C:\Users\Terry\AppData\Local\Programs\Python\Python37\python.exe:

‘**creates a dictionary**  
>>> counties\_dict = {}  
  
‘**Fills a dictionary**  
>>> counties\_dict["Arapahoe"] = 422829  
  
“**displays a dictionary**  
>>> counties\_dict  
{'Arapahoe': 422829}  
  
“**adds more to the dictionary**>>> counties\_dict["Denver"] = 463353  
>>> counties\_dict["Jefferson"] = 432438  
  
‘**displays a dictionary**>>> counties\_dict  
{'Arapahoe': 422829, 'Denver': 463353, 'Jefferson': 432438}  
  
‘**Length of a dictionary**  
>>> len(counties\_dict)

3

**If we add the items() method to the end of counties\_dict, we'll get this output:**

>>> counties\_dict.items()

dict\_items([('Arapahoe', 422829), ('Denver', 463353), ('Jefferson', 432438)])

In the output, the information inside the dict\_items([]) is what is known as a view object. A view object will give us a snapshot of what is in the dictionary. In this output, we can see each key and their respective values.

**The keys() method will return a view object that contains the keys of the dictionary as a list**

>>> counties\_dict.keys()

dict\_keys(['Arapahoe', 'Denver', 'Jefferson'])

**To retrieve only the values from a dictionary, add the values() method to the end of the dictionary, like this:**

>>> counties\_dict.values()

dict\_values([422829, 463353, 432438])

**Let's "get" the value or the number of registered voters, in Denver County.**

>>> counties\_dict.get("Denver") or can use dictionary\_name[“Denver”]  
463353

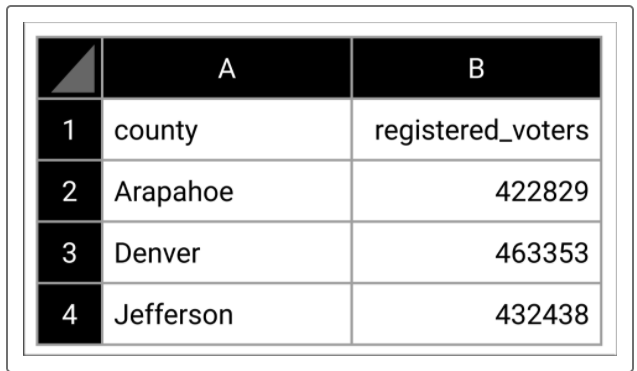
[{key1:value1, key2:value2}, {key1:value3, key2:value4}].

This is referred to as a list of dictionaries because each dictionary, {}, is wrapped in brackets.  
>>> voting\_data = []  
>>> voting\_data.append({"county":"Arapahoe", "registered\_voters": 422829})

>>> voting\_data.append({"county":"Denver", "registered\_voters": 463353})

>>> voting\_data.append({"county":"Jefferson", "registered\_voters": 432438})

Now type >>> voting\_data to see the table of the output



counties.insert(2, "El Paso")

if condition:  
statement 1 ‘must be indented properly  
statement 2

else:  
statement 1  
statement 2

# Determine the grade. Using an elif statement syntax

if score >= 90:  
 print('Your grade is an A.')  
elif score >= 80:  
 print('Your grade is a B.')  
elif score >= 70:  
 print('Your grade is a C.')  
elif score >= 60:  
 print('Your grade is a D.')  
else:  
 print('Your grade is an F.')

Problem with course, this was in the instructions in a table format (membership operators):

counties =   
["Arapahoe","Denver","Jeffers  
on"] if "Arapahoe" in   
counties: print("True") else:  
 print("False")

This prints "True" because Arapahoe is in the counties list.

**in vs not in**

counties = ["Arapahoe","Denver","Jefferson"]  
if "El Paso" in counties:  
 print("El Paso is in the list of counties.")  
else:  
 print("El Paso is not the list of counties.")

**And, Or, Not**

x = 5  
y = 5

if x == 5 **and** y == 5:  
 print("True")   
else:  
 print("False")

x = 5  
y = 5

if **not**(x > y):  
 print("True")   
else:  
 print("False")

**For Loop**  
for item in [items]:  
 statement 1  
 statement 2

**For Loop:** ‘Example 1  
numbers = [0, 1, 2, 3, 4]  
for **county** in **counties**:  
 print(**county**)

**For Loop:** ‘Example 2  
numbers = [0, 1, 2, 3, 4]  
for **num** in **numbers**:  
 print(**num**)

**For Loop:** ‘Example 3  
numbers = [0, 1, 2, 3, 4]  
for **num** in range(5):  
 print(**num**)

**For Loop:** ‘Example 4  
counties = ["Arapahoe","Denver","Jefferson"]  
for i in range(len(counties)):  
 print(counties[i])

for county in counties\_dict: ‘for a dictionary  
 print(county)

Arapahoe  
Denver  
Jefferson

for county in counties\_dict.**keys**(): ‘for the keys in a dictionary  
 print(county)

422829  
463353  
432438

**dictionary\_name[key]**  
for county in counties\_dict:  
 print(counties\_dict[**county**])

**get() method**for county in counties\_dict:  
 print(counties\_dict.**get**(county))

**Items method**  
for county, voters in counties\_dict.items():  
 print(county, voters)  
**output:**  
Arapahoe 422829  
Denver 463353  
Jefferson 432438

voting\_data = [{"county":"Arapahoe", "registered\_voters": 422829},  
 {"county":"Denver", "registered\_voters": 463353},  
 {"county":"Jefferson", "registered\_voters": 432438}]

for county\_dict in voting\_data:  
 print(county\_dict)

**Range**numbers = [0, 1, 2, 3, 4]  
for num in range(5): **‘Range is each element within the dictionary so** Print(num) **‘would give you 1-5 on separate lines.**

**How would you use the range() function to iterate over the list of dictionaries and print the counties in voting\_data?**for i in range(len(voting\_data)): ‘3 records of voting data so range covers all 3 records or the table.

print(voting\_data[i])

**Nested for loops (get to the record then to the column)**  
for county\_dict in voting\_data:  
 for value in county\_dict.values():  
 print(value)

Arapahoe  
422829  
Denver  
463353  
Jefferson  
432438

**F String**

percentage\_votes = (my\_votes / total\_votes) \* 100  
print("I received " + str(percentage\_votes)+"% of the total votes.")

becomes  
print(f"I received {my\_votes / total\_votes \* 100}% of the total votes.")

{candidate\_votes:,} ‘the :, tells python to format the number using a comma  
{candidate\_votes / total\_votes \* 100:.2f} ‘the :.2f tells python to format to a precision of hundredths.

For county, voters in county\_dictionary.items():

Print (f”{county} county has {voters:,} registered voters.”)

Skill drill @bottom of 3.2.11

counties\_dict = {"Arapahoe": 422829, "Denver": 463353, "Jefferson": 432438}  
For county, voters in county\_dictionary.items():  
Print (f”{county} county has {voters:,} registered voters.”)

voting\_data = [``  
{"county":"Arapahoe", "registered\_voters": 422829},   
{"county":"Denver", "registered\_voters": 463353},   
{"county":"Jefferson", "registered\_voters": 432438}]

for i in range(len(voting\_data)):  
Print (f”{voting\_data.county} county has {voting\_data.voters:,} registered voters.”)