# Conditions and Logic

Computer programs make decisions based on logic: if some condition applies, do something, otherwise, do something else.

## Model 1   Comparison Operators

In Python, a comparison (e.g., `100 < 200`) will yield a ***Boolean*** value of either `True` or `False`. Most data types (including `int`, `float`, and `str`) can be compared using the following operators:

| Operator | Meaning |
|:---:|:---:|
| < | less than |
| <= | less than or equal |
| > | greater than |
| >= | greater than or equal |
| == | equal |
| != | not equal |

Consider the following lines that were entered into a Python Shell.

| Python code | Shell output |
|---|---|
| `type(True)` | <class 'bool'> |
| `type(true)` | NameError |
| `type(3 < 4)` | <class 'bool'> |
| `print(3 < 4)` | True |
| `three = 3` | |
| `four = 4` | |
| `print(three == four)` | False |
| `check = three > four` | |
| `print(check)` | False |
| `type(check)` | <class 'bool'> |
| `print(three = four)` | TypeError |
| `three = four` | |
| `print(three == four)` | True |

# Questions  (10 min) <span style="float:right">**Start time:**</span>

**1**. Write an asterisk (*) next to any row your group has questions about.

**2**. What is the name of the data type for Boolean values?

**3**. Do the words `True` and `False` need to be capitalized? Explain how you know.

**4**. For each of the following terms, identify examples from the table in Model 1:

   a) Boolean variables:

   b) Boolean operators:

   c) Boolean expressions:

**5**. Explain why the same expression `three == four` had two different results.

**6**. What is the difference between the = operator and the == operator?

**7**. Write a Boolean expression that uses the `!=` operator and evaluates to `False`.

# Model 2   Boolean Operations

Expressions may include Boolean operators to implement logic. If all three operators appear in the same expression, Python will evaluate not first, then and, and finally or. If there are multiple of the same operator, they are evaluated from left to right.

| Python code | Predicted output | Actual output |
|---|---|---|
| `print(a < b and b < c)` | | |
| `print(a < b or b < c)` | | |
| `print(a < b and b > c)` | | |
| `print(a < b or b > c)` | | |
| `print(not a < b)` | | |
| `print(a > b or not a > c and b > c)` | | |

## Questions  (20 min)                                    Start time:

**8**.  What data type is the result of `a < b`? What data type is the result of `a < b and b < c`?

**9**.  Predict the output of each print statement, based on the variables `a = 3`, `b = 4`, and `c = 5`. Then execute each line in a Python Shell to check your work.

**10**.  Based on the variables in #9, what is the value of a < b? What is the value of b < c?

**11**.  If two `True` Boolean expressions are combined using the `and` operator, what is the resulting Boolean value?

**12**.  Using the variables defined in #9, write an expression that will combine two `False` Boolean expressions using the `or` operator. Check your work using a Python Shell.

**13**. Assuming `P` and `Q` each represent a Boolean expression that evaluates to the Boolean value indicated, complete the following table. Compare your group's answers with another group's, and resolve any inconsistencies.

| P | Q | P and Q | P or Q |
|---|---|---------|--------|
| False | False | | |
| False | True | | |
| True | False | | |
| True | True | | |

**14**. Consider two Boolean expressions that are combined using the `and` operator. If the value of the first expression is `False`, is it necessary to determine the value of the second expression? Explain why or why not.

**15**. Consider two Boolean expressions that are combined using the `or` operator. If the value of the first expression is `True`, is it necessary to determine the value of the second expression? Explain why or why not.

**16**. Suppose you wanted to print a result only when both `x` and `y` are positive. Determine the appropriate operators, and write a single Boolean expression for the `if`-statement condition.

**17**. Rewrite the expression from #16 using the `not` operator. Your answer should yield the same result as in #16, not the opposite. Describe in words what the new expression means.

**18**. Suppose that your group needs to print a result, except for when both `x` and `y` are positive. Write a Boolean expression for this condition. How is this different from the previous question?