Web Application Development using Python

Tuples, Sets and Dictionaries

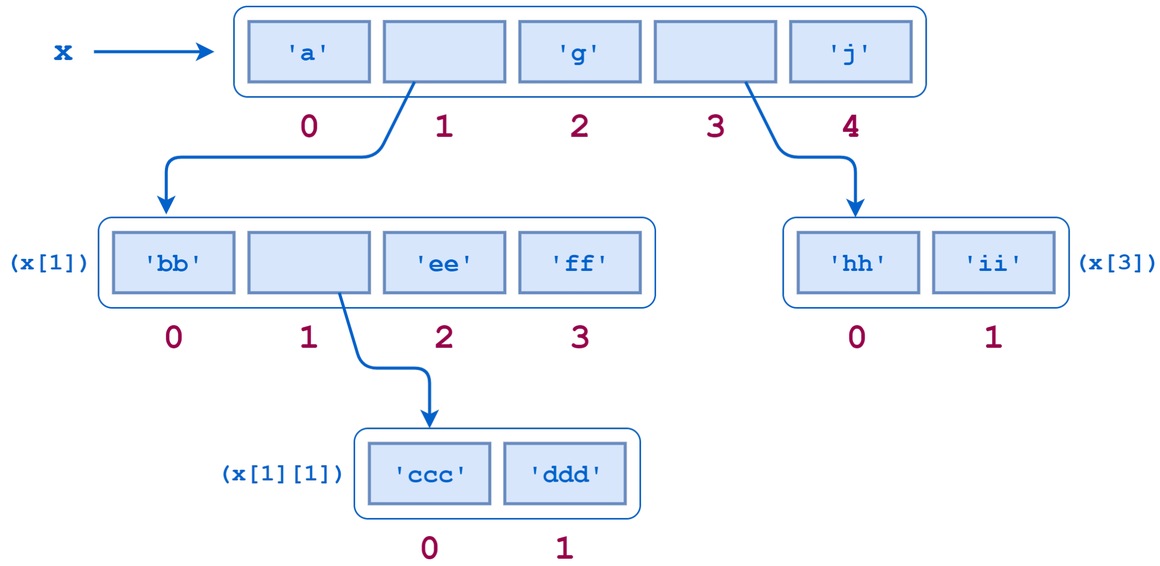
Prepared by George Khoury

Outline

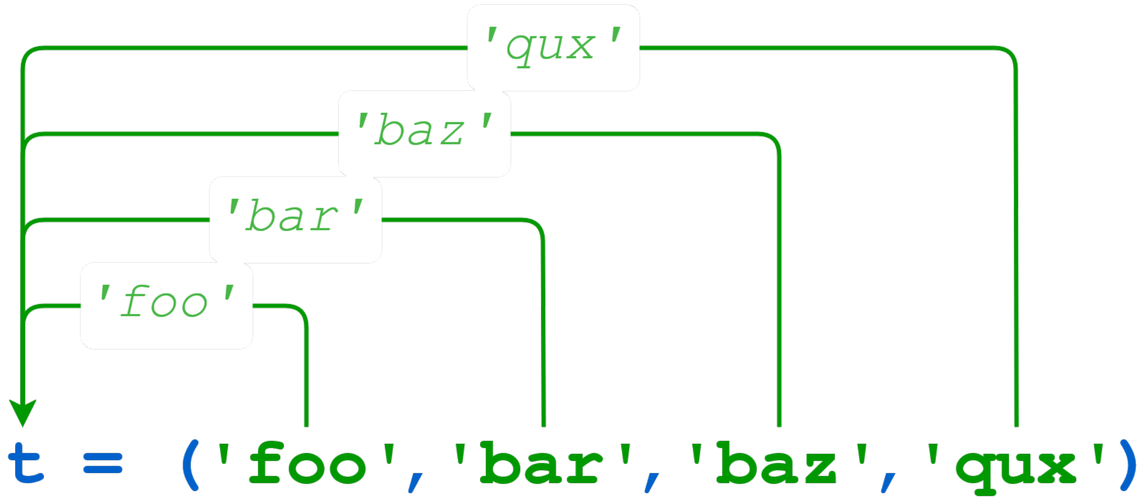
- Review: Lists
- Tuples
- Sets
- Dictionaries

