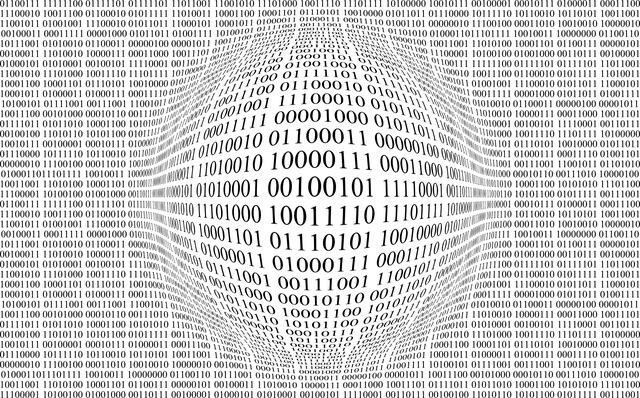
Sharp Q’s: Learning C# Through Conway’s Game of Life Part 2



Contents

[Lesson 1 – Starting Our Calculator App 4](#_Toc513812692)

[What We Are Going To Building in Part 2 4](#_Toc513812693)

[Practice 4](#_Toc513812694)

[Lesson 2 – Loops 4](#_Toc513812695)

[Types of loops in c# 4](#_Toc513812696)

[While loop 4](#_Toc513812697)

[Foreach Loop 5](#_Toc513812698)

[For 5](#_Toc513812699)

[Further Learning 5](#_Toc513812700)

[Lesson 3 – Debugging your code! 5](#_Toc513812701)

[What Is Debugging 5](#_Toc513812702)

[Further Learning 6](#_Toc513812703)

[Lesson 4 – Breaking Out of Loops Early 6](#_Toc513812704)

[Continue 6](#_Toc513812705)

[Break 6](#_Toc513812706)

[Further Reading 6](#_Toc513812707)

[Lesson 5 – What Is a Method 6](#_Toc513812708)

[What is a method 7](#_Toc513812709)

[Further Reading 7](#_Toc513812710)

[Lesson 6 – Classes 7](#_Toc513812711)

[What Are Classes? 7](#_Toc513812712)

[What Does a Class Look Like? 7](#_Toc513812713)

[Further Reading 8](#_Toc513812714)

[Lesson 7 – Access Modifiers 8](#_Toc513812715)

[What Is an Access Modifier 9](#_Toc513812716)

[Further Reading 9](#_Toc513812717)

[Lesson 8 – Creating a New Class for our Calculator 9](#_Toc513812718)

[Creating the File 9](#_Toc513812719)

[Further Reading 11](#_Toc513812720)

[Lesson 9 – If, Else, and Switch 11](#_Toc513812721)

[Further Reading 12](#_Toc513812722)

[Lesson 10 – Looping the user Input 12](#_Toc513812723)

[Infinite Loops 12](#_Toc513812724)

[Our Menu Loop 12](#_Toc513812725)

[Hints 13](#_Toc513812726)

[Further Reading 13](#_Toc513812727)

[Lesson 11 - 13](#_Toc513812728)

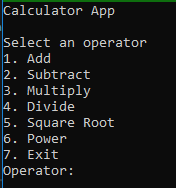
# Lesson 1 – Starting Our Calculator App

## What We Are Going To Building in Part 2

In part 2 of this course, we are going to be introducing Loops, classes, and methods, while building a command line calculator application. This part of the course is going to get more advanced, so make sure you have a solid base of knowledge from part 1 to build on!

## Practice

To start, open Calculator\_1 in the Part2 solution, and let’s build a simple menu system for the user to choose between Addition, Subtraction, Multiplication, Division, Square Root, and Power. Give them a choice to quit as well. Don’t worry about how to implement these operations, for now just make sure that when they launch the application they have these options. It should look something like the image below when you are done. Remember your Console.WriteLine and Console.ReadLine knowledge from part 1! Also, make sure you are converting the string input from ReadLine into an integer!



