**Topic 3: Methods Passing by Value and by Reference**

Top of Form

Bottom of Form

**Content**

**Reading**

[Reading](https://dmacc.blackboard.com/webapps/blackboard/content/listContent.jsp?course_id=_102593_1&content_id=_7286949_1)

Pass by Value and By Reference

* + Microsoft:docs.microsoft.com
    - [Passing Value-Type Parameters](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/passing-value-type-parameters)
    - [Passing Reference-Type Parameters](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/passing-reference-type-parameters)
  + GeeksForGeeks: geeksforgeeks.com
    - [ref in C#](https://www.geeksforgeeks.org/ref-in-c-sharp/)
  + TutorialsPoint: tutorialspoint.com
    - [Passing Parameters by value](https://www.tutorialspoint.com/csharp/csharp_value_parameters.htm)
    - [Passing Paramters by reference](https://www.tutorialspoint.com/csharp/csharp_reference_parameters.htm)
  + W3Schools: w3schools.com
    - [Method Parameters](https://www.w3schools.com/cs/cs_method_parameters.asp)

**Passing by Value**

[Passing by Value](https://dmacc.blackboard.com/webapps/blackboard/content/listContent.jsp?course_id=_102593_1&content_id=_7286949_1)

You have seen passing parameters by value already, in fact it is the only way you have passed parameters to methods so far:

private static void DisplayMessage()

{

Console.WriteLine("Print a message");

}

The method takes one argument, as a value. It contains a method call to Console.WriteLine() with a string literal argument. Yes, Console is a class and WriteLine() is the first method you used!

You learned about scope as well, that when you passed a parameter by value, it was a copy of the value and was not affected by the action of the methods.

https://youtu.be/UIcOGa0Qwxg

using System;

using System.Data;

namespace Methods

{

class Program

{

private static void DisplayMessage(String name, int score)

{

score++;

Console.WriteLine("Nice job {0:G} on your score of {1:G} out of {2:G}", name, score, possible);

}

public static void Main(string[] args)

{

// Method call using literals

DisplayMessage("Mohammed", 90);

// Method call using variables

String name = "Ayah";

int score = 80;

DisplayMessage(name, score);

}

}

}

**Passing by Reference**

[Passing by Reference](https://dmacc.blackboard.com/webapps/blackboard/content/listContent.jsp?course_id=_102593_1&content_id=_7286949_1)

In the previous examples, passing by value, the method recieves a copy of the value and any variable in the method call statement renames unchanged. The local method may change the value, but it is only changed for the scope of the method call. Once the method terminates, execution returns to the statement immediately proceeding the method call, and the local method variables are not accessible.

If you want to change the value a variable in the method call statement scope, then you must pass by reference. This is easily accomplished by using the keyword ref preceding the parameter data type in the method argument list. Note the location in the example below.

private void SetToZero(ref int number)

{

number = 0;

}

Another way this might be useful is to overcome the limitation of methods that they have one return type. In an earlier exmaple (shown below) you see that two separate methods were need to gather input for two separate variables from the user. How could you use one method instead?

private static string GetUserName()

{

string name; // local variable

Console.WriteLine("Please enter your name");

name = Console.ReadLine();

return name; // return the value

}

private static int GetUserLevel()

{

int level = -1 ; // initialize due to try/catch

Console.WriteLine("Please enter your level");

try

{

level = Convert.ToInt32(Console.ReadLine());

}

catch (FormatException e)

{

Console.WriteLine("Error getting user level");

}

return level; // returns the value

}

public static void Main(string[] args)

{

string name;

int level;

// get user input

name = GetUserName();

level = GetUserLevel();

// print output

Console.WriteLine("User {0:G} is at level {1:G}", name, level);

}

How could you use one method instead?

https://youtu.be/n4LVa3tWE4c

private static void GetUserInfo(ref string name, ref int level)

{

Console.WriteLine("Please enter your name");

name = Console.ReadLine();

Console.WriteLine("Please enter your level");

try

{

level = Convert.ToInt32(Console.ReadLine());

}

catch (FormatException e)

{

Console.WriteLine("Error getting user level");

}

}

public static void Main(string[] args)

{

string name = null;

int level = -1;

// get user input

GetUserInfo(ref name, ref level);

// print output

Console.WriteLine("User {0:G} is at level {1:G}", name, level);

}

**Methods Pass by Reference Pass by Value Practice**

[Methods Pass by Reference Pass by Value Practice](https://dmacc.blackboard.com/webapps/blackboard/content/listContent.jsp?course_id=_102593_1&content_id=_7286949_1)

Write a Console App update the four people's ages.

Write a method to print four ages.

* + Use Pascal Case to write a descriptive name for a method to print four ages
  + Method accepts 4 ages as parameters, you determine the data type and if pass by value or by reference
  + Determine the return type for the method
  + Write a print statement to print the 4 ages (with enough information for the user to understand the output)
  + Add a comment in the method stating if method is pass by reference or pass by value and why
  + Add a comment in the method body to explain the return type you selected

Write ONE method only to update the four people's ages.

* + Use Pascal Case to write a descriptive name for a method to update ages
  + Method accepts 4 ages as parameters, you determine the data type and if pass by value or by reference
  + Determine the return type for the method
  + Use the increment operator in the method
  + Add a comment in the method stating if method is pass by reference or pass by value and why
  + Add a comment in the method body to explain the return type you selected

In the Main() method

* + Declare 4 variable representing the ages of 4 people, selecting appropriate data types, names and following naming conventions
  + Make a method call to print the ages
  + Make a method call to update the ages
  + Make a method call to print the ages