Lists

Review



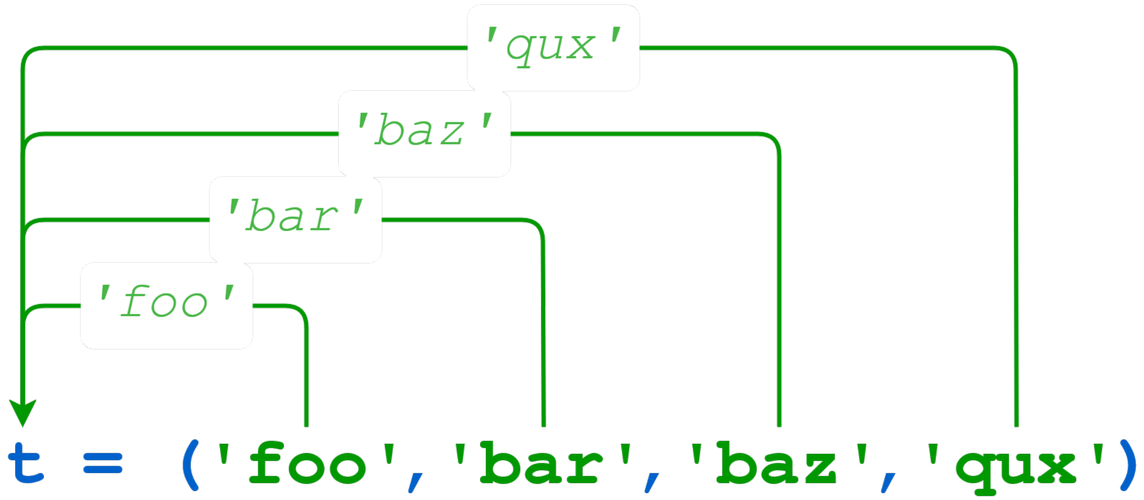
Tuples



Tuples

W2/S3/data_structures/tuples/

- A tuple in Python is similar to a list.
- The difference between the two is that tuples are immutable.
  - We cannot change the elements of a tuple once they are assigned.



Tuples

W2/S3/data_structures/tuples/

- **Packing a tuple**
  - colors = (‘Red’, ‘Blue’, ‘Green’)
- **Unpacking a tuple**
  - color1, color2, color3 = colors
  - print(color1, color2, color3)

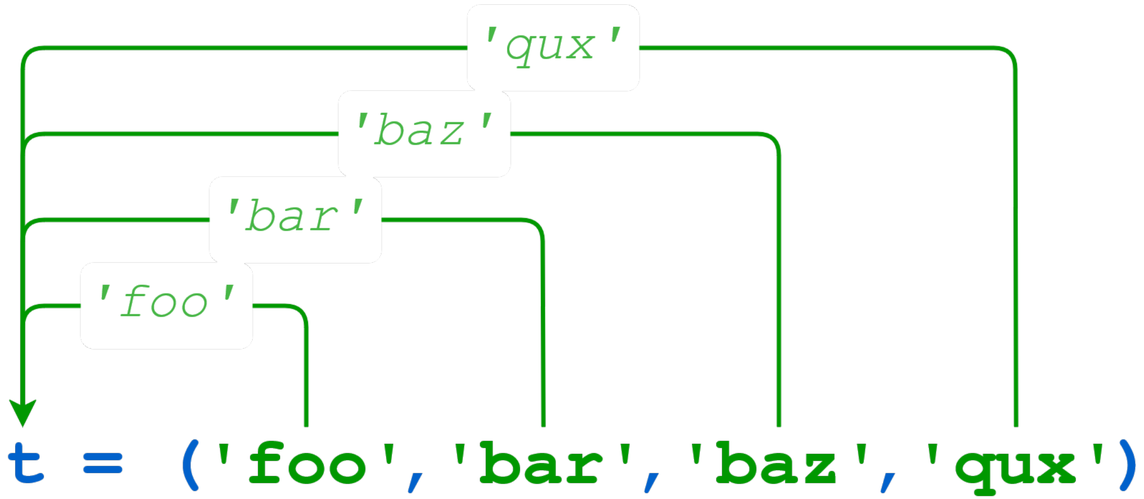
- **We can use positive and negative indexing to access the items in a tuple.**
- **We can use slicing to access a range in a tuple.**