# Lesson 2 – Loops

## Types of loops in c#

C# has a few different types of loops to work with. The most common are while, for, and foreach.

## While loop

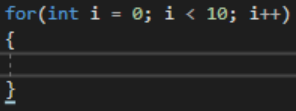
The while loop is a conditional loop that will continually execute as long as the condition inside evaluates to true. See the further reading section for examples and how to’s.

## Foreach Loop

The Foreach loop iterates over every item of a list, hence the name for each.

## For

The for loop iterates as long as a given condition is true. There are three distinct parts of the for loop. The first part runs once when the loop is encountered, the second part is the condition to evaluate, and the third and final part runs after each iteration of the loop. In the example below:



The first part of this is int i = 0; This creates a variable of type integer named i and sets it to zero. This is very common in for loops, which generally define i as a shorthand for ‘index’. The second part, i < 10, is the condition the loop evaluates. You can read this as “If i is less than 10”. The third part, i++, is shorthand for saying i = i + 1; It simply increments i by one after each loop.

## Further Learning

<https://www.youtube.com/watch?v=SPE_HXHwJSU>

<https://www.youtube.com/watch?v=eIsGXwisKPE>

<https://www.youtube.com/watch?v=2zbFaDbADsQ>

<https://www.tutorialspoint.com/csharp/csharp_loops.htm>

# Lesson 3 – Debugging your code!

## What Is Debugging

Debugging is an ***extremely*** important part of finding and fixing bugs. Hence, de-bugging! Debugging could be its own course on its own, and I encourage you to spend as much time as possible learning all the in’s and out of debugging!

## Further Learning

<http://csharp.net-tutorials.com/debugging/introduction/>

<https://www.youtube.com/watch?v=u-HdLtqEOog>

<https://www.dotnetperls.com/debugging>

# Lesson 4 – Breaking Out of Loops Early

## Continue

Sometimes, you will want to continue to the next iteration of a loop without completing the current iteration. Let’s say you have a for loop and want to print only the multiples of three. You could use the Modulus operator (%), which returns the remainder of division, to check if the current number is a multiple of three, and the continue operator to continue to the next iteration if it is not.

## Break

Sometimes, you want to exit the loop completely. Let’s say you are looking for user input that is either a Y or an N (a common yes or no answer). You can use the break to exit a loop once you have gotten the proper result! This is a good way to validate input from the command line.

## Further Reading

<https://stackoverflow.com/questions/6414/c-sharp-loop-break-vs-continue>

<https://teamtreehouse.com/community/what-is-the-difference-between-break-and-continue-in-c>

<https://www.tutorialspoint.com/csharp/csharp_continue_statement.htm>

<https://www.c-sharpcorner.com/blogs/break-and-continue-statement-in-c-sharp1>

# Lesson 5 – What Is a Method

## What is a method

A method is a way to reuse code. In programming, there is the DRY concept, which stands for Don’t Repeat Yourself. Methods allow you to write code once and call it many places. You have already used methods even if you didn’t know it! The WriteLine and ReadLine calls you made where actually calls to methods inside of the Console class! When doing your further reading on methods, make sure to look up local functions, which were introduced in c# 7!

## Further Reading

<https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/methods>

<https://www.c-sharpcorner.com/article/C-Sharp-methods/>

<https://www.tutorialspoint.com/csharp/csharp_methods.htm>

<https://www.youtube.com/watch?v=7JBpDZL0lfk>

# Lesson 6 – Classes

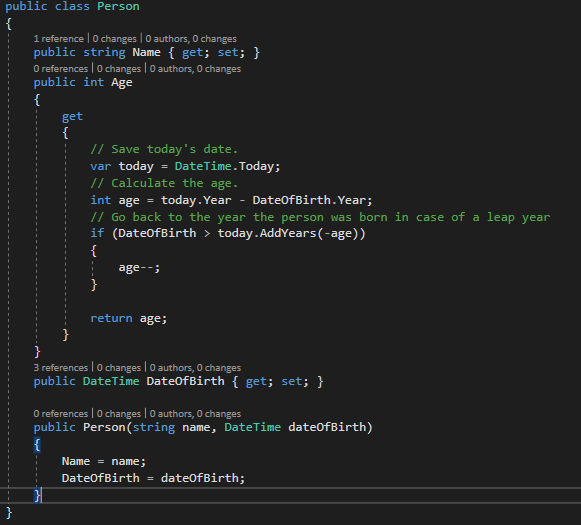
## What Are Classes?

