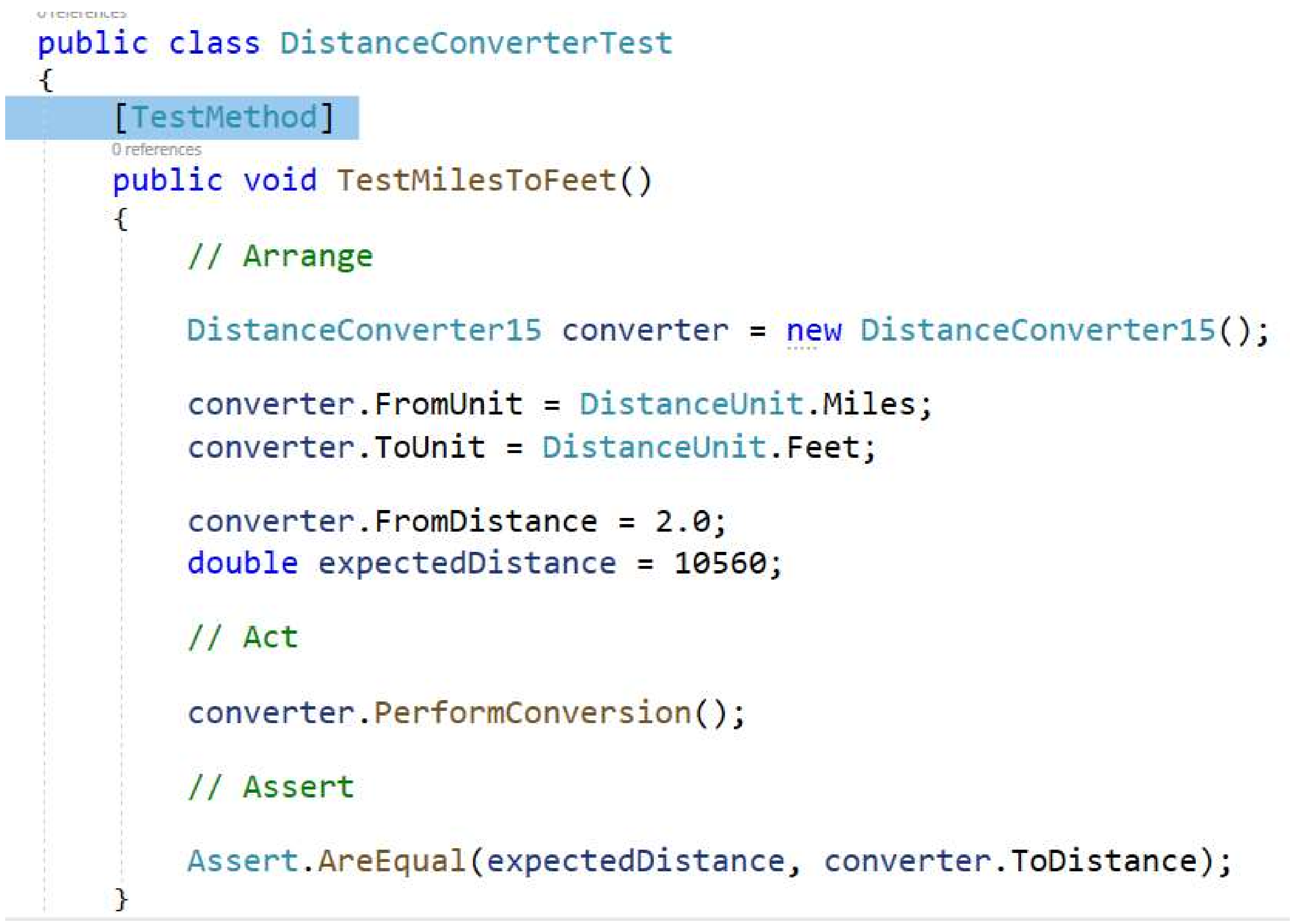
```
/// <summary>
/// Convert the fromDistance to the toDistance based
/// on which fromUnits and toUnits have been selected
/// </summary>
1 reference
public void PerformConversion()
{
    if ((fromUnit == MILES) && (toUnit == FEET))
    {
        toDistance = fromDistance * FEET_IN_MILES;
```



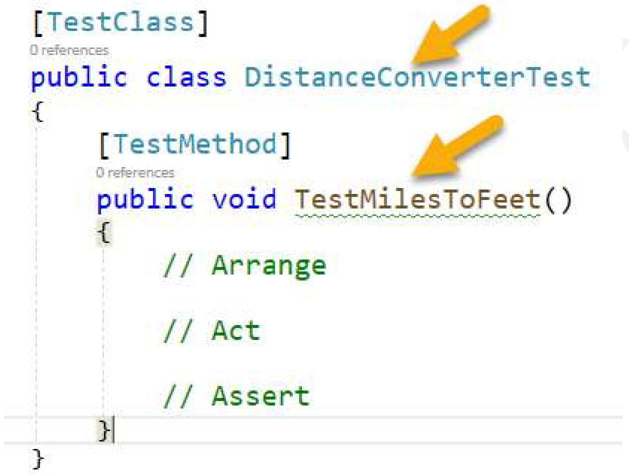
```
// Distance properti

9 references
public double FromDi
8 references
public double ToDist

// Unit properties

11 references
public string fromUn
10 references
public string toUnit
```

This method is separated from ConvertDistance

TestMilesToFeet()