colors =	(‘Red’,	Blue’,	Green’)
Index	0	1	2

Tuple operations

W2/S3/data_structures/tuples/

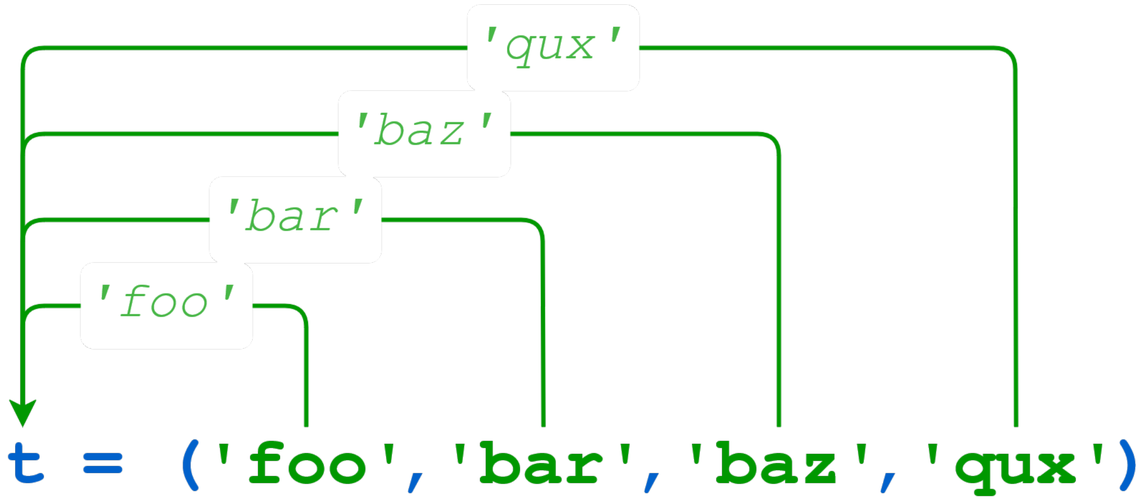
- There are many operations that can be performed with tuples.
  - List length
  - List concatenation
  - List repetition
  - List membership test
  - Other built-in functions



Tuple functions

W2/S3/data_structures/tuples/

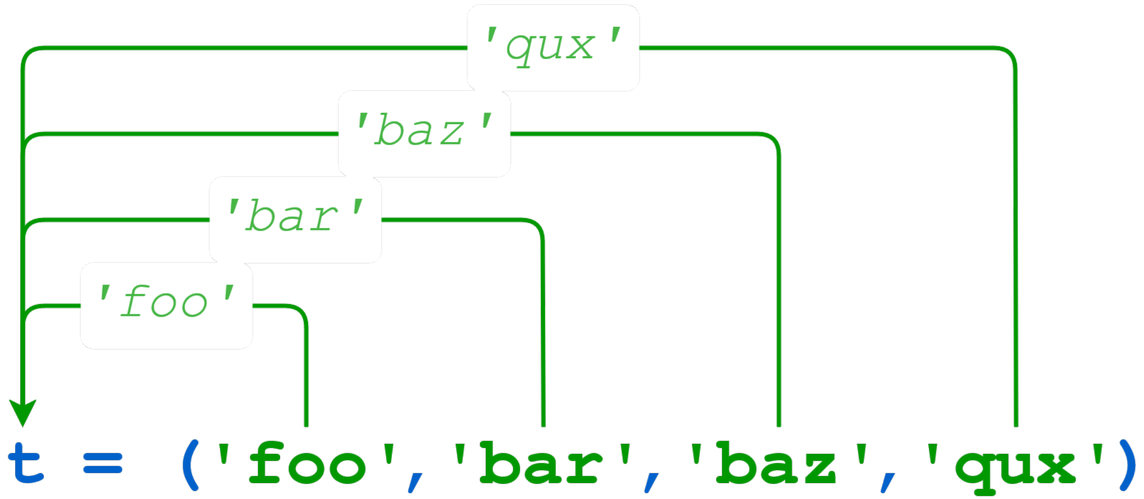
- There are numerous built-in functions available to use with tuples.
- Some of the commonly used methods are
  - **len()**
  - **min(), max()**
  - **sum()**
- Tuples implement all of the common sequence operations.
  - https://docs.python.org/3.3/library/stdtypes.html?highlight=tuple#typesseq-common