MS Docs says:

A class is a construct that enables you to create your own custom types by grouping together variables of other types, methods and events. A class is like a blueprint. It defines the data and behavior of a type.

## What Does a Class Look Like?

Below is a sample of a Person class with three properties. See if you can figure out what the Age property does!



You will notice that Name and DateOfBirth have what we call Auto Getter and Setter. For Age, we implemented our own Get operation.

## Further Reading

<https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/classes>

<https://www.tutorialspoint.com/csharp/csharp_classes.htm>

<https://www.youtube.com/watch?v=OUV4LWcLclo>

<https://www.youtube.com/watch?v=s2hHjpZaSyI>

# Lesson 7 – Access Modifiers

## What Is an Access Modifier

MS Docs says:

Access modifiers are keywords used to specify the declared accessibility of a member or a type. This section introduces the four access modifiers:

public

protected

internal

private

Make sure you read up on these, as they are important!

## Further Reading

<https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/access-modifiers>

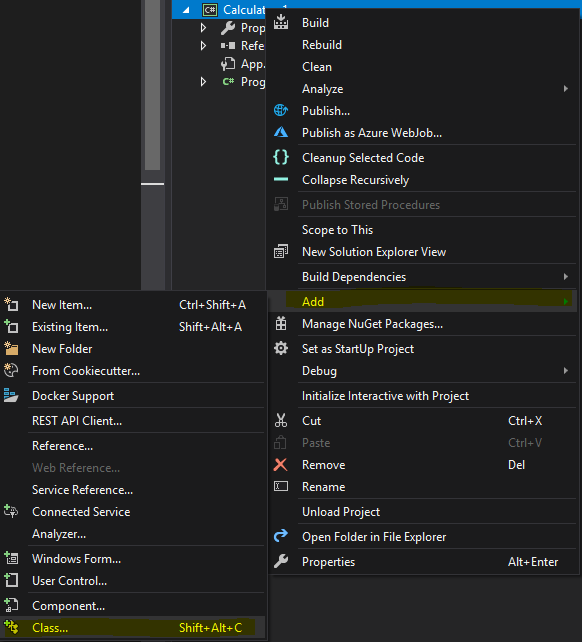
<https://www.c-sharpcorner.com/uploadfile/puranindia/what-are-access-modifiers-in-C-Sharp/>

