String Operations: Takeaways

- Split a string into a list on the newline delimiter

```
lyrics = ' white lips, pale face \n breathing in snowflakes \n burnt lungs sour taste \n lights gone days end \n struggling to pay rent \n long nights strange men \n and they say \n shes in the class a team'
a_team = lyrics.split("\n")
```

–Lowercasing Strings

```
ed_sheeran = [list(line) for line in a_team]
for line in ed_sheeran:
    line[1] = line[1].lower()
```

- Uppercasing Strings

```
ed_sheeran = [list(line) for line in a_team]
for line in ed_sheeran:
    line[1] = line[1].upper()
```

- Joining a list of strings into one string

```
ed_sheeran = ["".join(line) for line in ed_sheeran]
ed_sheeran_full = "\n".join(ed_sheeran)
```

- String concatenation

```
ed_sheeran_full = ''
for line in ed_sheeran:
    ed_sheeran_full += line + "\n"
```

- Replacing values in a string

```
ed_sheeran_full.replace("angel", "wing")
```

Concepts

- When building a more intricate pipeline of transformations, it's often helpful to first plan out your logic using **psuedo-code**.
- Python has rich support for a wide range of string operations, including replacing values within a string, formatting values, and converting between string and list representations.

Resources

- Common String Operations