Intermediate

# Python For Data Science

April, 2018

GSA

D2D

DATA TO DECISIONS

## Questions?

Plotting

Pandas

Dictionaries

Conditionals

Boolean Logic

Loops & Control Structures

## Exercise 1 (Code Along)

### Given the following lists:

x1 = [10,20,30]

y1 = [10,40,10]

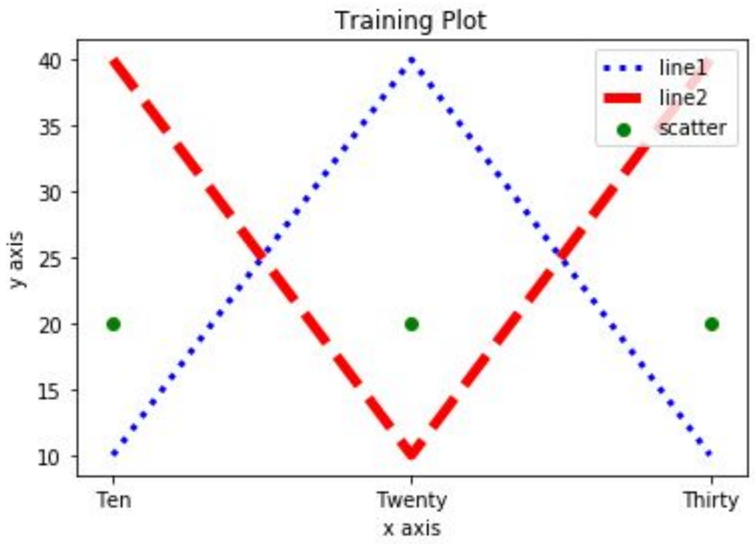
x2 = [10,20,30]

y2 = [40,10,40]

x3 = [10,20,30]

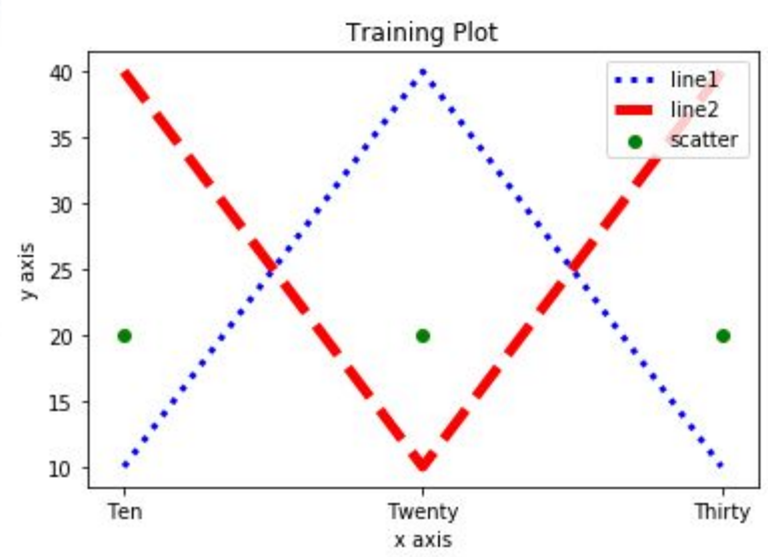
y3 = [20,20,20]

### Use matplotlib to produce the following chart:



## Exercise 1 (Code Along)

```
import matplotlib.pyplot as plt
# line 1 points
X1, y1 = [10,20,30],[10,40,10]
# line 2 points
X2,y2 = [10,20,30],[40,10,40]
# line 3 points
X3,y3 = [10,20,30],[20,20,20]
# Definition of tick_val and tick_lab
tick_val = [10,20,30]
tick_lab = ['Ten','Twenty','Thirty']
# Adapt the ticks on the x-axis
plt.xticks(tick_val,tick_lab)
# Set the x axis label of the current axis.
plt.xlabel('x axis')
# Set the y axis label of the current axis.
plt.ylabel('y axis')
# Plot lines and/or markers to the Axes.
plt.plot(x1,y1, color='blue', linewidth = 3,  label = 'line1',linestyle='dotted')
plt.plot(x2,y2, color='red', linewidth = 5,  label = 'line2', linestyle='dashed')
plt.scatter(x3,y3, color='green', label='scatter')
# Set a title
plt.title("Training Plot")
# show a legend on the plot
plt.legend()
# function to show the plot
plt.show()
```



