

# Part 03 - Flow Control - Conditionals

## Part 03 - Flow Control - Conditionals

### If Statement

#### Definition: if statement

A control statement that contains one or more Boolean expressions whose results determine whether to execute other statements within the If statement.

An `if` statement allows you to travel down multiple logical paths, depending on a condition given. If the condition given is `true`, the block of code associated with it will be run.

#### if statement

```
i = 5
if i == 5:
    print "i is equal to 5."
```

#### Output

```
i is equal to 5.
```

#### Be careful

notice the difference between `i = 5` and `i == 5`.

`i = 5` is an *assignment*

`i == 5` is a *comparison*

If you try an assignment while running a conditional, Boo will emit a warning.

You may have noticed that unlike other languages, there is no then-endif or do-end or braces `{ }`. Blocks of code are determined in Boo by its indentation. By this, your code blocks will always be noticeable and readable.

#### Recommendation

Always use tabs for indentation.

In your editor, set the tab-size to view as 4 spaces.

You can have multiple code blocks within each other as well.

## Multiple if statements

```
i = 5
if i > 0:
    print "i is greater than 0."
    if i < 10:
        print "i is less than 10."
        if i > 5:
            print "i is greater than 5."
```

## Output

```
i is greater than 0.
i is less than 10.
```

## If-Else Statement

With the `if` statement comes the `else` statement. It is called when your `if` statement's condition is false.

## if-else statement

```
i = 5
if i > 5:
    print "i is greater than 5."
else:
    print "i is less than or equal to 5."
```

## Output

```
i is less than or equal to 5.
```

Quite simple.

## If-Elif-Else Statement

Now if you want to check for a condition after your `if` is false, that is easy as well. This is done through the `elif` statement.

### if-elif-else statement

```
i = 5
if i > 5:
    print "i is greater than 5."
elif i == 5:
    print "i is equal to 5."
elif i < 5:
    print "i is less than 5."
```

## Output

```
i is equal to 5.
```

You can have one `if`, any number of `elif` s, and an optional `else` .

## Unless Statement

The `unless` statement is handy if you want a readable way of checking if a condition is not true.

## unless statement

```
i = 5
unless i == 5:
  print "i is not equal to 5."
```

## Output

It didn't output because i was equal to 5 in that case.

## Statement with Modifier

Like in Ruby and Perl, you can follow a statement with a modifier.

## Statement with modifier

```
i = 0
print i
i = 5 if true
print i
i = 10 unless true
print i
```

## Output

```
0
5
5
```

### ✔ Recommendation

Don't use Statement with Modifier on a long line. In that case, you should just create a code block.

A good rule of thumb is to not use it if the statement is more than 3 words long.

This will keep your code readable and beautiful.

Some common conditionals:

| Operator | Meaning                  | Example           |
|----------|--------------------------|-------------------|
| ==       | equal                    | 5 == 5            |
| !=       | not equal                | 0 != 5            |
| >        | greater than             | 4 > 2             |
| <        | less than                | 2 < 4             |
| >=       | greater than or equal to | 7 >= 7 and 7 >= 4 |
| <=       | less than or equal to    | 4 <= 8 and 6 <= 6 |

## Not Condition

To check if a condition is not true, you would use `not`.

### not condition

```
i = 0
if not i > 5:
    print 'i is not greater than 5'
```

### Output

```
i is not greater than 5
```

## Combining Conditions

To check more than one condition, you would use `and` or `or`. Use parentheses ( ) to change the order of operations.

## combining conditions

```
i = 5
if i > 0 and i < 10:
    print "i is between 0 and 10."
if i < 3 or i > 7:
    print "i is not between 3 and 7."
if (i > 0 and i < 3) or (i > 7 and i < 10):
    print "i is either between 0 and 3 or
between 7 and 10."
```

Note that `and` requires that both comparisons are true, while `or` requires that only one is true or both are true.

## Output

```
i is between 0 and 10.
```

## Exercises

1. Given the numbers `x = 4`, `y = 8`, and `z = 6`, compare them and print the middle one.

Go on to [Part 04 - Flow Control - Loops](#)