Mitchell Dodson
Dr. Ryan Weber
EH 301
Ferbruary 18, 2020

How to write comprehensions in Python

Python comprehensions offer a quick and visually intuitive (but initially somewhat esoteric) method for creating and restructuring iterable objects such as tuples, dictonaries, and lists.

```
In [22]: [ name for name in Dogs.keys() if Dogs[name]["age"] > 6 ]
Out[22]: ['Allie', 'Bard']
```

The example we will use (pictured above) supposes we want a list of the names of dogs older than 6, and refers to the "Dogs"dictionary (defined in the image to the right).

```
Dogs = {
    "Allie":{
        "age":10,
        "breed":"terrier",
        "trained":True,
        },
    "Max":{
        "age":6,
        "breed":"retriever",
        "trained":False,
        },
    "Bard":{
        "age":12,
        "breed":"poodle",
        "trained":True,
        },
    "Axle":{
        "age":3,
        "breed":"pug",
        "trained":False,
        }
    }
```

1. Decide what data type you want your comprehension to create. In the example, the output will be a *list*, denoted by the square brackets surrounding the definition

2. Pick the item that you want to compose your new list with as well as the existing iterable from which you will derive the new list. in the example, we want to make a list of the dogs' names, which are the keys of the Dogs dictionary. Use the **in** operator to specify the parent iterable.

3. Apply a constraint to narrow down the items that get included in the output list. In our example, we use an **if** statement to only select dogs whoge age is over 6

4. If you only want to include items *derived* from the members of the parent iterabel (as shown below), you can modify the selector to modify the selected item

```
In [23]: [ Dogs[name]["breed"] for name in Dogs.keys() if Dogs[name]["age"] > 6 ]
Out[23]: ['terrier', 'poodle']
```

If used correctly, the interpreter will return a new iterable with the specified items. Comprehensions are ultimately clear and easy-to-use but powerful operators; enjoy your ability to find new and creative ways to implement them!