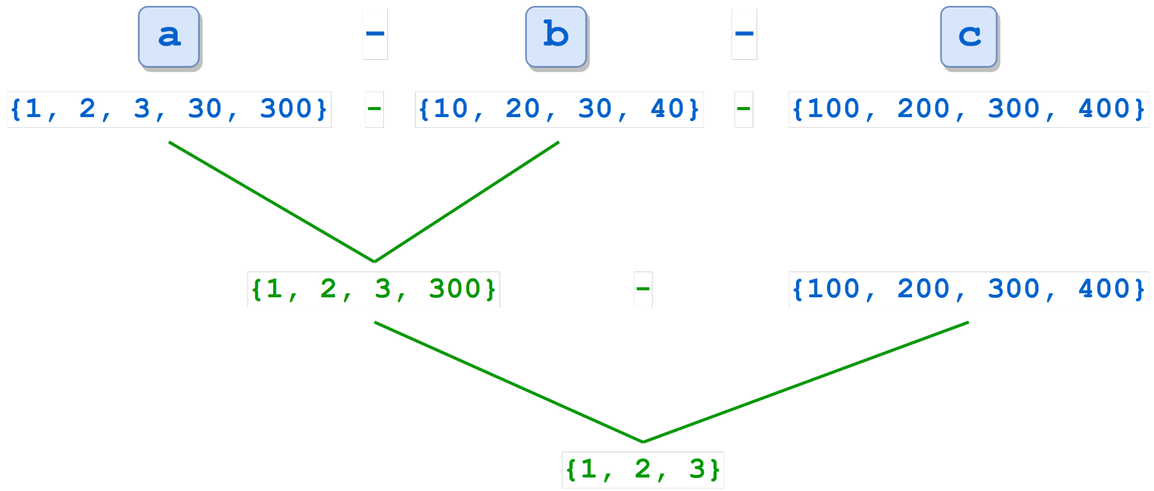
Tuple methods

W2/S3/data_structures/tuples/

- **count(obj)** — Count the number of elements in the list
- **index(obj)** — Returns the index of the first occurrence of an object



