Dictionary

Dictionaries are unordered sets of key-value pairs. Keys can only be immutable types (strings, numbers, tuples), but their corresponding value can be anything! To create a dictionary, use the following syntax:

```
>>> singers = { 'Adele': 'Hello', 'The Weeknd': ['The Hills', 'Earned It'] }
```

The curly braces denote the key-value pairs in your dictionary. Each key-value pair is separated by a comma. For each pair, the key appears to the left of the colon and the value appears to the right of the colon. Note keys/values do not all have to be the same type, as you can see we have strings, integers and lists! Each key only appears once in a dictionary. You can retrieve values from your dictionary by "indexing" using the key:

```
>>> singers['Adele']
'Hello'
>>> songs = singers['The Weeknd']
>>> songs[0]
'The Hills'
```

You can add an entry or update an entry for an existing key in the dictionary using the following syntax. Note they are identical syntax, so be careful! You might end updating (and overwriting an old value) even if you intended to add, and vice versa.

```
>>> singers['Adele'] = 'Rolling in the Deep'
>>> singers['Adele']
'Rolling in the Deep'
>>> singers['Kanye West'] = 'Real Friends' # New entry!
>>> singers['Kanye West']
'Real Friends'
```

You can also check for membership of keys!

```
>>> 'Adele' in singers
True
```

Finally, here are some useful dictionary functions:

dict.keys() will return a sequence of keys.

```
>>> list(singers.keys()) # We use list() to turn the sequence into a list
['Adele', 'The Weeknd']
```

dict.values() will return a sequence of values.

```
>>> list(singers.values())
['Hello', ['The Hills', 'Earned It']]
```

dict.items() will return a sequences of keys-value pairs.

```
>>> list(singers.items())
[('Adele', 'Hello'), ('The Weekend', ['The Hills', 'Earned It'])]
```