Lecture 05: Data Structures: Dictionaries and Tuples

February 20, 2023

1 Python Data Structure: Dictionaries & Tuples

Ahmed Moustafa

2 Dictionaries

- A dictionary is an unordered collection of key-value pairs.
- Each key is unique and maps to a value.
- The syntax for creating a dictionary is:

[2]:
```
my_dict = {"key1": "value1", "key2": "value2", "key3": "value3"}
my_dict
```

[2]:
```
{'key1': 'value1', 'key2': 'value2', 'key3': 'value3'}
```

You can access values in a dictionary using their keys:

[3]:
```
print(my_dict["key1"])
```

```
value1
```

3 Example

[4]:
```
states = {}                    # Create empty dict
states['ca'] = 'California'    # 1. Set key/value pairs into dict
states['ok'] = 'Oklahoma'
states['nj'] = 'New Jersey'
```

```
states['tx'] = 'Texas'
states
```

[4]:
```
{'ca': 'California', 'ok': 'Oklahoma', 'nj': 'New Jersey', 'tx': 'Texas'}
```

[5]:
```
type(states)
```

[5]:
```
dict
```

```
<IPython.core.display.Image object>
```

Source: Nick Parlante's Python guide

4 Dictionary commonly used functions

- keys(): Returns a list of all the keys in the dictionary
- values(): Returns a list of all the values in the dictionary
- items(): Returns a list of all the key-value pairs in the dictionary

[7]:
```
phonebook = {"Alice": "555-1234", "Bob": "555-5678", "Charlie": "555-9012"}
phonebook
```

[7]:
```
{'Alice': '555-1234', 'Bob': '555-5678', 'Charlie': '555-9012'}
```

[8]:
```
print(phonebook["Alice"])
```

```
555-1234
```

[9]:
```
print(phonebook.keys())
```

```
dict_keys(['Alice', 'Bob', 'Charlie'])
```

[10]:
```
print(phonebook.values())
```

```
dict_values(['555-1234', '555-5678', '555-9012'])
```

[11]:
```
print(phonebook.items())
```

```
dict_items([('Alice', '555-1234'), ('Bob', '555-5678'), ('Charlie',
'555-9012')])
```

[12]:
```
phonebook["John"]
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Cell In[12], line 1
----> 1 phonebook["John"]
```

```
KeyError: 'John'
```

5 Tuples

- A tuple is an ordered collection of values.
- Tuples are immutable, meaning you can't modify their values once they're created.
- The syntax for creating a tuple is:

[13]:
```
my_tuple = ("value1", "value2", "value3")
my_tuple
```

[13]:
```
('value1', 'value2', 'value3')
```

You can access values in a tuple using their index:

[14]:
```
print(my_tuple[0])
```

```
value1
```

6 Tupples commonly used functions:

- `count()`: Returns the number of times a value appears in the tuple
- `index()`: Returns the index of the first occurrence of a value in the tuple

[15]:
```
days_of_week = ("Monday", "Tuesday", "Wednesday", "Thursday", "Friday",␣
↪"Saturday", "Sunday")
days_of_week
```

[15]:
```
('Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday')
```

[16]:
```
print(days_of_week[0])
```

```
Monday
```

[17]:
```
print(days_of_week.count("Monday"))
```

```
1
```

[18]:
```
print(days_of_week.index("Wednesday"))
```

```
2
```

7 Differences between Dictionaries and Tuples

- Dictionaries and tuples are used for different purposes.
- Dictionaries are used to map unique keys to values, while tuples are used to store collections of values.
- Dictionaries are mutable, while tuples are immutable.
- Dictionaries are unordered, while tuples are ordered.

8 More Example

Storing information about a person, such as their name, address, and phone number:

[19]:
```
person = {"name": "Jane Doe", "address": "123 Main St", "phone": "555-1234"}
person
```

[19]:
```
{'name': 'Jane Doe', 'address': '123 Main St', 'phone': '555-1234'}
```

Storing a dictionary of translations between different languages:

[20]:
```
translations = {"hello": {"spanish": "hola", "french": "bonjour"}}
translations
```

[20]:
```
{'hello': {'spanish': 'hola', 'french': 'bonjour'}}
```

Representing a point in two-dimensional space:

[21]:
```
point = (3, 5)
point
```

[21]:
```
(3, 5)
```

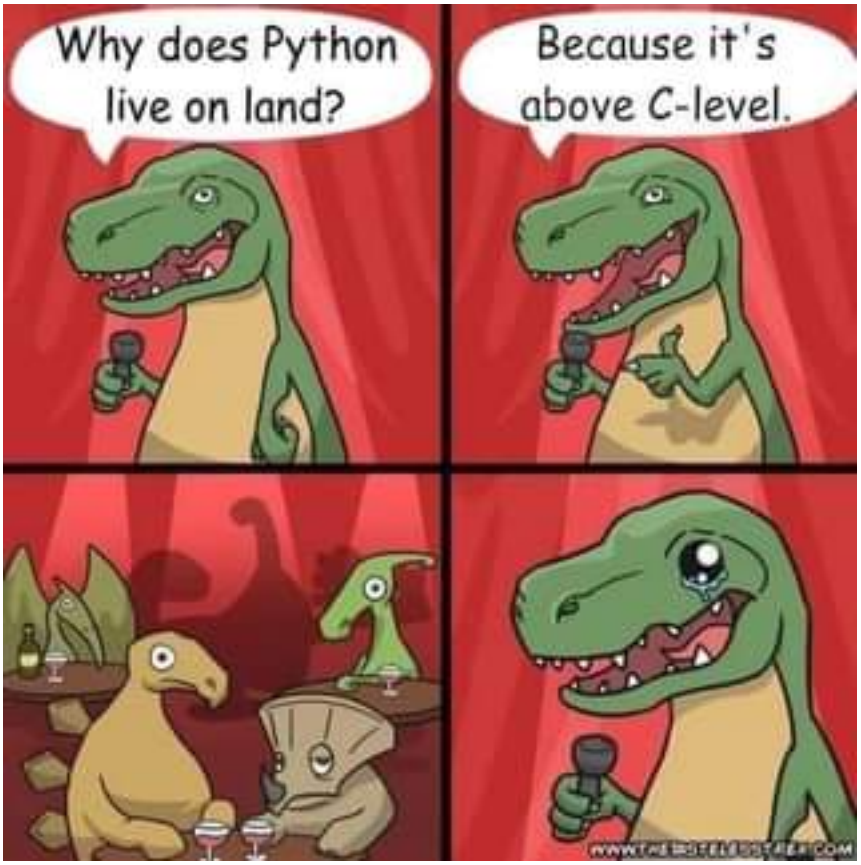
Storing the RGB color values for a particular color:

[22]:
```
color = (255, 128, 0)
color
```

[22]:
```
(255, 128, 0)
```

[24]:
```
set_background(color)
```

```
<IPython.core.display.HTML object>
```


Why does Python live on land?
Because it's above C-level.