Comparison Operators with Strings

In this reading, you will learn more about what comparison operators can and cannot do. If you use the **==** (equality) and **!=** (not equal to) operators with strings, you can check if two strings contain the same text or not. You can also alphabetize strings using **>** (greater than), **<** (less than)**, >=** (greater than or equal to)**, <=** (less than or equal to) comparison operators. As with numeric data types, comparison operators used with strings will return Boolean (**True**, **False**) results.

PART 1: Equality **==** and Not Equal to **!=** Operators with Strings

In Python, you can use comparison operators to compare strings. The equality **==** and the not equal to **!=** operators are helpful when you need to search for a specific string in a body of text, a log file, a spreadsheet, a database, and more. You can also check user input strings to compare them to another string. Note that Boolean data types are not string data types (Boolean **True** is not equal to the string "True").

Examples:

```
# The == operator can check if two strings are equal to each other.
# If they are equal, the Python interpreter returns a True result.
print("a string" == "a string")
True


# In this example, the equality == comparison is between "4 + 5" and
# 4 + 5. Since the left data type is a string and the right data type
# is an integer, the two values cannot be equal. So, the comparison
# returns a False result.
print("4 + 5" == 4 + 5)
False


# The != operator can check if the two strings are NOT equal to each
# other. If they are indeed not equal, then Python returns a True result.
print("rabbit" != "frog")
True


# In this example, the variable event_city has been assigned the string
# value "Shanghai". This variable is compared to a static string,
# "Shanghai", using the != operator. As, the strings "Shanghai" and
# "Shanghai" are the same, the comparison of "Shanghai" != "Shanghai"
# is false. Accordingly, Python will return a False result.
event_city = "Shanghai"
print(event_city != "Shanghai")
False

# This last example illustrates the result of trying to compare two
# items of different data types using the equality == operator. The
# two items are not equal, so the comparison returns False.
print("three" == 3)
False

```

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

The comparison operators greater than **>** and less than **<** can be used to alphabetize words in Python. The letters of the alphabet have numeric codes in Unicode (also known as ASCII values). The uppercase letters A to Z are represented by the Unicode values 65 to 90. The lowercase letters a to z are represented by the Unicode values 97 to 122.

Uppercase		Uppercase		Lowercase		Lowercase	
Unicode #	Character	Unicode #	Character	Unicode #	Character	Unicode #	Character
65	A	78	N	97	a	110	n
66	B	79	O	98	b	111	o
67	C	80	P	99	c	112	p
68	D	81	Q	100	d	113	q
69	E	82	R	101	e	114	r
70	F	83	S	102	f	115	s
71	G	84	T	103	g	116	t
72	H	85	U	104	h	117	u
73	I	86	V	105	i	118	v
74	J	87	W	106	j	119	w
75	K	88	X	107	k	120	x
76	L	89	Y	108	l	121	y
77	M	90	Z	109	m	122	z

- To check if the first letter(s) of a string have a larger Unicode value (meaning the letter is closer to 122 or lowercase z) than the first letter of another string, use the greater than operator: **>**
- To check if the first letter(s) of a string have a smaller Unicode value (meaning the letter is closer to 65 or uppercase A) than the first letter of another string, use the less than operator: **<**

Like numeric comparisons with the greater than **>** and less than **<** operators, comparisons between strings also return Boolean **True** or **False** results.

Examples:

```
# The greater than > operator checks if the left string has a higher
# Unicode value than the right string. If true, the Python interpreter
# returns a True result. Since W has a Unicode value of 87, and you can
# easily calculate that F has a Unicode value of 70, this comparison is
# the same as 87 > 70. As this is true, Python will return a True
# result.
print("Wednesday" > "Friday")
True


# The less than < operator checks if the left string has a lower
# Unicode value than the right string. If you reference the Unicode
# chart above, you can see that all lowercase letters have higher
# Unicode values than uppercase letters. We can see that B has a
# Unicode value of 66 and b has a Unicode value of 98. This
# comparison is the same as 66 < 98, which is true. So, Python will
# return a True result.
print("Brown" < "brown")
True


# If the strings have the same first few letters, the comparison will
# cycle through each letter of each string, from left to right until it
# finds two letters that have different Unicode values. In this example,
# both strings share the initial substring "sun", but then have
# different letters with different Unicode values in the fourth place
# in each string. So, the fourth letters 'b' and 't' of the two
# strings are used for the comparison. Since 'b' does not have a higher
# Unicode value than 't', the comparison returns a False result.
print("sunbathe" > "suntan")
False


# If two identical strings are compared using the less than < comparison
# operator, this will produce a False result because they are equal.
print("Lima" < "Lima")
False


# This last example illustrates the result of trying to compare two
```

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

The greater than or equal to **>=** and less than or equal to **<=** operators can be used with strings as well. Like the other comparison operators, they will return a **True** or **False** Boolean result when a comparison is made between two strings.

- To check if a string has a larger or equal Unicode value than the first letter(s) of another string, use the greater than or equal to operator: **>=**
- To check if a string has a smaller or equal Unicode value than the first letter(s) of another string, use the less than or equal to operator: **<=**

At this point, you should be familiar with how comparison operators work in Python. Can you determine what the results will be from the comparisons listed below? When you are ready to check your answers, click Run.

1. "my computer" >= "my chair"
2. "Spring" <= "Winter"
3. "pineapple" >= "pineapple"

```
# Use the Unicode chart in Part 2 to determine if the Unicode values of
# the first letters of each string are higher, lower, or equal to one
# another.


var1 = "my computer" >= "my chair"
var2 = "Spring" <= "Winter"
var3 = "pineapple" >= "pineapple"

print("Is \"my computer\" greater than or equal to \"my chair\"? Result: ", var1)
print("Is \"Spring\" less than or equal to \"Winter\"? Result: ", var2)
print("Is \"pineapple\" less than or equal to \"pineapple\"? Result: ", var3)
```

Run

Reset

PART 4: Practice

If you would like more practice using the comparison (**==**, **!=**, **>**, **<**, **>=**, **<=**) operators with strings, 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 are for 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