Working with Dictionaries - Lab

Introduction

Now that we know about dictionaries in Python, it is time to practice using them. In this lesson, you'll use your knowledge of dictionaries to retrieve and assign data about various cities.

Objectives

You will be able to:

- Assign values in a dictionary
- Access keys and values in a dictionary

Instructions

Here is a dictionary representing the city of Greenville, North Carolina. The area is in kilometers squared.

```
greenville = {'Area': 68, 'City': 'Greenville', 'Country': 'USA', 'Population': 8455
```

> Remember to press shift + enter to run the code.

Let's retrieve the population of the city and assign it to the variable `greenville_population`.

```
greenville_population = greenville['Population'] # change None
greenville_population # 84554
```

```
84554
```

Now retrieve the area of Greenville and assign it to the variable `greenville_area`.

```
greenville_area = greenville['Area']
greenville_area # 68
```

```
68
```

Now let's take a look at all of the keys in the `greenville` dictionary and coerce them into a list. Assign this variable to the list `city_keys`.

```
city_keys = list(greenville.keys())
city_keys # ['Area', 'City', 'Country', 'Population']
```

```
['Area', 'City', 'Country', 'Population']
```

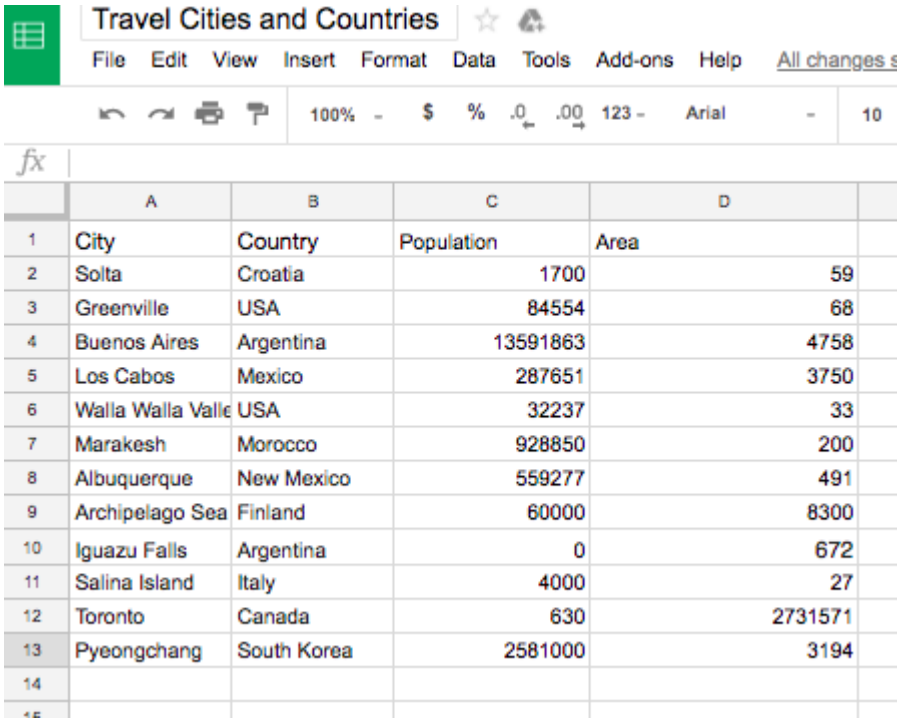
Alright, next let's get all of the values in our greenville dictionary and coerce it into a list. Assign that list to the variable `city_values`.

```
city_values = list(greenville.values())
city_values # [68, 'Greenville', 'USA', 84554]
```

```
[68, 'Greenville', 'USA', 84554]
```

Working with multiple cities

We can retrieve our data from an excel or Google sheet like the one shown here named Travel Cities and Countries.



	A	B	C	D
1	City	Country	Population	Area
2	Solta	Croatia	1700	59
3	Greenville	USA	84554	68
4	Buenos Aires	Argentina	13591863	4758
5	Los Cabos	Mexico	287651	3750
6	Walla Walla Valle	USA	32237	33
7	Marakesh	Morocco	928850	200
8	Albuquerque	New Mexico	559277	491
9	Archipelago Sea	Finland	60000	8300
10	Iguazu Falls	Argentina	0	672
11	Salina Island	Italy	4000	27
12	Toronto	Canada	630	2731571
13	Pyeongchang	South Korea	2581000	3194
14				

Luckily for us, we already have the spreadsheet downloaded and located in the current folder. You can find the file (`cities.xlsx`) in this lesson's GitHub repository. Next, we will use a library called **Pandas** to get this data from the excel file into Python code. We already have the code for reading an excel file into Python written for us below. Let's check it out.

