Dictionaries and Frequency Tables: Takeaways

Syntax

- Creating a dictionary:

```
# First way
dictionary = {'key_1': 1, 'key_2': 2}
# Second way
dictionary = {}
dictionary['key_1'] = 1
dictionary['key_2'] = 2
```

- Retrieving individual dictionary values:

```
dictionary = {'key_1': 100, 'key_2': 200}
dictionary['key_1']  # Outputs 100
dictionary['key_2']  # Outputs 200
```

- Checking if a certain value exists in the dictionary as a key:

```
dictionary = {'key_1': 100, 'key_2': 200}
'key_1' in dictionary  # Outputs True
'key_5' in dictionary  # Outputs False
100 in dictionary  # Outputs False
```

- Updating dictionary values:

```
dictionary = {'key_1': 100, 'key_2': 200}
dictionary['key_1'] += 600  # This will change the value to 700
```

- Creating a frequency table for the unique values in a column of a data set:

```
frequency_table = {}
for row in a_data_set:
    a_data_point = row[5]
    if a_data_point in frequency_table:
        frequency_table[a_data_point] += 1
    else:
        frequency_table[a_data_point] = 1
```

Concepts

- We call the index of a dictionary value a **key**. In `'4+': 4433`, the dictionary key is `'4+'`, and the dictionary value is `4433`. As a whole, `'4+': 4433` is a **key-value pair**.
- Dictionary values can be any data type: strings, integers, floats, Booleans, lists, and even dictionaries. Dictionary keys can be almost any data type, except lists and dictionaries. If we use lists or dictionaries as dictionary keys, we'll get an error.
- We can check if a certain value exists in the dictionary as a key using an `in` operator. An `in` expression always returns a Boolean value.

- We also call the number of times a unique value occurs the **frequency**. We call tables that map unique values to their frequencies **frequency tables**.
- When we iterate over a dictionary with a `for` loop, we loop over the dictionary keys by default.

Resources

- Dictionaries in Python