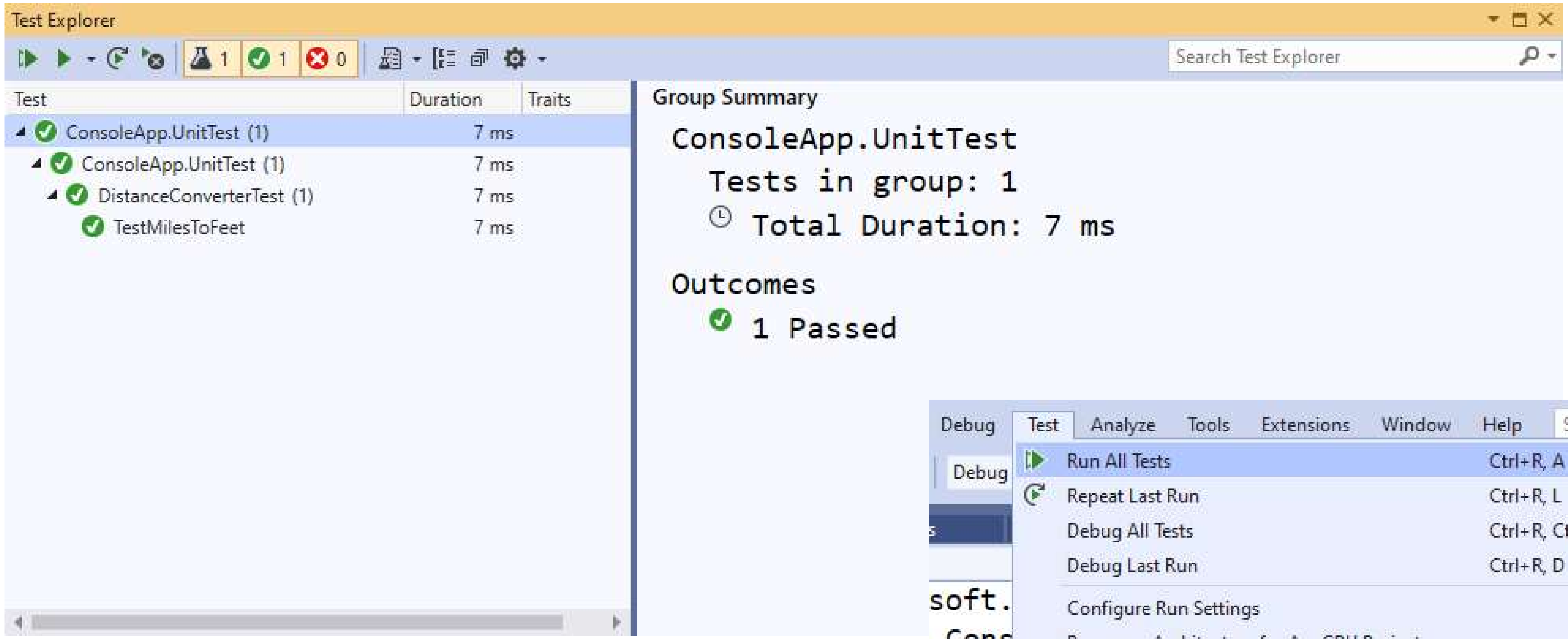
```
0 references
public class DistanceConverterTest
{
    [TestMethod]
    0 references
    public void TestMilesToFeet()
    {
        // Arrange

        DistanceConverter15 converter = new DistanceConverter15();

        converter.FromUnit = DistanceUnit.Miles;
        converter.ToUnit = DistanceUnit.Feet;

        converter.FromDistance = 2.0;
        double expectedDistance = 10560;

        // Act

        converter.PerformConversion();

        // Assert

        Assert.AreEqual(expectedDistance, converter.ToDistance);
    }
```



```
[TestClass]
0 references
public class DistanceConverterTest
{
    [TestMethod]
    0 references
    public void TestMilesToFeet()
    {
        // Arrange

        // Act

        // Assert
    }
}
```

All 6 conversions can be tested and easily re-tested

Running Tests

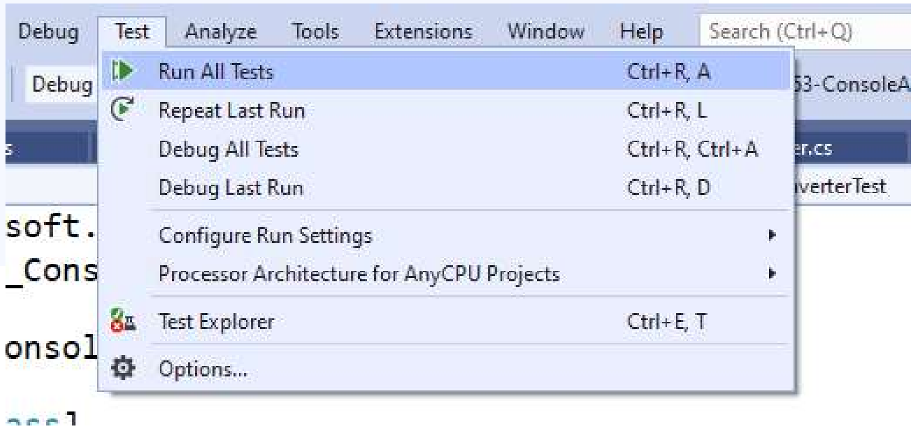


Test Explorer

Search Test Explorer

Test	Duration	Traits
ConsoleApp.UnitTest (1)	7 ms	
ConsoleApp.UnitTest (1)	7 ms	
DistanceConverterTest (1)	7 ms	
TestMilesToFeet	7 ms	

```
Group Summary
 ConsoleApp.UnitTest
   Tests in group: 1
   Total Duration: 7 ms

 Outcomes
   1 Passed
```



Week 3: Independent Study

App01 Distance Converter

Complete making full use of UserLib and Complete all six unit-tests and document

App02 BMI Calculator

Complete making full use of UserLib, test fully and document