> **Note:** To import a library or module in Python, we do so by writing `import` followed by the name of the thing we want to import. We can optionally include an *alias* for our import, which is done by writing **as** after the name of the thing we are importing followed by the name we would like to use for our *alias*. **Do not worry** about aliases right now. Just know that the *convention* for importing the Pandas library is to import it and alias it as `pd` like we see below.
> We'll talk about packages and Pandas specifically in much more detail soon enough!

```
import pandas as pd
file_name = './cities.xlsx'
travel_df = pd.read_excel(file_name)
cities = travel_df.to_dict('records')
```

> Remember to press shift + enter.

Great! We just used Pandas to read the data from our excel file and turn each row of data into a dictionary. Again, don't worry about exactly how Pandas is doing this, but know that Pandas is a great tool when trying to accomplish a task such as turning data from an excel file into data we can use in Python.

Run the cell below to see what our data looks like now.

```
cities
```

```
[{'City': 'Solta', 'Country': 'Croatia', 'Population': 1700, 'Area': 59},
 {'City': 'Greenville', 'Country': 'USA', 'Population': 84554, 'Area': 68},
 {'City': 'Buenos Aires',
  'Country': 'Argentina',
  'Population': 13591863,
  'Area': 4758},
 {'City': 'Los Cabos',
  'Country': 'Mexico',
  'Population': 287651,
  'Area': 3750},
 {'City': 'Walla Walla Valley',
  'Country': 'USA',
  'Population': 32237,
  'Area': 33},
 {'City': 'Marakesh', 'Country': 'Morocco', 'Population': 928850, 'Area': 200},
 {'City': 'Albuquerque',
  'Country': 'New Mexico',
  'Population': 559277,
  'Area': 491},
 {'City': 'Archipelago Sea',
  'Country': 'Finland',
  'Population': 60000,
  'Area': 8300},
 {'City': 'Iguazu Falls',
  'Country': 'Argentina',
  'Population': 0,
  'Area': 672},
 {'City': 'Salina Island', 'Country': 'Italy', 'Population': 4000, 'Area': 27},
 {'City': 'Toronto', 'Country': 'Canada', 'Population': 630, 'Area': 2731571},
 {'City': 'Pyeongchang',
  'Country': 'South Korea',
  'Population': 2581000,
  'Area': 3194}]
```

Now we will work with reading and manipulating this list of dictionaries, `cities`.

Working with our list of cities

First, access the third to last element and set it equal to the variable `salina`.

```
salina = cities[-3]
salina
# {'Area': 27, 'City': 'Salina Island', 'Country': 'Italy', 'Population': 4000}
```

```
{'City': 'Salina Island', 'Country': 'Italy', 'Population': 4000, 'Area': 27}
```

Now access the fourth city in the list, and set its population equal to a variable called `los_cabos_pop`.

```
los_cabos_pop = cities[3]['Population']
los_cabos_pop # 287651
```

```
287651
```

Now calculate the number of cities in the list and assign the number to the variable `city_count`.

```
city_count = len(cities)
city_count # 12
```

```
12
```

Finally, change the spelling of the South Korean city, Pyeongchang, to the string `'PyeongChang'`, its alternative spelling.

```
cities[11]['City'] = "PyeongChang"
cities[11]['City'] # 'PyeongChang'
```

```
'PyeongChang'
```

Now let's work on retrieving a collection of information about a dictionary. Use the appropriate dictionary method to return a list of values in the dictionary regarding Pyeongchang. Assign the list to the variable `pyeongchang_values`.

```
pyeongchang_values = list(cities[11].values())

pyeongchang_values # ['PyeongChang', 'South Korea', 2581000, 3194]
type(pyeongchang_values) # list
```

```
list
```

And now set `pyeongchang_keys` equal to a list of keys in the dictionary regarding Pyeongchang.

```
pyeongchang_keys = list(cities[11].keys())

pyeongchang_keys # ['City', 'Country', 'Population', 'Area']
type(pyeongchang_keys) # list
```

```
list
```

Summary

In this section, we saw how to assign, retrieve, and re-assign data in a dictionary. We saw how we can retrieve a collection of information from a dictionary, like a list of its keys and values, and we saw how we can work with a list of dictionaries.

Releases

No releases published

Packages

No packages published

Contributors 7

Environments 1

github-pages Active

Languages

- Jupyter Notebook 86.7%
- Python 13.3%