

# Reading and Research - Selection Statements

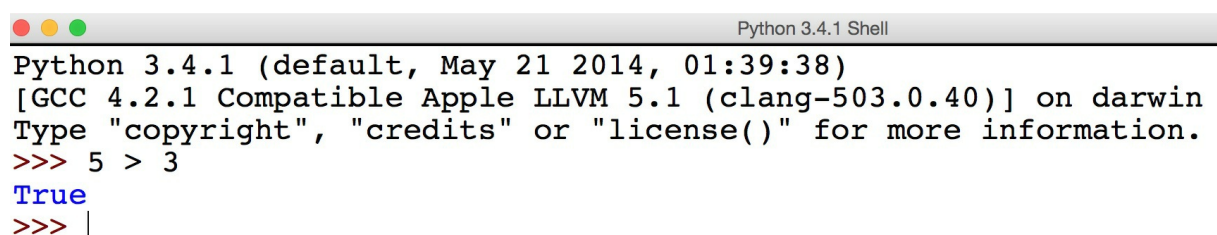
These tasks are designed to introduce you to the programming topic we will be studying in class next lesson. You **must** complete these activities prior to the lesson.

## Boolean Expressions

One of the most common tasks in computer programming is to **evaluate an expression**. An expression allows us to test whether a value (or set of values) meets particular criteria. The Python shell can evaluate expressions, we will use this to investigate expressions further.

### Task 1

Use the Python shell to investigate the expressions given below, describe what each symbol represents and indicate whether the expression evaluates to `True` or `False`.



```
Python 3.4.1 Shell
Python 3.4.1 (default, May 21 2014, 01:39:38)
[GCC 4.2.1 Compatible Apple LLVM 5.1 (clang-503.0.40)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> 5 > 3
True
>>> |
```

Expression	Symbol description	Result
<code>2 == 4</code>	<i>equals</i>	<code>False</code>
<code>5 &gt; 3</code>	<i>greater than</i>	<code>True</code>
<code>4 &gt;= 4</code>	<i>if number on left is greater or equal to number on right</i>	<code>True</code>
<code>3 &lt; 2</code>	<i>less than</i>	<code>False</code>

Expression	Symbol description	Result
<code>7 &lt;= 7</code>	<i>if number on left is less than or equal to number on right</i>	
<code>8 != 9</code>	<i>Checks if the value of two numbers are equal or not, if values are not equal then answer is true</i>	true

The symbols in **Task 1** are called **relational operators** and when an expression containing a relational operator is evaluated it returns a **boolean value** (True or False) as an answer.

In addition to evaluating expressions containing numbers we can also use **variables** in expressions. For example, imagine we had the following variable:

```
test_score = 56
```

We could use boolean expressions to evaluate whether testScore meets certain criteria (for example whether it is greater than the pass mark of 50). Let's test this out:

## Task 2

Enter `testScore = 56` into the Python shell and then investigate the expressions below.

```
Python 3.4.1 Shell
Python 3.4.1 (default, May 21 2014, 01:39:38)
[GCC 4.2.1 Compatible Apple LLVM 5.1 (clang-503.0.40)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> test_score = 56
>>> test_score == 50
False
>>> |
```

Expression	Symbol description	Result
<code>test_score == 50</code>	<i>equals</i>	False
<code>test_score &gt; 40</code>	<i>greater than</i>	true
<code>test_score &gt;= 60</code>	<i>if number on left is greater or equal to number on right</i>	false

Expression	Symbol description	Result
test_score < 40	less than	true
50 <= test_score	if number on left is less than or equal to number on right	true
56 != test_score	Checks if the value of two numbers are equal or not, if values are not equal then answer is true	true

## More complex boolean expressions

Sometimes it is not enough to evaluate an expression on a single criteria. We can create more complicated boolean expressions using boolean operators. There are three boolean operators that we must consider in programming:

### Operator

and

or

not

The `and` and `or` operators can be used to join expressions together into more complex expressions. The `not` operator is used to invert an expressions evaluation. For example if an expression evaluated to `True` using the `not` operator would make the result equal `False`.

## Task 3

Let's look at some straightforward examples. Use the Python shell to evaluate the following expressions:

Expression	Result
True and True	True

Expression	Result
True and False	False
False and True	False
False and False	False
True or True	True
True or False	True
False or True	True
False or False	False
not(True)	False
not(False)	True

Having completed the above table, use the space below to describe when `and` and `or` evaluate to `True` :

Operator	When it evaluates to <code>True</code>
<code>and</code>	<i>When its True and True</i>
<code>or</code>	<i>True and True, True and False, False and True</i>

## Selection statements

Before we find out more about selection statements let look at an example:

```
test_score = 56
if test_score >= 50:
    print("Pass")
if test_score < 50:
    print("Fail")
```

## Task 4

Without entering the code into Python, attempt to explain what the code does, using the space below for your answer:

**answer**

---

**the test score is 56, and if test score is greater or equal to 50 then display pass, if test score is less than 50 display false**

Now that we have looked at an example it is time to investigate selection statements in more detail. We will use the [Python School website](#) to do this.

## Task 5

Read the following two pages on Python Summer School and attempt the exercises mentioned.

1. [The IF Statement in Python](#)
  - The exercise at the bottom of the page
2. [More on IF Statements in Python](#)
  - The **first** exercise at the bottom of the page

## Task 6

In the space below **paste** the code from each of the exercises in Task 5 and include a screenshot of you running each program successfully.

```
#task 5.1

age = int(input("please enter your age:"))
if age >= 18:
    print("You are old enough to vote")
else:
    print("Sorry you are not old enough to vote")

retire = 65-age
print("you can also consider to retire in {0}years
time.".format(retire))
```

```

Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 00:54:21)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> WARNING: The version of Tcl/Tk (8.5.9) in use may be unstable.
Visit http://www.python.org/download/mac/tcltk/ for current information.
===== RESTART =====
>>>
please enter your age:17
Sorry you are not old enough to vote
you can also consider to retire in 48years time.
>>> ===== RESTART =====
>>>
please enter your age:21
You are old enough to vote
you can also consider to retire in 44years time.
>>>

```

```

#task 5.2
number = int(input("Guess a number between 1 and 20:"))
if number >=9:
    print("Your number is too high")
elif number <=7:
    print("Your number is too low")
elif number ==8:
    print("CONGRATULATIONS!! You have guessed the correct
number")

```

```

Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 00:54:21)
[GCC 4.2.1 (Apple Inc. build 5666) (dot 3)] on darwin
Type "copyright", "credits" or "license()" for more information.
>>> ===== RESTART =====
>>>
Guess a number between 1 and 20:7
Your number is too low
>>> ===== RESTART =====
>>>
Guess a number between 1 and 20:10
Your number is too high
>>> ===== RESTART =====
>>>
Guess a number between 1 and 20:8
CONGRATULATIONS!! You have guessed the correct number
>>> |

```

## Summary

In this R&R you have investigated selection statements. You have seen how expressions are constructed from relational operators and boolean operators. You have have seen the structure and syntax of a basic selection statement and had the opportunity to create programs that use this statement.

Please make sure you have completed this R&R fully before your next programming lesson as it will form the basis of the initial classroom discussion and starter tasks.