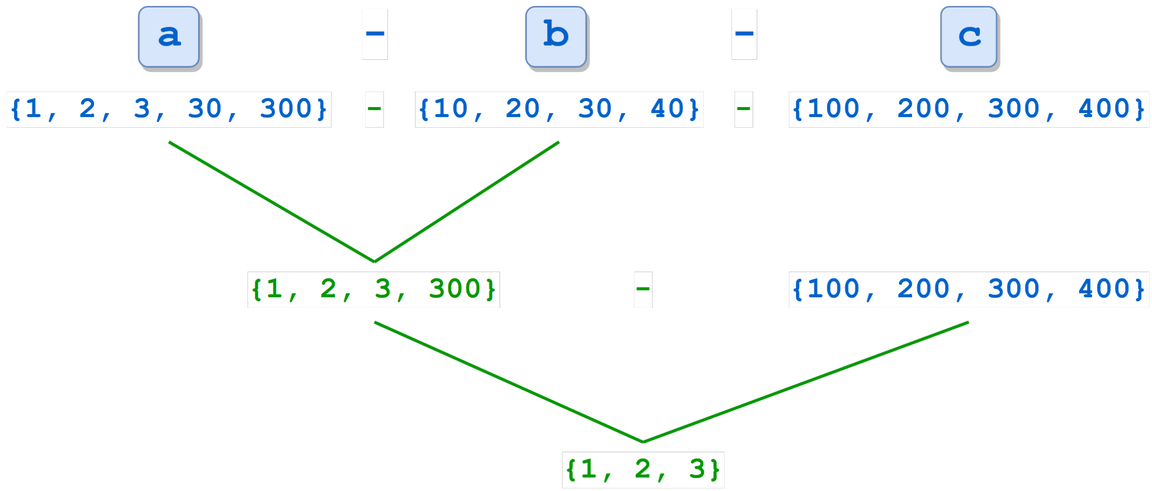
Sets


a
-
b
-
c
{1, 2, 3, 30, 300}
-
{10, 20, 30, 40}
-
{100, 200, 300, 400}
{1, 2, 3, 300}
-
{100, 200, 300, 400}
{1, 2, 3}

Sets

W2/S3/data_structures/sets/

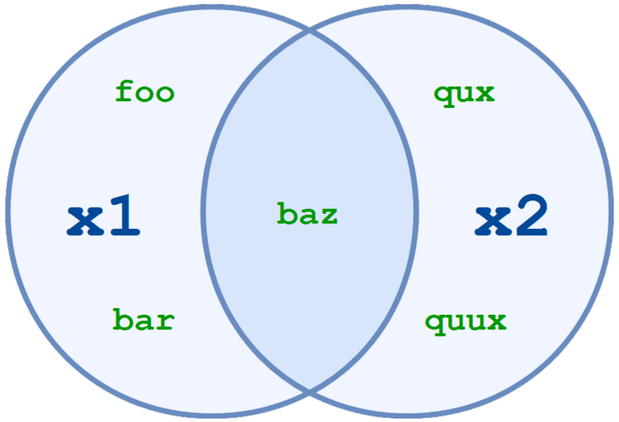
- A set is an unordered collection of items.
- Every set element is unique (no duplicates).
- Sets are mutable.
- Sets can also be used to perform mathematical set operations like union, intersection, symmetric difference, etc.



Sets

W2/S3/data_structures/sets/

- **Creating a set**
  - my_set = {1,2,3,4}
- **Accessing a set**
  - print(my_set)

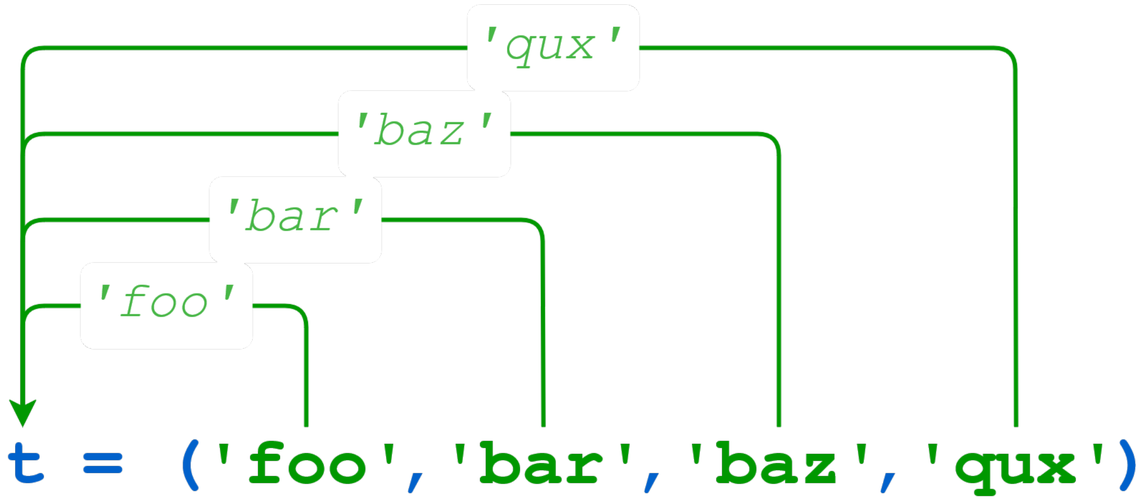


- **Modifying a set**
  - We can add a single element using the **add()** method
  - We can add a multiple elements using the **update()** method
- **Removing elements from a set**
  - Items can be removed from a set using the methods **discard()** and **remove()**

