Strings, Lists and Dictionaries - Reading Material

String Reference Cheat Sheet

In Python, there are a lot of things you can do with strings. In this cheat sheet, you'll find the most common string operations and string methods.

String operations

- len(string) Returns the length of the string
- for character in string Iterates over each character in the string
- if substring in string Checks whether the substring is part of the string
- string[i] Accesses the character at index i of the string, starting at zero
- string[i:j] Accesses the substring starting at index i, ending at index j-1. If i is omitted, it's 0 by default. If j is omitted, it's len(string) by default.

String methods

- string.lower() / string.upper() Returns a copy of the string with all lower / upper case characters
- string.lstrip() / string.rstrip() / string.strip() Returns a copy of the string without left / right / left or right whitespace
- string.count(substring) Returns the number of times substring is present in the string
- string.isnumeric() Returns True if there are only numeric characters in the string. If not, returns False.
- string.isalpha() Returns True if there are only alphabetic characters in the string. If not, returns False.
- string.split() / string.split(delimiter) Returns a list of substrings that were separated by whitespace / delimiter
- string.replace(old, new) Returns a new string where all occurrences of old have been replaced by new.
- delimiter.join(list of strings) Returns a new string with all the strings joined by the delimiter

Check out the official documentation for all available String methods.

Formatting Strings Cheat Sheet

Python offers different ways to format strings. In the video, we explained the format() method. In this reading, we'll highlight three different ways of formatting strings. For this course you only need to know the format()

method. But on the internet, you might find any of the three, so it's a good idea to know that the others exist.

Using the format() method

The format method returns a copy of the string where the {} placeholders have been replaced with the values of the variables. These variables are converted to strings if they weren't strings already. Empty placeholders are replaced by the variables passed to format in the same order.

Formatting expressions

Expr	Meaning	Example
{:d}	integer value	'{:d}'.format(10.5) → '10'
{:.2f}	floating point with that many decimals	'{:.2f}'.format(0.5) → '0.50'
{:.2s}	string with that many characters	'{:.2s}'.format('Python') → 'Py'
{:<6s}	string aligned to the left that many spaces	'{:<6s}'.format('Py') → 'Py '
{:>6s}	string aligned to the right that many spaces	'{:<6s}'.format('Py') → ' Py'
{:^6s}	string centered in that many spaces	'{:<6s}'.format('Py') → ' Py '

Check out the official documentation for all available expressions.

Lists and Tuples Operations Cheat Sheet

Lists and tuples are both sequences, so they share a number of sequence operations. But, because lists are mutable, there are also a number of methods specific just to lists. This cheat sheet gives you a run down of the common operations first, and the list-specific operations second.

Common sequence operations

- len(sequence) Returns the length of the sequence
- for element in sequence Iterates over each element in the sequence

- if element in sequence Checks whether the element is part of the sequence
- sequence[i] Accesses the element at index i of the sequence, starting at zero
- sequence[i:j] Accesses a slice starting at index i, ending at index j-1. If i is omitted, it's 0 by default. If j is omitted, it's len(sequence) by default.
- for index, element in enumerate(sequence) Iterates over both the indexes and the elements in the sequence at the same time

Check out the official documentation for sequence operations.

List-specific operations and methods

- list[i] = x Replaces the element at index i with x
- list.append(x) Inserts x at the end of the list
- list.insert(i, x) Inserts x at index i
- list.pop(i) Returns the element a index i, also removing it from the list. If i is omitted, the last element is returned and removed.
- list.remove(x) Removes the first occurrence of x in the list
- list.sort() Sorts the items in the list
- list.reverse() Reverses the order of items of the list
- list.clear() Removes all the items of the list
- list.copy() Creates a copy of the list
- list.extend(other_list) Appends all the elements of other_list at the end of list

Most of these methods come from the fact that lists are mutable sequences. For more info, see the official documentation for mutable sequences and the list specific documentation.

List comprehension

- [expression for variable in sequence] Creates a new list based on the given sequence. Each element is the result of the given expression.
- [expression for variable in sequence if condition] Creates a new list based on the given sequence. Each element is the result of the given expression; elements only get added if the condition is true.

Dictionary Methods Cheat Sheet

Definition

```
x = {key1:value1, key2:value2}
```

Operations

- len(dictionary) - Returns the number of items in the dictionary
- for key in dictionary - Iterates over each key in the dictionary

- for key, value in dictionary.items() - Iterates over each key,value pair in the dictionary
- if key in dictionary - Checks whether the key is in the dictionary
- dictionary[key] - Accesses the item with key key of the dictionary
- dictionary[key] = value - Sets the value associated with key
- del dictionary[key] - Removes the item with key key from the dictionary

Methods

- dict.get(key, default) - Returns the element corresponding to key, or default if it's not present
- dict.keys() - Returns a sequence containing the keys in the dictionary
- dict.values() - Returns a sequence containing the values in the dictionary
- dict.update(other_dictionary) - Updates the dictionary with the items coming from the other dictionary. Existing entries will be replaced; new entries will be added.
- dict.clear() - Removes all the items of the dictionary

Check out the official documentation for dictionary operations and methods.