## Checkpoint

- **matplotlib**:
  - One of the most widely used libraries for plotting in Python
    - Others do exist (ggplot anyone?)
  - The first Python data visualization library
  - While good for getting a “feel” of the data, value is dependent on your goal:
    - Embedding on a report or a paper publication? Go for it
    - Prototyping a dashboard, creating a web interactive chart? Look else where!
  - Very powerful, charts can get very complex!
  - A lot of chart types are available!
  - Official Reference:

    https://matplotlib.org/

DATA TO DECISIONS

## Exercise 2 (Code Along)

**Write a python program that asks user to input a food, and display the calories in that food. Use the following dictionary:**

```
calories = { 'Beef' : 200,
             'Chicken' : 140,
             'Fish' : 120,
             'Apples': 30,
             'Bananas': 15}
```

***Hint:* use the input("Prompt") function to capture user input into a variable**

**Sample*:***

```
What will you eat? Beef
Beef Contains 200 calories
```

## Exercise 2 (Code Along)

**Write a python program that asks user to input a food, and display the calories in that food. Use the following dictionary:**

```
calories = { 'Beef' : 200,
           'Chicken' : 140,
           'Fish' : 120,
           'Apples': 30,
           'Bananas': 15}
```

***Hint:* use the input("Prompt") function to capture user input into a variable**

```
food = input("What will you eat? ")
calories = { 'Beef' : 200,
             'Chicken' : 140,
             'Fish' : 120,
             'Apples': 30,
             'Bananas': 15}
print( food in calories)
print(food+" Contains " +str(calories[food]) +" calories")
```

GSA

D2D
DATA TO DECISIONS

## Exercise 3

### Which of the following is valid?

- x={[1,2]:4,[3,4]:5}
- x={1:'one', 'two':2}
- x,y={1:2,3:4},{5:6,7:8}
  z=x+y
- x={1:2,1:3,1:4,1:5,1:6}

## Exercise 3

**Which of the following is valid?**

- x={[1,2]:4,[3,4]:5} ← Error – unhashable (Mutable)
- x={1:'one', 'two':2} ← Valid
- x,y={1:2,3:4},{5:6,7:8}
  z=x+y ←Error
- x={1:2,1:3,1:4,1:5,1:6} ← Valid

DATA TO DECISIONS

## Exercise 4 (Code Along)

**Convert the following JSON into a python dictionary:**

```
{
    "menu": {
        "id": "file",
        "menuitems": {
      "new": {
             "value": "New"
            },
  },
        "value": "File"
    }
}
```

## Exercise 4 (Code Along)

**Convert the following JSON into a python dictionary:**

```
{
    "menu": {
        "id": "file",
        "menuitems": {
      "new": {
             "value": "New"
            },
  },
        "value": "File"
    }
}
```

```
new_dict={"value":"New"}
menu_items = {"new":new_dict}
menu = {"id":"file", "menuitems":menu_items, "value":"File"}
final={"menu":menu}
print(final)
```

## Exercise 5 (Code Along)

**Create one dictionary by merging the following dictionaries:**

```
d1 = {'a': 100, 'b': 200}
d2 = {'x': 300, 'y': 200, 'a': 500}
```

## Exercise 5 (Code Along)

**Create one dictionary by merging the following dictionaries:**

```
d1 = {'a': 100, 'b': 200}
d2 = {'x': 300, 'y': 200, 'a': 500}

d = d1.copy()
d.update(d2)
print(d)
```

**Question: Why not d=d1?**

GSA

D2D
DATA TO DECISIONS

## Checkpoint

- Dictionaries are Unordered key value pairs
- All Keys MUST be immutable
- Are great for lookups
  - ➢Very efficient too!
- Use in clause to check for key
- Use x.keys() to get all keys
- Use x.values() to get all values
- Add key value pairs by “assignment”
  - ➢X[‘new_value’]=‘New Text’
- Update an existing value by “assignment” (when the value already exists)
  - ➢X[‘old_value’]=‘New Text’

## Checkpoint

**REMEMBER:**

- Python binds variables to **object references**

**Mutable**:

- content of objects of immutable types can be changed after they are created
- More memory is assigned than needed
- Support methods that change the object in place
- Examples: list, set, dict

**Immutable**:

- content of objects of immutable types cannot be changed after they are created
- Hashable!
- Examples: tuple, frozenset, float

## Exercise 6

### Using the demo datasets, Use Pandas to:

- Read data from csv files into  dataframes
  - ➢datagovlserexus.csv
  - ➢datagovbldgrexus.csv
- Merge the two dataframes on LocationCode
- Slice the dataframe to extract specific columns
  - ➢CongressionalDistrict
  - ➢LeaseANSIRentableSqft
  - ➢LeaseAnnualRentAmount
- Filter the sliced dataframe
- Create a new Metric based on column calculations
  - ➢Annual Price/SQFT
- Create some statistics on the dataframe
  - ➢Mean, std….etc
- Loop over dataframe, and print each row
  - ➢Congressional District
  - ➢Annual Price/SQFT