Set operations

W2/S3/data_structures/sets/

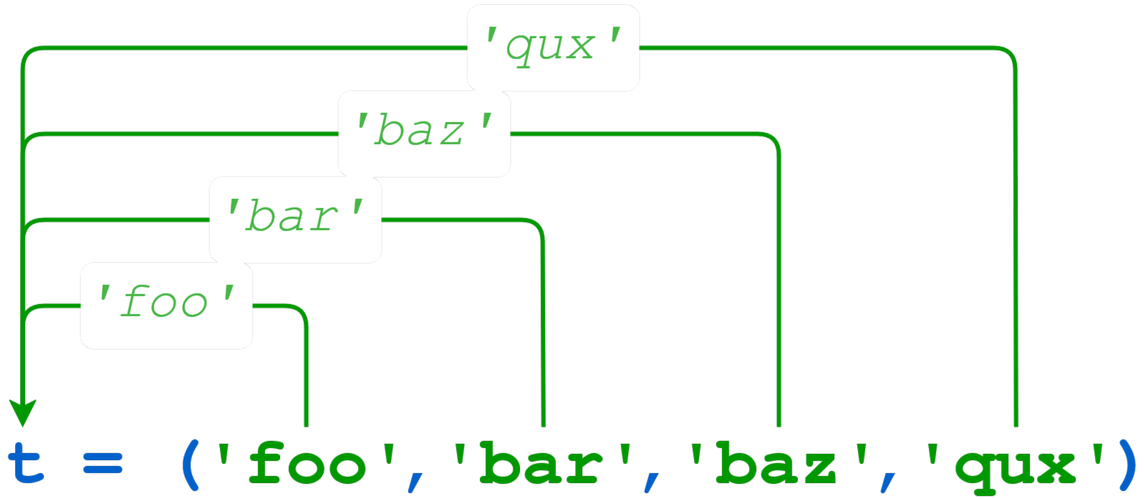
- There are many operations that can be performed with sets.
  - List length
  - List concatenation
  - List repetition
  - List membership test
  - Other built-in functions



Set functions

W2/S3/data_structures/sets/

- There are numerous built-in functions available to use with sets.
- Some of the commonly used methods are
  - **len()**
  - **min(), max()**
  - **sum()**
- Tuples implement all of the common sequence operations.
  - https://docs.python.org/3.3/library/stdtypes.html?highlight=tuple#typesseq-common