<https://www.codeproject.com/questions/418798/access-modifiers-in-csharp>

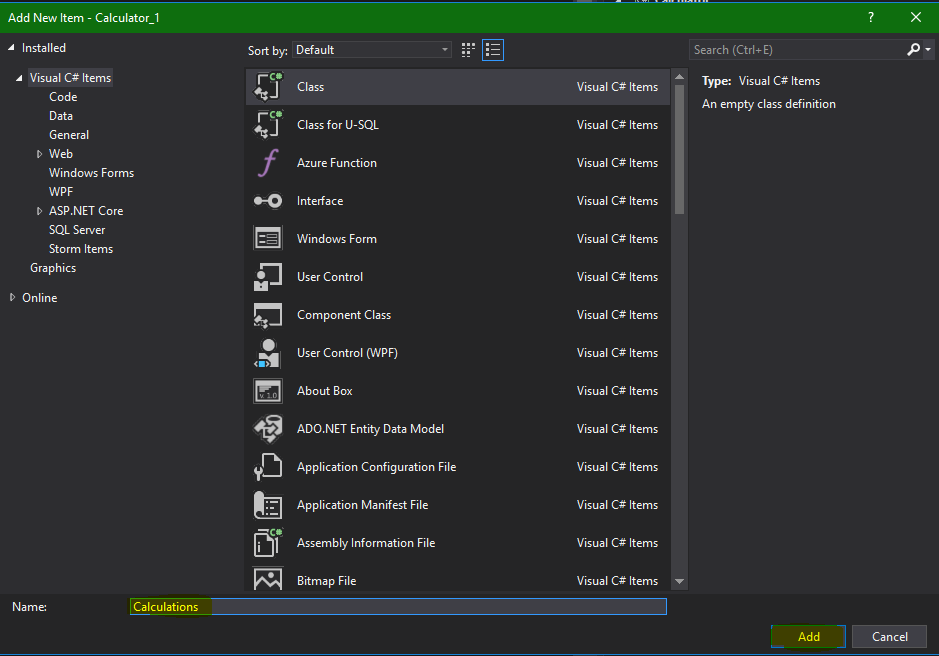
# Lesson 8 – Creating a New Class for our Calculator

## Creating the File

The first thing we need to do is add a class file to our project. To do this, we are going to right click on the Calculator\_1 project and select New > Class



Give the class a name of Calculations and press Add.



Make the class a public class, and add the Add method. I will give you the method signature for this, but the rest is up to you! Make sure you have methods for Add, Subtract, Multiply, and Divide, and try to add methods for Square Root and Power (or exponent).



## Further Reading

<https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/double>

<https://www.dotnetperls.com/double>

<https://msdn.microsoft.com/en-us/library/system.math(v=vs.110).aspx>

# Lesson 9 – If, Else, and Switch

Logic is very important in all programming languages, and the backbone of logic is the If…Else statement and Switch statements. Watch some video’s to familiarize yourself with the concepts!

## Further Reading

<https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/if-else>

<https://www.dotnetperls.com/if>

<https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/switch>

<https://www.dotnetperls.com/switch>

# Lesson 10 – Looping the user Input

Head back to the Program.cs file, and let’s create a loop that runs indefinitely for user input. Many people call this type of loop a “game loop”, because it runs continuously until the user quits the program.

## Infinite Loops

It may seem like infinite loops are a bad thing, but actually the can be quite useful! Think about your cell phone for instance. Your phone has an infinite loop that looks something like this:

Check for new text messages

Display the messages

Go to sleep for x seconds

This loop only ends when the power is turned off of your cell phone! Without the loop being infinite, it would have to be manually triggered by you or some other process!

## Our Menu Loop

Our menu loop is going to look something like this:

Ask for input  
Run the correct calculation  
Print the answer

This loop will continue until the user selects the option to exit, in which case the program will shut down. The proper way to make an infinite loop, according to the ISO standards, is

for(;;)

If you think back to lesson 2, you know that the for loop takes three parameters, running at the beginning of the loop, checking a condition, and at the end of each iteration through the loop. However, none of these parameters are actually *required* to be present! In this case, the loop does nothing at the beginning, nothing after each iteration, and never checks whether it should stop running!

## Hints

Some hints to get you started:

* Use the for(;;) loop
* Ask for input and parse it into an integer data type
* Use a switch statement to make your code cleaner when calling the correct method from the Calculations class
* Use Environment.Exit(0); to exit the program.

## Further Reading

<https://docs.microsoft.com/en-us/dotnet/csharp/tutorials/console-teleprompter>

<https://msdn.microsoft.com/en-us/library/system.environment(v=vs.110).aspx>

# Lesson 11 – Cleaning up our Calculations class

## Add Two Methods to Your Calculations Class

Let’s clean up our code by adding two methods to our Calculations class, with the signatures:

 and 

Can you figure out how to implement these two methods to get the numbers needed? Hint: remember our loops! Now implement them into your classes!