

Assignment—1

Aim: - Installation and Basic Tutorial

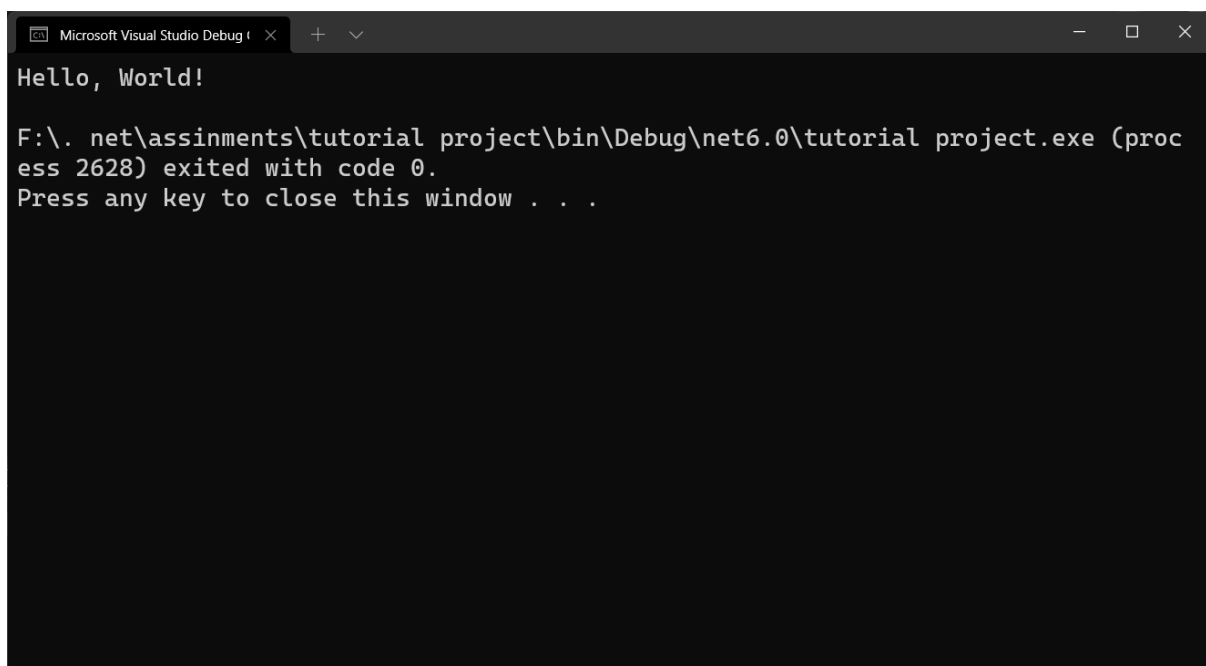
Step 1: Intro

This is a .NET console application written in C#. Select **Run Code** to try it out.

Code:

```
Console.WriteLine("Hello, World!");
```

Output:

A screenshot of a Microsoft Visual Studio Debug Console window. The window has a title bar with the text "Microsoft Visual Studio Debug Console" and standard window controls. The console output shows "Hello, World!" on the first line. Below it, there is a message indicating the process has exited: "F:\.net\assinments\tutorial project\bin\Debug\net6.0\tutorial project.exe (process 2628) exited with code 0." followed by "Press any key to close this window . . .".

```
Microsoft Visual Studio Debug Console
Hello, World!
F:\.net\assinments\tutorial project\bin\Debug\net6.0\tutorial project.exe (process 2628) exited with code 0.
Press any key to close this window . . .
```

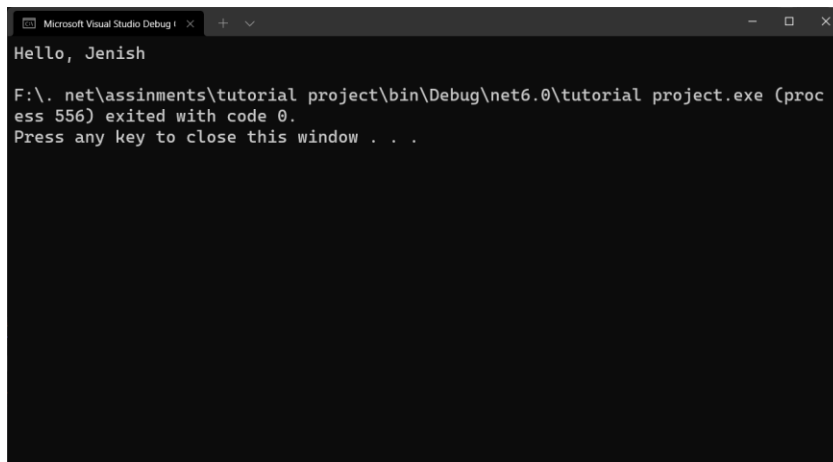
Step 2: Strings

Try modifying the code so that the console says hello to your name, instead of the world (for example, **Hello Ana!**). Select **Run Code** to see if you got it right.

