😊 Advanced Dictionaries

1. Unlike some of the other Data Structures we've worked with, most of the really useful methods available to us in Dictionaries have already been explored throughout the labs we have covered so far.
2. The dictionary comprehension is just like List Comprehensions; Dictionary Data Types also support their version of comprehension for quick creation. It is not as commonly used as List Comprehensions, but the syntax is:

[1]:
```
{x:x**2 for x in range(10)}
```

[1]:
```
{0: 0, 1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81}
```

3. Dictionaries can be iterated over using the keys(), values() and items() methods. For example.

[2]:
```
d = {'k1':1,'k2':2}
```

[3]:
```
for k in d.keys():
    print(k)
```

```
k1
k2
```

[4]:
```
for v in d.values():
    print(v)
```

```
1
2
```

[5]:
```
for item in d.items():
    print(item)
```

```
('k1', 1)
('k2', 2)
```

4. Using the below code we are going to view keys, values, and items.
5. By themselves the keys(), values() and items() methods return a dictionary *view object*. This is not a separate list of items. Instead, the view is always tied to the original dictionary.

[6]:
```
key_view = d.keys()

key_view
```

[6]:
```
dict_keys(['k1', 'k2'])
```

[7]:
```
d['k3'] = 3

d
```

[7]:
```
{'k1': 1, 'k2': 2, 'k3': 3}
```

[8]:
```
key_view
```

[8]:
```
dict_keys(['k1', 'k2', 'k3'])
```