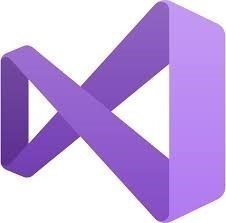
**CS 131C# - Beginner**

**HOP04 – Methods and Scope**

1/5/2020 Developed by Kim Nguyen

Center for Information Assurance (CIAE) @City University of Seattle (CityU)



**Before You Start**

* Version numbers may not match with the most current version at the time of writing. If given the option to choose between stable release (long-term support) or most recent, please choose the stable release rather than beta-testing version.
* This tutorial targets Windows users and MacOS users.
* There might be subtle discrepancies along the steps. Please use your best judgement while going through this cookbook style tutorial to complete each step.
* For your working directory, use your course number. This tutorial may use a different course number as an example.
* The directory path shown in screenshots may be different from yours.
* If you are not sure what to do or confused with any steps:
  1. Consult the resources listed below.
  2. If you cannot solve the problem after a few tries, ask a TA for help.

**Learning Outcomes**

Students will be able to:

* Understand, define and call methods
* Understand, define and call parameters and return types
* Understand and differentiate scopes of variables.

**Resources**

* C# Tutorials | W3Schools.com- <https://www.w3schools.com/cs/default.asp>
* C# Tutorials | tutorials.com- [https://www.tutorialspoint.com/csharp/](https://www.tutorialspoint.com/csharp/csharp_strings.htm)
* Scope of Variables in C# | geeksforgeeks.org - <https://www.geeksforgeeks.org/scope-of-variables-in-c-sharp/>
* Understanding Classes and Objects the C# Way| informit.com - <http://www.informit.com/articles/article.aspx?p=1609145&seqNum=4>

**Methods**

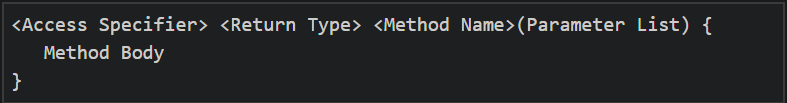
A method is a block of code or a group of statements that together perform a task when it is called. Every C# program has at least one class with a method named Main.

Methods are used to perform certain actions, and they are also known as functions.

You can pass data, known as parameters, into a method.

A method is defined with the name of the method, followed by parentheses (). C# provides some pre-defined methods, which you already are familiar with, such as Main(), but you can also create your own methods to perform certain actions.

When you define a method, you basically declare the elements of its structure. The syntax for defining a method in C# is as follows:



Following are the various elements of a method:

**Access Specifier:** This determines the visibility of a variable or a method from another class.

**Return type:** A method may return a value. The return type is the data type of the value the method returns. If the method is not returning any values, then the return type is void.

**Method name:** Method name is a unique identifier and it is case sensitive. It cannot be same as any other identifier declared in the class.

**Parameter list:** Enclosed between parentheses, the parameters are used to pass and receive data from a method. The parameter list refers to the type, order, and number of the parameters of a method. Parameters are optional; that is, a method may contain no parameters.

**Method body:** This contains the set of instructions needed to complete the required activity.

**Create a project**

1. Open Visual Studio.
2. File > New > Project
3. Select Console App (.NET Core), click Next
4. Type “Methods” in the Project name and save it in the following locations:

**If you are an online student:**

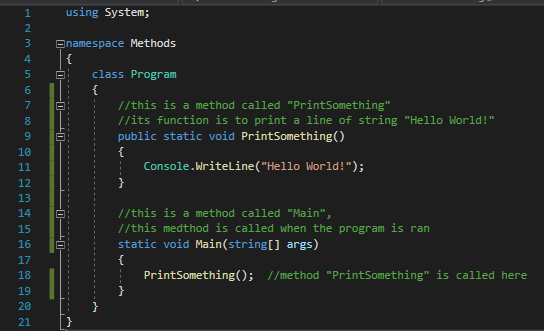
Save it here > CS131-Winter-2020\**ON**\FirstnameLastname/Module4/Methods-Scopes

**If you are an onsite student:**

Save it here > CS131-Winter-2020\**IN**\FirstnameLastname/Module4/Methods-Scopes

You should be in > FirstnameLastname/Module4/Methods-Scopes/Methods

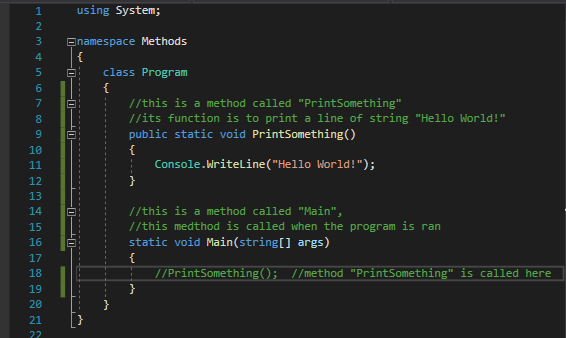
1) Modify your code to match the following screenshot:



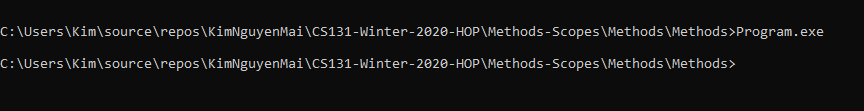
2) Run your code, you should see “Hello World” printed:



3) Now, let’s comment out the method call to see what happen (line 18):