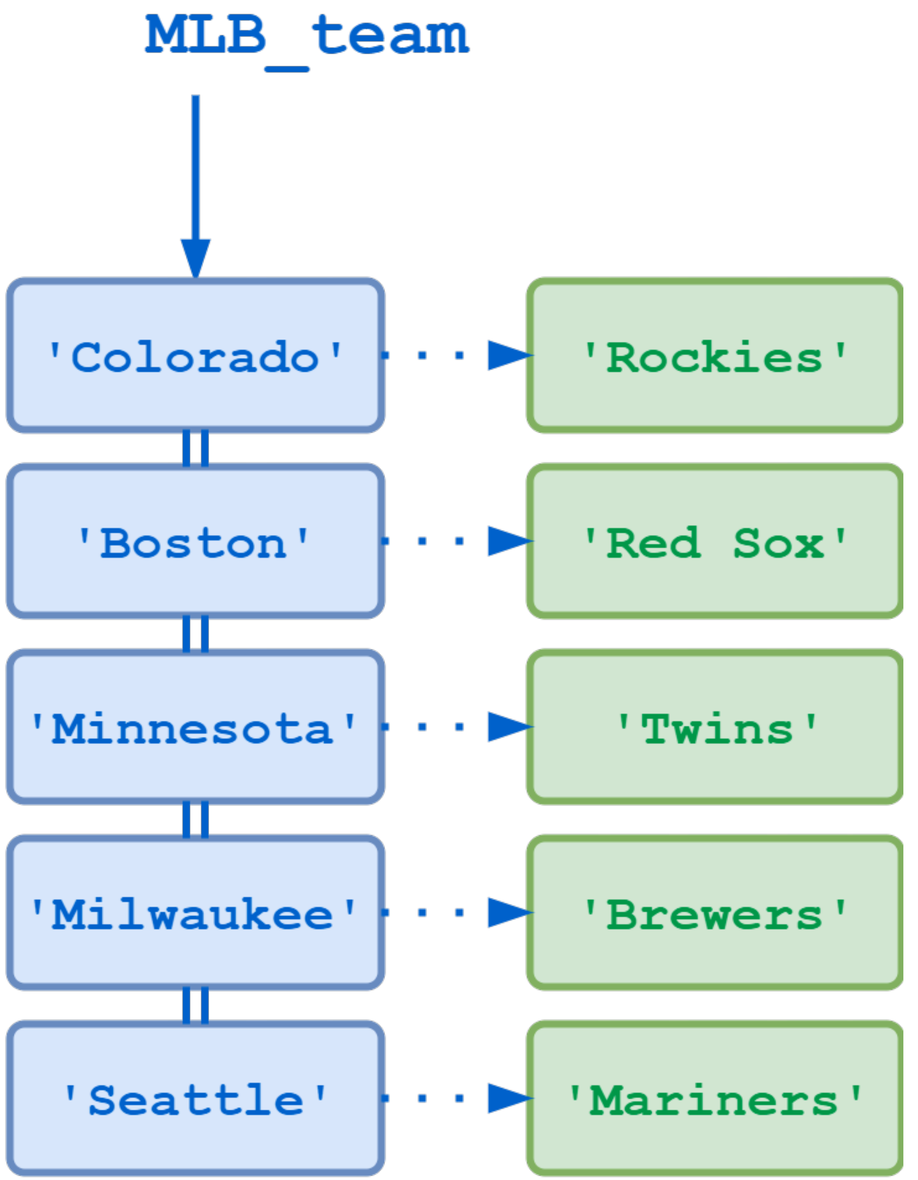


Set Methods

W2/S3/data_structures/sets/

- **add()** — Adds an element to the set
- **clear()** — Removes all elements from the set
- **copy()** — Create a copy of the set
- **pop()** — Removes an arbitrary element from the set
- **remove()** — Removes element from the set
- **discard()** — Checks then removes an element from the set if exists
- **union()** — Returns a new set with the union of both sets
- **intersection()** — Returns a set containing all elements in both sets
- **difference()** — Returns a new set with the difference between the two sets

