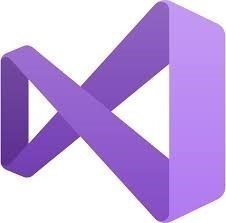
**CS 131C# - Beginner**

**HOP05 – Conditional Structures and Loops**

02/04/2020 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 and use conditional statements
* Understand, use and get familiar with more loops, particularly "while" loop.

**Resources**

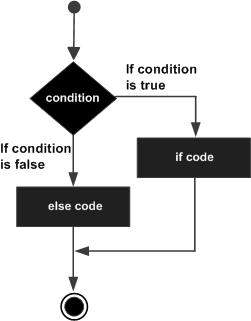
* C# Tutorial | Freecodecamp.org- <https://youtu.be/GhQdlIFylQ8>
* C# Tutorials | W3Schools.com- <https://www.w3schools.com/cs/default.asp>
* C# Tutorials | tutorials.com- [https://www.tutorialspoint.com/csharp/](https://www.tutorialspoint.com/csharp/csharp_strings.htm)

**Conditional Statements**

An if statement can be followed by an optional else statement, which executes when the boolean expression is false.

Use the if statement to specify a block of C# code to be executed if a condition is True.

Use the else statement to specify a block of code to be executed if the condition is False.



If the boolean expression evaluates to true, then the if block of code is executed, otherwise else block of code is executed.

An if statement can be followed by an optional else if...else statement, which is very useful to test various conditions using single if...else if statement.

When using if, else if, else statements there are few points to keep in mind.

An if can have zero or one else's and it must come after any else if's.

An if can have zero to many else if's and they must come before the else.

Once an else if succeeds, none of the remaining else if's or else's will be tested.

**Create a project**

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

**If you are an online student:**

Save it here > CS131-Spring-2020\**ON**\FirstnameLastname/Module5/Conditions-Loops

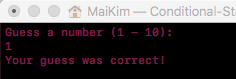
**If you are an onsite student:**

Save it here > CS131-Spring-2020\**IN**\FirstnameLastname/Module 5/Conditions-Loops

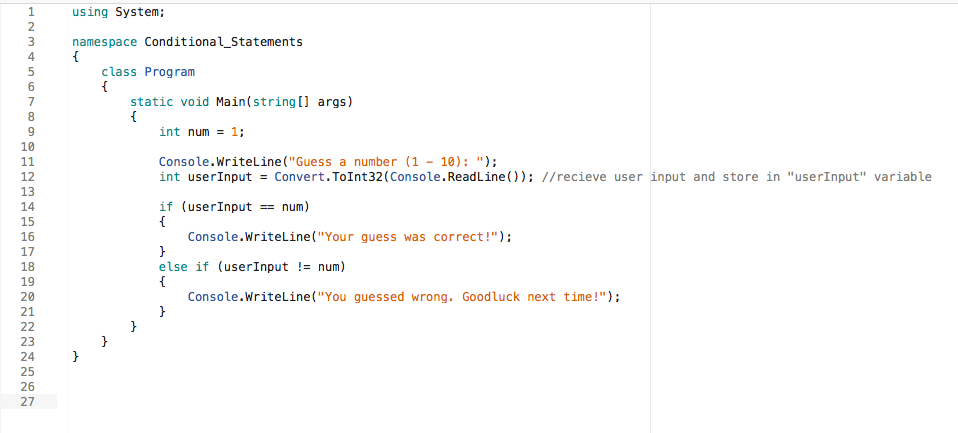
1) Type the following code into your Program.cs:



2) Run your program and type any number you want (from 1 – 10) into the console to check the result:

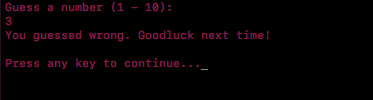


3) Now, let's update your code to match the following screenshot (line 18):



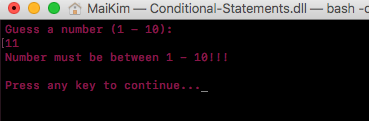
As explained above, it is possible to use one or multiple "else – if " statements following the initial if statement.

