Ternary Operators

The *ternary operator* allows for a compact syntax in the case of binary decisions. Like an if...else statement, it evaluates a single condition and executes one expression if the condition is true and the second expression otherwise.

Here's an example of a ternary operator in action:

```
string color = "blue";
string result = (color == "blue") ? "blue" : "NOT blue";
Console.WriteLine(result);
```

In this example, we create a variable result and save the outcome of the ternary operator expression. The expression starts with the Boolean expressions (color == "blue"), followed by the ternary operator ?, then the two possible outcomes, separated by a colon : . The first outcome, "blue" will be saved to result if the Boolean expression evaluates to true, otherwise it will store the second outcome.

Ternary operators can also be chained, like else if statements. But careful! Since the entire expression exists on one line, it can quickly become unreadable.

When using ternary operators, make sure to pay attention to:

Parentheses: we place the boolean expression that the statement is evaluating in parentheses ().

The ? operator: make sure this comes after the statement and before the outcomes.

Colon: This separates the two possible outcomes.

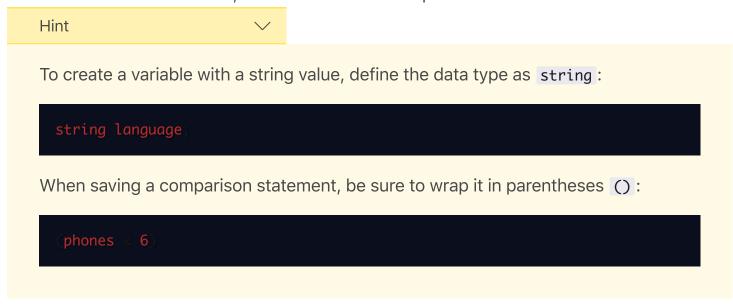
Instructions

1.

You're growing peppers and wrote a program that lets you know if it's time to pick them. If a pepper is at least 3.5 inches, it's time to be picked. If it's not ready, the program should tell you to "wait a little longer".

Start by creating a string variable named message. Save the comparison statement that checks to see if the pepperLength is 3.5 inches or more.

Note: This will throw an error, since we have not completed our statement.



2. Next, write out your ternary operation. If a pepper is ready to be picked (3.5 inches or longer) your program should set message to "ready!" If it's not ready it should set message to "wait a little longer".



To use a ternary operator, first write your condition, followed by the ternary operator? Follow with an output if the condition is true, followed by a colon:, and then the output if the condition is false.

```
string color = "blue";
string result = (color == "blue") ? "blue" : "NOT blue";
Console.WriteLine(result);
```

