Advanced String Methods

We've covered a bunch of String class methods already, so let's keep building on those and run down some more advanced methods.

The string method **lower** will return the string with all characters changed to lowercase. The inverse of this is the **upper** method, which will return the string all in uppercase. Just like with previous methods, we call these on a string using dot notation, like **"this is a string".upper()**. This would return the string **"THIS IS A STRING"**. This can be super handy when checking user input, since someone might type in all lowercase, all uppercase, or even a mixture of cases.

You can use the **strip** method to remove surrounding whitespace from a string. Whitespace includes spaces, tabs, and newline characters. You can also use the methods **lstrip** and **rstrip** to remove whitespace only from the left or the right side of the string, respectively.

The method **count** can be used to return the number of times a substring appears in a string. This can be handy for finding out how many characters appear in a string, or counting the number of times a certain word appears in a sentence or paragraph.

If you wanted to check if a string ends with a given substring, you can use the method **endswith**. This will return True if the substring is found at the end of the string, and False if not.

The **isnumeric** method can check if a string is composed of only numbers. If the string contains only numbers, this method will return True. We can use this to check if a string contains numbers before passing the string to the **int()** function to convert it to an integer, avoiding an error. Useful!

We took a look at string concatenation using the plus sign, earlier. We can also use the **join** method to concatenate strings. This method is called on a string that will be used to join a list of strings. The method takes a list of strings to be joined as a parameter, and returns a new string composed of each of the strings from our list joined using the initial string. For example, **" ".join(["This","is","a","sentence"])** would return the string **"This is a sentence"**.

The inverse of the join method is the **split** method. This allows us to split a string into a list of strings. By default, it splits by any whitespace characters. You can also split by any other characters by passing a parameter.