Lists and For Loops: Takeaways ⤴

Syntax

- Creating a list of data points:

```
row_1 = ['Facebook', 0.0, 'USD', 2974676, 3.5]
row_2 = ['Instagram', 0.0, 'USD', 2161558, 4.5]
```

- Creating a list of lists:

```
data = [row_1, row_2]
```

- Retrieving an element of a list:

```
first_row = data[0]
first_element_in_first_row = first_row[0]
first_element_in_first_row = data[0][0]
last_element_in_first_row = first_row[-1]
last_element_in_first_row = data[0][-1]
```

- Retrieving multiple list elements and creating a new list:

```
row_1 = ['Facebook', 0.0, 'USD', 2974676, 3.5]
rating_data_only = [row_1[3], row_1[4]]
```

- Performing list slicing:

```
python
row_1 = ['Facebook', 0.0, 'USD', 2974676, 3.5]
second_to_fourth_element = row_1[1:4]
```

- Opening a dataset file, using it to create a list lists and closing the file:

```
opened_file = open('AppleStore.csv')
from csv import reader
read_file = reader(opened_file)
opened_file.close()
apps_data = list(read_file)
```

- Repeating a process using a for loop:

```
row_1 = ['Facebook', 0.0, 'USD', 2974676, 3.5]
for data_point in row_1:
    print(data_point)
```

Concepts

- A **data point** is a value that offers us some information.
- A set of data points make up a **dataset**. A table is an example of a dataset.
- **Lists** are data types which we can use to store datasets.
- Repetitive processes can be automated using **for loops**.

Resources

- [Python Lists](#)
- [Python For Loops](#)
- [More on CSV files](#)
- [A list of keywords in Python](#) — `for` and `in` are examples of keywords (we used `for` and `in` to write for loops)
- [Why should I close files in Python?](#)