Comparison Operators with Equations

The following examples demonstrate how to use comparison operators with the data types **int** (integers, whole numbers) and **float** (number with a decimal point or fractional value). Comparison operators return Boolean results. As you learned previously, Boolean is a data type that can hold only one of two values: **True** or **False**.

The comparison operators include:

- **==** (equality)
- **!=** (not equal to)
- **>** (greater than)
- **<** (less than)
- **>=** (greater than or equal to)
- **<=** (less than or equal to)

PART 1: Equality **==** and Not Equal To **!=** Operators

In Python, you can use comparison operators to compare values. When a comparison is made, Python returns a Boolean result: **True** or **False**. Note that Boolean data types are not string data types (Boolean **True** is not equal to the string "True").

- To check if two values are the same, use the **equality operator**: **==**
- To check if two values are not the same, use the **not equal to operator**: **!=**

The print() function can be used to display the results of the comparisons.

Examples:

```
False              # is false. So, Python returns a False value.
```

The equality **==** operator versus the equals **=** operator

It is important to note that the equality **==** comparison operator performs a different task than the equals **=** assignment operator. The equals **=** operator assigns the value on the right side of the equals **=** to the object (e.g., a variable) on the left side of the equals **=** operator.

Examples:

```

# The = equals assignment operator is used to assign a value to a
# variable.

my_variable = 3*5           # Assigns a value to my_variable
print(my_variable)          # Printing the variable returns the
15                          # value assigned to the variable.



# The == equality comparison operator checks if the values of the two
# expressions on either side of the == operator are equivalent to one
# another.

print(my_variable == 3*5)   # Printing the variable returns a Boolean
True                        # True or False result.
```

PART 2: Greater Than **>** and Less Than **<** Operators

The comparison operators greater than **>** and less than **<** also return a **True** or **False** Boolean result after comparing two values.

- To check if one value is larger than another value, use the greater than operator: **>**
- To check if one value is smaller than another value, use the less than operator: **<**

Examples:

```

print(11 > 3*3)         # The > operator checks if the left value is
True                    # greater than the right value. If true, it
                        # returns a True result.


print(4/2 > 8-4)        # If the > operator finds that the left value
False                   # is NOT greater than the right value, the
                        # comparison will return a False result.


print(4/2 < 8-4)        # The < operator checks  if the left value is
True                    # less than the right side. If true, the
                        # comparison returns a True result.


```

```
print(11 < 3*3)         # If the < operator finds that the left side is False
                        # a False result.
```

PART 3: Greater Than or Equal to **>=** and Less Than or Equal to **<=** Operators

Like the other comparison operators, the greater than or equal to **>=** and less than or equal to **<=** operators return a **True** or **False** Boolean result when a comparison is made.

- To check if one value is larger than or equal to another value, use the greater than or equal to operator: **>=**
- To check if one value is smaller than or equal to another value, use the less than or equal to operator: **<=**

Examples:

```

print(12*2 >= 24)   # The >= operator checks if the left value is
True                # greater than or equal to the right value.
                    # If one of these conditions is true,
                    # Python returns a True result. In this case
                    # the two values are equal. So, the comparison
                    # returns a True result.


print(18/2 >= 15)   # If the >= comparison determines that the left False
False               # value is NOT greater than or equal to the
                    # right, it returns a False result.

print(12*2 <= 30)   # The <= operator checks if the left value is
True                # less than or equal to the right value. In
                    # this case, the left value is less than the
                    # right value. Again, if one of the two
                    # conditions is true, Python returns a True
                    # result.


print(15 <= 18/2)   # If the <= comparison determines that the left
False               # value is NOT less than or equal to the right
                    # value, the comparison returns a False result.
```

PART 4: Practice

If you would like more practice using the logical (**and**, **or**, **not**) operators, feel free to create your own comparisons using the code block below. Note that there is no feedback associated with this code block.

```

```

Run

Reset

For additional Python practice, the following links will take you to several popular online interpreters and codepads:

- Welcome to Python
- Online Python Interpreter
- Create a new Repl
- Online Python-3 Compiler (Interpreter)
- Compile Python 3 Online
- Your Python Trinket

Key takeaways

Python comparison operators return Boolean results: **True** or **False**.

Symbol	Name	Expression	Description
==	Equality operator	a == b	a is equal to b
!=	Not equal to operator	a != b	a is **not** equal to b
>	Greater than operator	a > b	a is larger than b
>=	Greater than or equal to operator	a >= b	a is larger than or equal to b
<	Less than operator	a < b	a is smaller than b
<=	Less than or equal to operator	a <= b	a is smaller than or equal to b

Resources for more information

For more information about the concepts covered in these practice exercises, please visit:

- Order of Operations in Python with Examples - A refresher in the mathematical Order of Operations and how they work in Python.

- Python Comparison Operators with Syntax and Example - Provides examples of more complex comparisons.
- Raise numbers to a power: here's how to exponentiate in Python - Explains multiple methods for calculating exponents in Python.