Module 7: List Comprehension

1

List Comprehension

- List comprehension is useful for efficient and concise list creation
  - Faster than a for loop due to C implementation.
  - Sometimes they are less readable than for loops.
  - Very useful for reading files.
  - There is also dictionary comprehension.
- We will begin with somewhat arbitrary uses and then move to more application that can make certain tasks way easier!

2

List Comprehension

- Basic use - let's create a list of the first 10 multiples of 3:

Before:

```
#For loop
mult_three = []
for i in range(10):
    mult_three.append(3*i)

mult_three
```

```
[0, 3, 6, 9, 12, 15, 18, 21, 24, 27]
```

Now:

```
#List comprehension
mult_three = [i*3 for i in range(10)]

mult_three
```

```
[0, 3, 6, 9, 12, 15, 18, 21, 24, 27]
```

3

Closer Look

- Let's take a closer look at the list comprehension

"Place i*3 in the list for each i from 0 to 9"

```
#List comprehension
mult_three = [i*3 for i in range(10)]
```

4

Adding an if Statement

- We can throw in an if statement in the list comprehension. Let's say we only want even multiples of 3:

```
even_threes = [i*3 for i in range(10)\
               if (i*3)%2==0]

even_threes
```

```
[0, 6, 12, 18, 24]
```

5

Dictionary Comprehension

- We also employ these ideas using dictionaries:

```
name_list =["Jake", "Joe"]
D = {name:[] for name in name_list}
D
```

```
{'Jake': [], 'Joe': []}
```

```
name_list =["Jake", "Joe", "Max"]
D = {name:[] for name in name_list\
    if name[0] == "J"}
D
```

```
{'Jake': [], 'Joe': []}
```

6