Reading and Research - Selection Statements

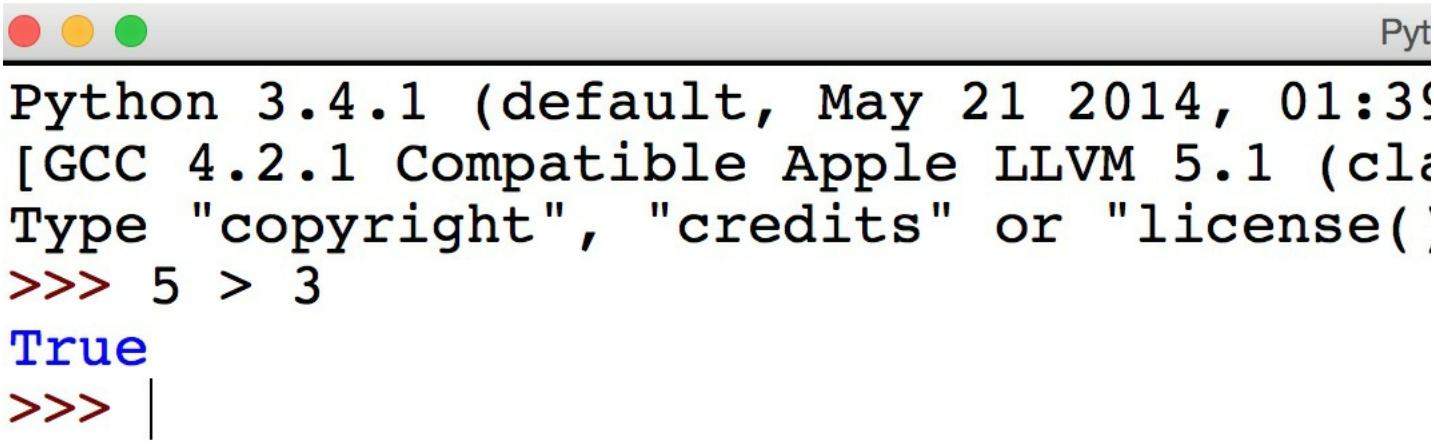
These tasks are designed to introduce you to the programming topic we will be studying in class next lesson. You **must** complete these activities prior to the lesson.

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.



Expression	Symbol description	Result
2 == 4	*Equals*	False
5 > 3	*Greater than*	True
4 >= 4	*Greater than or equal to*	True
3 < 2	*Less than*	False
7 <= 7	*Less than or equal*	True
8 != 9	*Not equal to*	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.

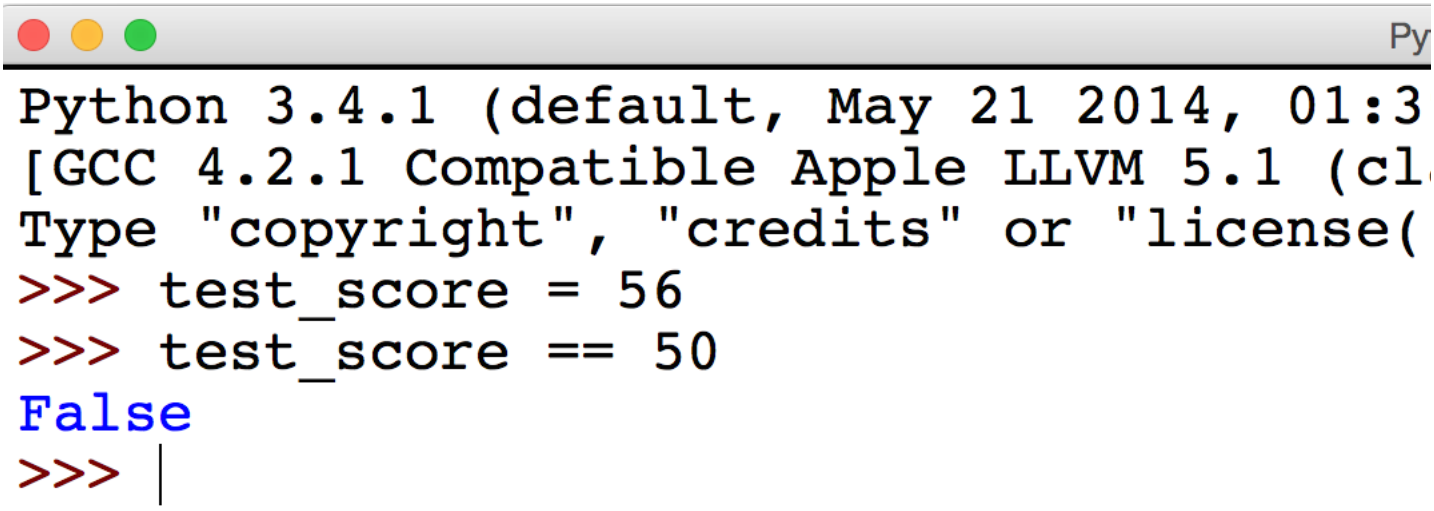
In addition to evaluating expressions containing numbers we can also use **variables** in expressions. For example, imagine we had the following variable:

```
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). Letâ€™s test this out:

Task 2

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



Expression	Symbol description	Result
test_score == 50	*equals*	False
test_score > 40	*Score greater than 40*	True
test_score >= 60	*score greater than or equal to 60*	False
test_score < 40	*score less than 40*	False
50 <= test_score	*50 is less than or equal to score*	True
56 != test_score	*56 is no equal to score*	False

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

Letâ€™s look at some straightforward examples. Use the Python shell to evaluate the following expressions:

Expression	Result
True and True	True
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 True
and	*becomes true only when both expressions are true*
or	*becomes true if more than one of the expressions are correct*

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

python takes the score "56", and if the score is greater than or equal to 50 it will then print pass. if the score is less than 50 it will print fail.

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 can vote")
else:
    print("you cannot vote yet")

retire = 65
if age > retire:
    print ("you can retire.")
else:
    print ("you can retire in {0} years.".format(retire-age))
```

```
#task 5.2
number = int(input("please enter a number between one and 20: "))

if number > 20:
    print("number is too high.")
elif number <0:
    print ("number is too low.")
else:
    print("number is within range")
```

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.