Code:

```
Console.WriteLine("Hello, World!");
```

Output



```
Microsoft Visual Studio Debug Console
Hello, Jenish

F:\. net\assinments\tutorial project\bin\Debug\net6.0\tutorial project.exe (process 556) exited with code 0.
Press any key to close this window . . .
```

Step 3: Variables

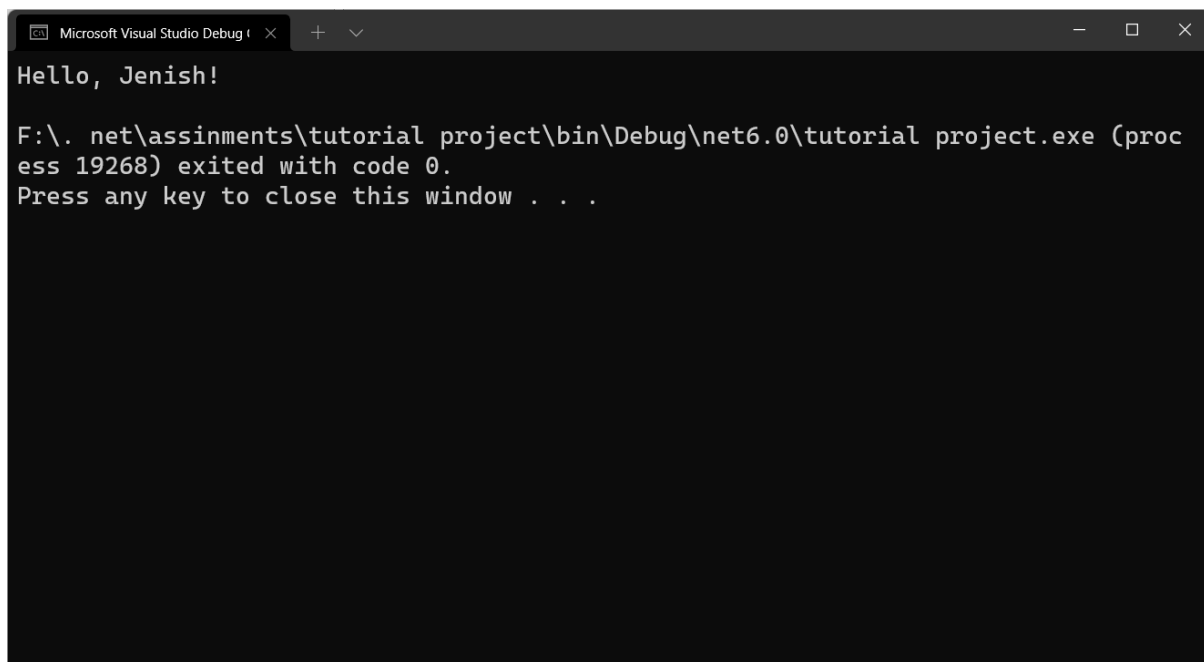
Variables hold values that you can use elsewhere in your code.

Let's store your name in a `name` variable, then read the value from that variable when creating the output message.

Code:

```
var name = "Jenish";
Console.WriteLine("Hello, " + name + "!");
```

Output:



```
Microsoft Visual Studio Debug Console
Hello, Jenish!

F:\. net\assinments\tutorial project\bin\Debug\net6.0\tutorial project.exe (process 19268) exited with code 0.
Press any key to close this window . . .
```

