

Iterations and List Comprehensions: Takeaways



by Dataquest Labs, Inc. - All rights reserved © 2018

Syntax

ADD COLUMNS TO LIST OF LISTS:

```
'''
door_count = [4, 4]
cars = [
    ["black", "honda", "accord"],
    ["red", "toyota", "corolla"]
]
for i, car in enumerate(cars):
    car.append(door_count[i])
'''
```

LIST COMPREHENSIONS

- Before condensing the loop:

```
animals = ["Dog", "Tiger", "SuperLion", "Cow", "Panda"]
animal_lengths = []
for animal in animals:
    animal_lengths.append(len(animal))
```

- After condensing the loop using a list comprehension:

```
animal_lengths = [len(animal) for animal in animals]
```

THE ITEMS METHOD

- To access the keys and values of a dictionary, use the `items()` method:

```
fruits = {  
    "apple": 2,  
    "orange": 5,  
    "melon": 10  
}  
  
for fruit, rating in fruits.items():  
    print(rating)
```

Concepts

- To loop through multiple lists, use the **`enumerate()`** function. Enumerate adds a counter to an iterable, resulting in a tuple.
- To condense a for loop into one line, use a **list comprehension**. A list comprehension is a more concise way of iterating over multiple values in a list.
- An easy way of accessing a dictionaries keys and values is the **`items`** method.

Resources

- [Python Documentation: `enumerate\(\)`](#)
- [Python Documentation: `items\(\)`](#)