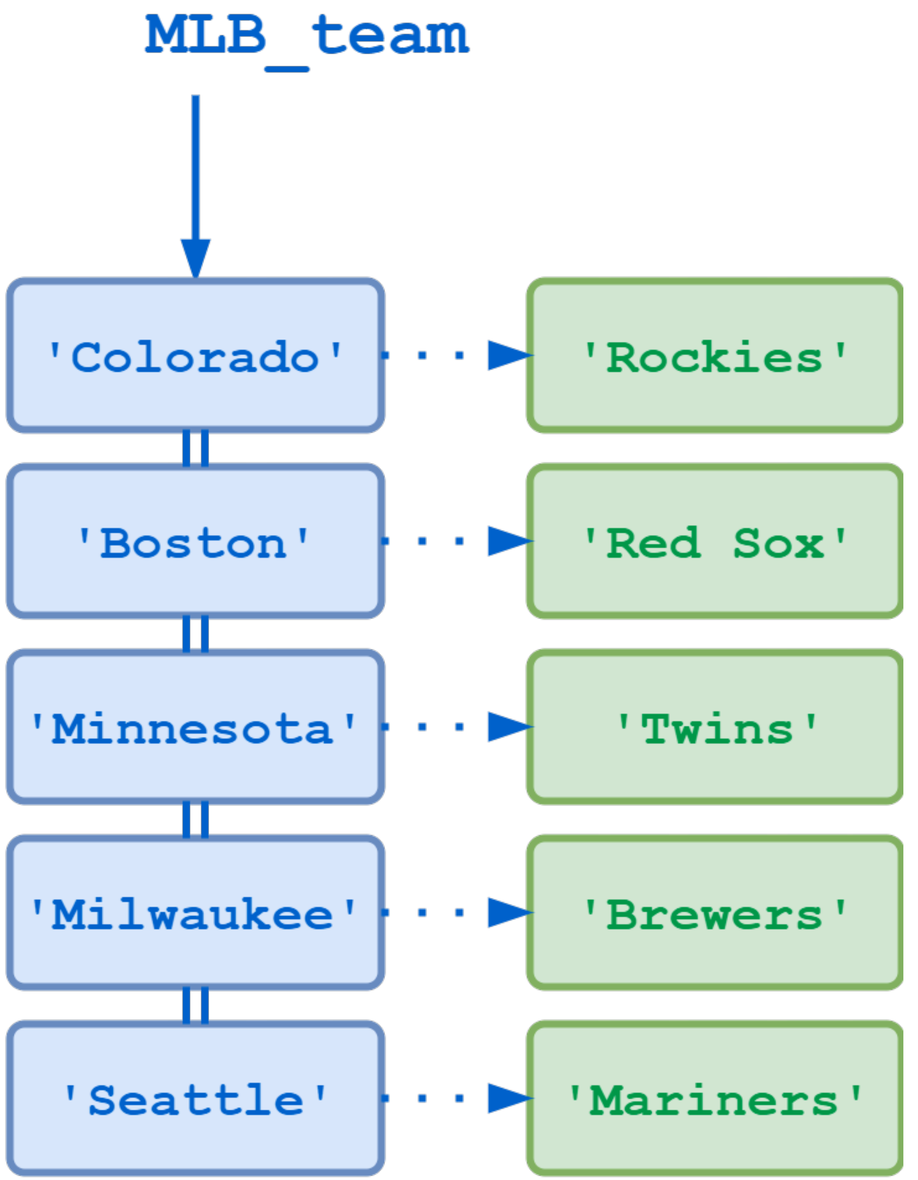
Dictionaries



Dictionaries

W2/S3/data_structures/dictionaries/

- A dictionary is an unordered collection of items.
- Each item of a dictionary has a key/value pair.
- Dictionaries are optimized to retrieve values when the key is known.



Dictionaries

W2/S3/dictionaries/

- **Creating a dictionary**
  - car = {
  - 'brand': 'Maserati',
  - 'model': 'Quattroporte'
  - }
- **Accessing elements**
  - car_model = car[‘model’]
    - **Be careful with invalid keys!**
  - car_model = car.get(‘model’)
- **Getting the list of items**
  - items = car.items()
- **Getting the list of keys**
  - keys = car.keys()
- **Getting the list of values**
  - values = car.values()
- **Updating a value**
  - car[‘model’]= “Levant”

Dictionary Methods

W2/S3/dictionaries/

- **clear()** — Removes all elements from the dictionary
- **copy()** — Returns a shallow copy of the dictionary
- **from_keys()** — Creates a new dictionary from the given sequence of elements with a value provided by the user
- **get(key)** — For key key, returns value or default if key not in dictionary
- **pop(key)** — Removes and returns an element from a dictionary having the given key
- **popitem()** — Removes and returns the (key, value) pair from the dictionary in the Last In, First Out (LIFO) order

Dictionary Methods

W2/S3/dictionaries/

- **update([other])** — Updates the dictionary with the key/value pairs from other, overwriting existing keys
- **values()** — Returns a new object of the dictionary's values
- **keys()** — Returns a new object of the dictionary's keys

Learning Resources

- https://docs.python.org/3/tutorial/introduction.html#lists
- https://docs.python.org/3/tutorial/datastructures.html#more-on-lists
- https://docs.python.org/3/tutorial/datastructures.html#tuples-and-sequences
- https://docs.python.org/3/tutorial/datastructures.html#sets
- https://docs.python.org/3/tutorial/datastructures.html#dictionaries