Step 4: String interpolation

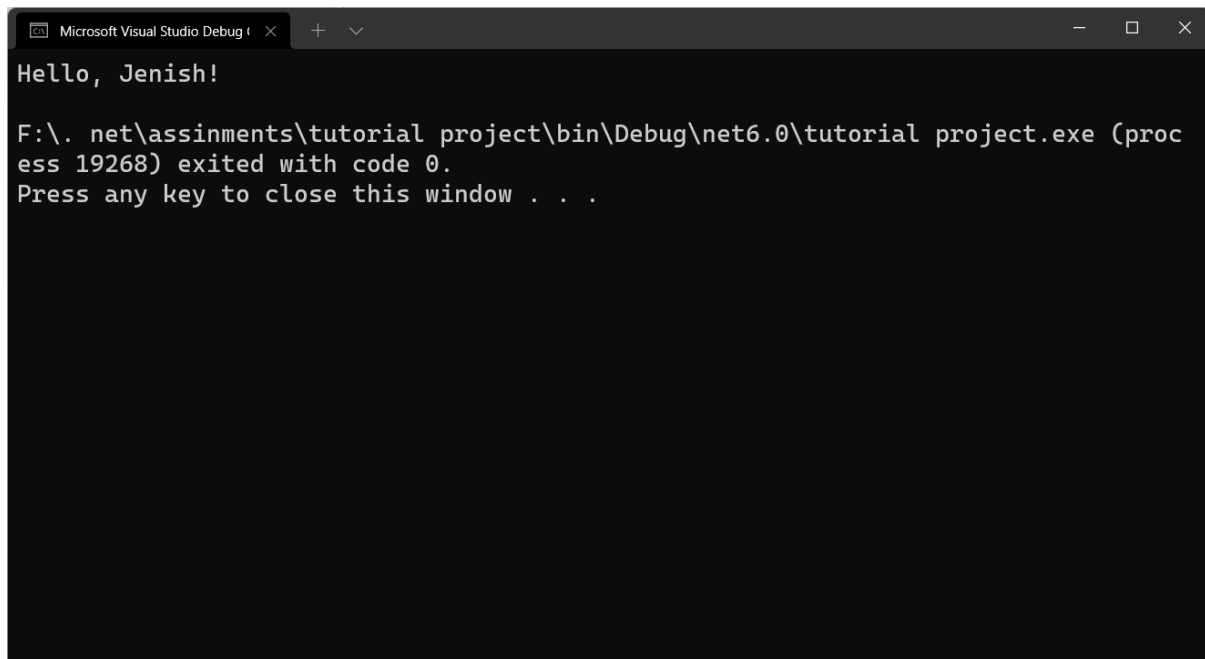
String interpolation lets you piece together strings in a more concise and readable way.

If you add a `$` before the opening quotes of the string, you can then include string values, like the `name` variable, inside the string in curly brackets. Try it out and select **Run Code**

Code:

```
var name = "jenish";  
Console.WriteLine($"Hello {name}!");
```

Output:

A screenshot of a Visual Studio Debug Console window. The window has a title bar that says "Microsoft Visual Studio Debug Console". The output text inside the console is: "Hello, Jenish!" followed by "F:\. net\assinments\tutorial project\bin\Debug\net6.0\tutorial project.exe (process 19268) exited with code 0." and "Press any key to close this window . . .".

```
Microsoft Visual Studio Debug Console  
Hello, Jenish!  
F:\. net\assinments\tutorial project\bin\Debug\net6.0\tutorial project.exe (process 19268) exited with code 0.  
Press any key to close this window . . .
```

Step 5: Methods

Methods take inputs, do some work, and sometimes return a result.

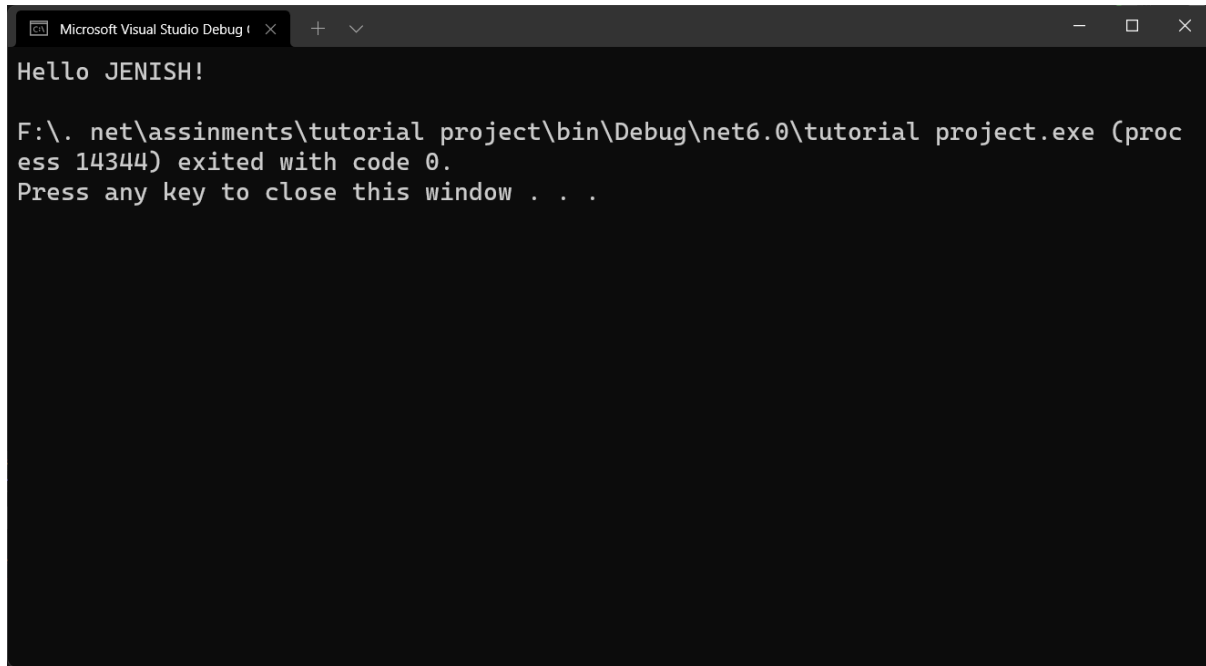
`ToUpper()` is a method you can invoke on a string, like the `name` variable. It will return the same string, converted to uppercase.