4) Run your program again, try to put any number that's not 1, we should expect this result:



**CHALLENGE:**

Write another condition that, if the user enter a number that's bigger than 10, a message shows on the screen: "Number must be between 1 – 10!!!". Expected:



*Hint: use another "else-if" statement.*

**Loops**

**For Loop:**

We have studied and practiced on for loop in the previous weeks. Please revisit the HOP if you do not remember.

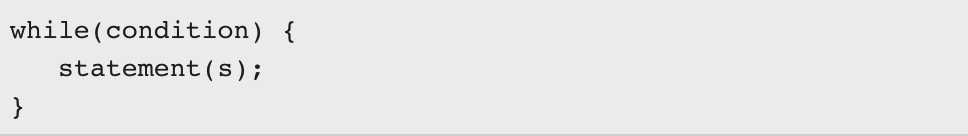
**While Loop**

A while loop statement in C# repeatedly executes a target statement as long as a given condition is true.

Loops can execute a block of code as long as a specified condition is reached.

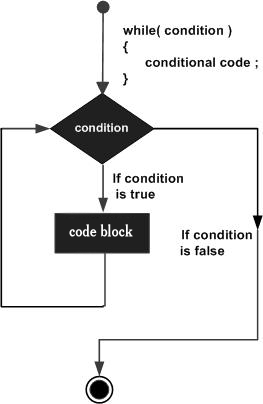
Loops are handy because they save time, reduce errors, and they make code more readable. The while loop loops through a block of code as long as a specified condition is True.

Syntax:



Here, statement(s) may be a single statement or a block of statements. The condition may be any expression, and true is any non-zero value. The loop iterates while the condition is true.

When the condition becomes false, program control passes to the line immediately following the loop



**Create a project**

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

**If you are an online student:**

Save it here > CS131-Spring-2020\**ON**\FirstnameLastname/Module5/Conditions-Loops

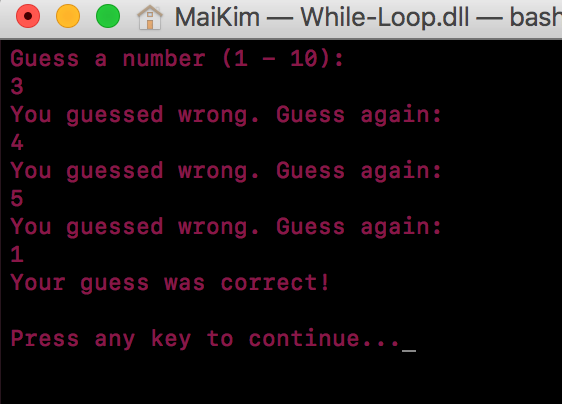
**If you are an onsite student:**

Save it here > CS131-Spring-2020\**IN**\FirstnameLastname/Module 5/Conditions-Loops

1) Write the following code into your Program.cs:

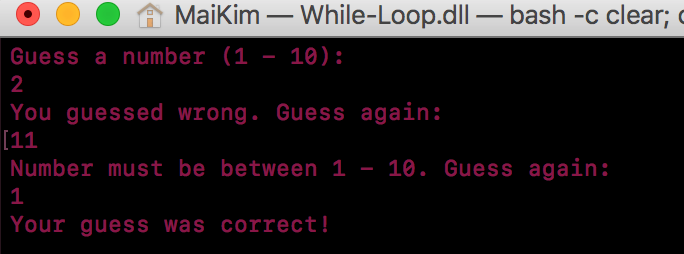


2) Run your program, try to guess numbers that is not 1, to see what happens:



**CHALLENGE:**

Write a loop or condition to cover the situation when user input is < 0 or user input is > 10, then a message "Number must be between 1 – 10, guess again: ", until the user guesses the correct number. Expected something like this:



Hint: You can update your code to match the following screenshot and just edit the existing while loop:



**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.