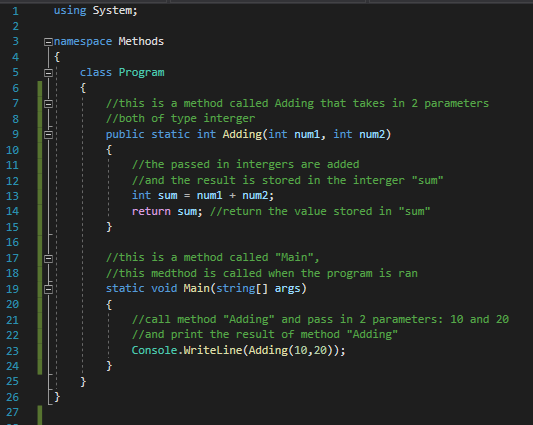


4) Run your program again, you should see a blank console:



This is because in step 3, we did not call the method “PrintSomething”, thus, even though the “PrintSomething” exist in the program, it was never called, thus, never ran to perform what it supposed to do, which is print “Hello World” on the screen.

5) Modify your code to match the following screenshot:



Notice that in line 9, the return type is set to “int”, this indicates that we expect the “Adding” method to produce a value that has type integer.

Return Type specifies the output type of a method. The output type can be a number,text etc..

If output of method is a number, the return type may be int or float or decimal etc..

If output of method type is text, the return type will be string etc..

Line 9, we indicate that method “Adding” needs 2 parameters to work. So when we call this method in line 23, we gave it 2 parameters as required (10 and 20).

Those 2 parameters (10 and 20), then passed to “num1” and “num2” respectively. They are added to each other arithmetically (10 + 20) and the result is stored in integer “sum” (30)

The return statement indicates that method “Adding” produces the value of “sum”

6) Now, let’s run your code to see the result:



**CHALLENGE:**

Write a method called “Mod” to find the modulus of a divided by b.

Mod takes 2 parameters a and b, both of type integer.

Mod returns the modulus of a/b

a has the value of 20, b has the value of 13

Expected result:



**Scope**

The part of the program where a particular variable is accessible is termed as the Scope of that variable. A variable can be defined in a class, method, loop etc. In C#, a scope is defined by the curly braces “{}”. If a variable that was declared inside a method, the variable has method level scope and is not accessible outside the method. Those variables are called “local variables”

Let’s see things in action.

**Create a project**

1. Open Visual Studio.
2. File > New > Project
3. Select Console App (.NET Core), click Next
4. Type “Scopes” in the Project name and save it in the following locations:

**If you are an online student:**

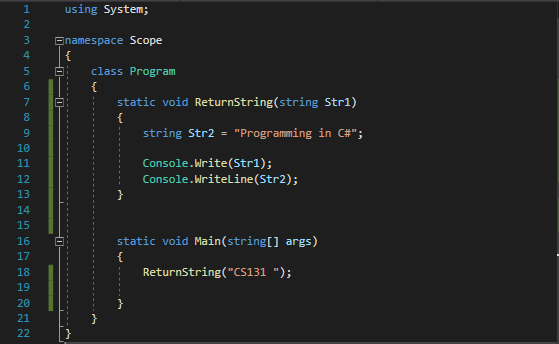
Save it here > CS131-Winter-2020\**ON**\FirstnameLastname/Module4/Methods-Scopes

**If you are an onsite student:**

Save it here > CS131-Winter-2020\**IN**\FirstnameLastname/Module4/Methods-Scopes

You should be in > FirstnameLastname/Module4/Methods-Scopes/Scopes

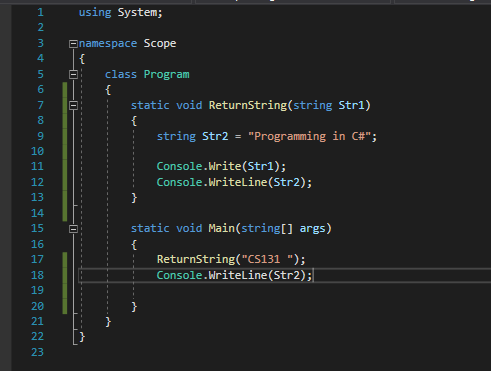
1) Add the following code in your Program.cs;



2 ) Run your program:



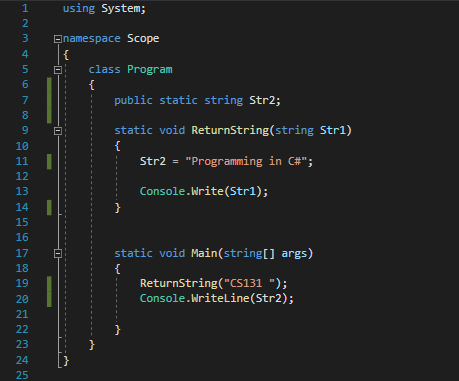
Line 12, we can call Str2 since it’s inside ReturnString() method scope. Now, let’s try to call it outside of ReturnString() scope. Add line 18 to your code as the following screenshot:



3) Run your program, you should receive the following error:



4) Let’s try something else. Add line 7 to your code and delete line 12 to match the following screenshot:



5) Run your code again:



This time we were able to call “Str2” from the Main() method because in line 7, we have made Str2 a public global variable, which means it was declared in the Class Scope, allowing any methods inside the Class Program scope to access it (Class Scope is higher than Method Scope).

No Challenge for this section 😊

**Push your work to GitHub**

**Commit changes**

1. Click on the **Home** button > **Changes**
2. Type commit message
3. Select **Commit All and Push**

**Create a pull request**

1. Go to your fork page on GitHub website
2. Near the top left side, change the active branch to your new branch
3. Click on the "New Pull Request" button next to the branch name.