Strings can be Null or Empty or Unassigned

Like other reference types, `string` references can be *null* or *unassigned*. They can also have a third value: *empty*.

```
// Unassigned
string s;
// Null
string s2 = null;
// Empty string
string s3 = "";
// Also empty string
string s4 = String.Empty;
// This prints true
Console.WriteLine(s3 == s4);
```

All of these signify a lack of text, but they each mean something slightly different:

unassigned means that the programmer did not give the variable any value

null means that the programmer intentionally made the variable refer to no object

an empty string signifies a piece of text with zero characters. This is often used to represent a blank text field. It can be represented by `""` or `String.Empty`

The Microsoft Programming Guide suggests using `String.Empty` or `""` instead of `null` to avoid `NullReferenceException` errors.

We can check for null OR empty strings using the static `String` method `IsNullOrEmpty()`. It's explained in more detail in the documentation.

☑**Instructions**

1.
Using `Console.Write()` and `Console.ReadLine()`, ask the user for input and capture it in a variable.

Hint

Print out a message asking for input. Use `Console.Write()`.

On the next line, call `Console.ReadLine()` and store the returned value in a `string` variable.

2.
Write an `if`-`else` statement that checks for a null or empty string. If it is null, print out the message:

```
"You didn't enter anything!"
```

Otherwise, print out the message:

```
"Thank you for your submission!"
```

Hint

Here's an example `if`-`else` statement:

```
string color = "red";

if (color == "blue")
{
  Console.WriteLine("color is blue");
}
else
{
  Console.WriteLine("color is NOT blue");
}
```

3.
Run the program using `dotnet run`.

What happens when you enter no text and hit Enter?