Update the greeting to change the name of the person being greeted to uppercase and select **Run Code**.

Code:

```
var name = "jenish";  
Console.WriteLine($"Hello {name.ToUpper()}!");
```

Output:



Step 6: Collections

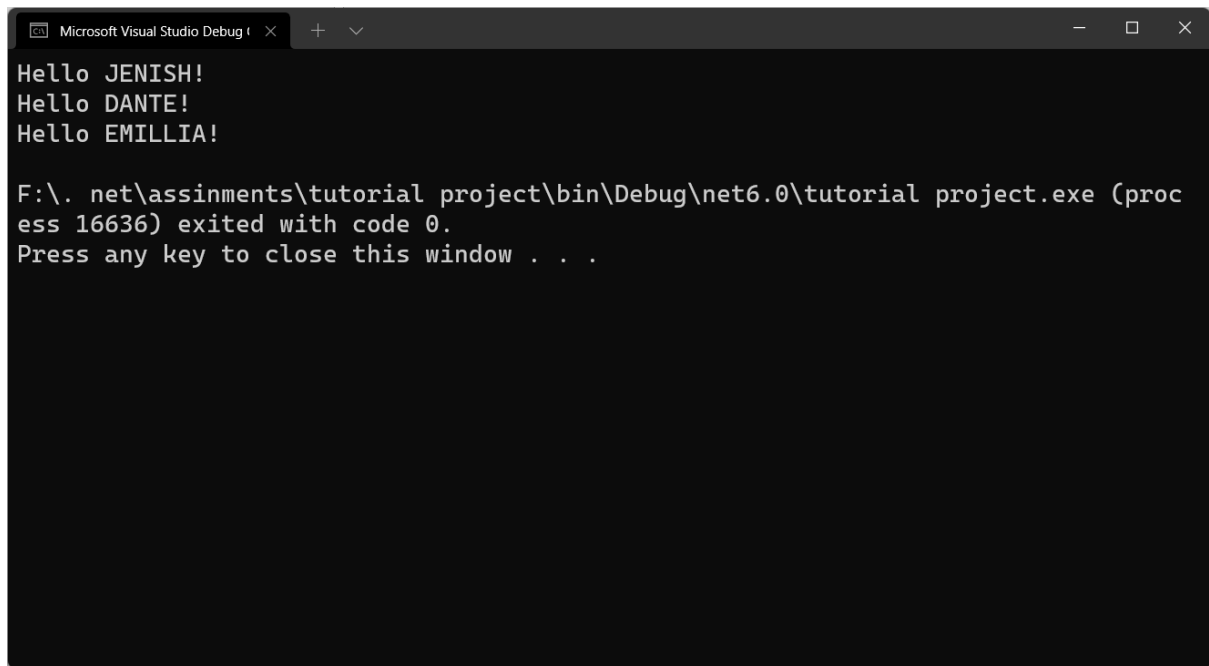
Collections hold multiple values of the same type.

Replace the `name` variable with a `names` variable that has a list of names. Then use a `foreach` loop to iterate over all the names and say hello to each person.

Code:

```
var names = new[] { "jenish", "Dante", "Emillia" };  
foreach (var name in names)  
{  
    Console.WriteLine($"Hello {name.ToUpper()}!");  
}
```

Output:



```
Microsoft Visual Studio Debug Console
Hello JENISH!
Hello DANTE!
Hello EMILLIA!

F:\. net\assinments\tutorial project\bin\Debug\net6.0\tutorial project.exe (process 16636) exited with code 0.
Press any key to close this window . . .
```