CSP1150/CSP5110: Programming Principles

Reading 2.3: Shorthand If-Then-Else

Many languages offer a shorthand version of an if-then-else statement which can be written on a single line. As with all shorthand it allows programmers to write more concise code, but can make code less readable – particularly to someone who is not aware of the shorthand version.

Here is a traditional if-then-else statement as it would appear in most languages:

Most Languages

```
if (score >= 50)
{
    result = 'Pass';
}
else
{
    result = 'Fail';
}
```

Here it is again in Python:

Python

```
if score >= 50:
    result = 'Pass'
else:
    result = 'Fail'
```

The code will store "Pass" into the result variable if the score variable is greater than or equal to 50, otherwise it will store "Fail" – simple enough!

Here is how that could be rewritten using the shorthand if-then-else in most languages:

Most Languages

```
result = (score >= 50) ? 'Pass' : 'Fail';
```

The structure of the shorthand if-then-else is:

```
<boolean expression> ? <true value> : <false value>
```

Python's version strives to be more readable:

Python

```
result = 'Pass' if score >= 50 else 'Fail'
```

The structure of the Python version is:

```
<true value> if <boolean expression> else <false value>
```

Apart from being concise, the shorthand version is different because it is an *operator*, rather than a statement. We've covered a number of operators already - arithmetic (+, -, *, /), relational (==, >=, !=, etc) and logical (and, or, not) ones. Operators operate on their *operands* (the values that you put around them), e.g. 5 and 2 are the operands in "5 + 2".

An operator *returns a result when it is evaluated* – e.g. the sum of the numbers in an addition, True/False for a relational or logical operator, and so on.

A shorthand if-then-else is exactly the same, except it involves *three* operands – the boolean expression, the true value and the false value. When it is evaluated, it returns the true value if the boolean expression evaluates to true, otherwise it returns the false value.

The shorthand if-then-else can be used within an expression, like any other operator. Expressions are evaluated when a statement is run, e.g. manipulating a value with arithmetic operators before assigning the result to a variable, or determining the True/False result of a boolean expression when running an if-then statement.

Being able to evaluate a condition as part of an expression can lead to some very concise and useful code, such as the following example:

```php
echo $qty. ' item'.($qty != 1 ? 's' : ''). ' in cart';
```

Shows "item" or "items" as appropriate based on the value in the $qty variable.
If $qty is not equal to 1, show "# items in cart", otherwise show "1 item in cart".

Try writing a Python version of this example if you can!

For more information (and examples in many different languages), see the ?: Wikipedia page.