## Exercise 6

- We will be using public data sets from **Data.gov**

- **Public Building Services data sets containing PBS building inventory that consists of both owned and leased buildings with active and excess status.**

- PBS REXUS Buildings:

https://catalog.data.gov/dataset/real-estate-across-the-united-states-rexus-inventory-building

- PBS REXUS Lease:

https://catalog.data.gov/dataset/real-estate-across-the-united-states-rexus-lease

## Exercise 6

### Code:

```
import pandas

# CSV reading from csv into df
df_db = pandas.read_csv(“\\file_location\\datagovlserexus.csv")
df_csv = pandas.read_csv("\\file_location\\datagovbldgrexus.csv")

new_df = pandas.merge(df_db, df_csv, on='LocationCode', how='inner')
sliced_df = new_df.loc[:, ['CongressionalDistrict','LeaseANSIRentableSqft','LeaseAnnualRentAmount']]

sliced_df = sliced_df.loc[(sliced_df['CongressionalDistrict'].isin(['1','2','3'])) &
(sliced_df['LeaseANSIRentableSqft']>200) ]
sliced_df['AnnualPricePerSqft']=sliced_df['LeaseAnnualRentAmount']/sliced_df['LeaseANSIRentableSqft']
std_df = sliced_df.groupby('CongressionalDistrict')['AnnualPricePerSqft'].std()
mean_df = sliced_df.groupby('CongressionalDistrict')['AnnualPricePerSqft'].mean()

#printing the standard deviation dataframe
print(sliced_df.describe())
#looping over dataframe
for index, row in sliced_df.iterrows():
    print(row['CongressionalDistrict'], row['AnnualPricePerSqft'])
```

## Checkpoint

- **pandas is a library providing high-performance, easy-to-use data structures and data analysis tools for Python**
- **The de facto python library when working with heterogeneous tabular data**
- **pandas DataFrame**:
  - ➢The primary pandas data structure
  - ➢Two-dimensional size, mutable, tabular data structure with labeled axes (rows and columns)
  - ➢Provide a common structure for all data sources
  - ➢Can be slices, filtered, merged
  - ➢"Built in" support for matplotlib plotting
    - Under the hood, pandas plots graphs with the matplotlib library
- **Get comfortable with pandas!**
- **pandas reference:**

**https://pandas.pydata.org/**

GSA

D2D
DATA TO DECISIONS

## Exercise 7

Which of the following is valid? And which evaluates to True, and which to False

- "Hello"==‘hello’
- 17<float("18")
- [1,7]<[4,5]
- [4,7]==[4,5]
- ‘3’>=3
- ‘3’!=3
- 1=>True
- 0==False
- not False > (3>4 and 5<6)

## Exercise 7

Which of the following is valid? And which evaluates to True, and which to False

- "Hello"==‘hello’ ← Valid, False
- 17<float("18") ← Valid, True
- [1,7]<[4,5] ← Valid, True
- [4,7]==[4,5] ← Valid, False
- ‘3’>=3 ← Invalid
- ‘3’!=3 ← Valid, True
- 1=>True ← Invalid
- 0==False ← Valid, True
- not False > (3>4 and 5<6) ← Valid, True

## Exercise 8

Rewrite the following While loop using:

1. A while loop with conditions
2. A for loop

```
x = 1
while x < 11:
    print("Round " + str(x))
    x+=1
```

## Exercise 8

Rewrite the following While loop using:

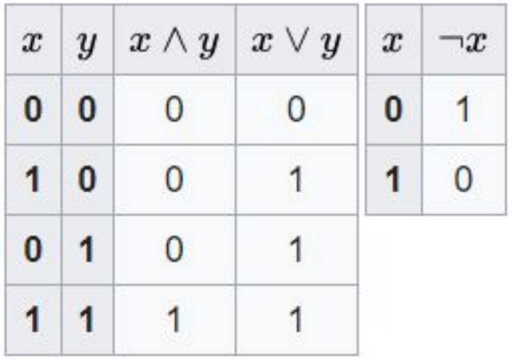
1. A while loop with conditions
2. A for loop

```
#same loop using a while break
x=1
while True:
    print("Round " + str(x))
    if x >= 10:
     break
    else:
     x+=1


#Same loop using for
for x in range(10):
    print("Round" +str(x+1))
```

## Checkpoint

- Conditions can be evaluated using **if, elif, else** statements
- = used for assignment, == used for comparison
- != is the opposite of ==
- Sequence objects may be compared to other objects with the same sequence type. The comparison uses lexicographical ordering
- Logical Operations Truth Table:



| $x$ | $y$ | $x \wedge y$ | $x \vee y$ |
|---|---|---|---|
| **0** | **0** | 0 | 0 |
| **1** | **0** | 0 | 1 |
| **0** | **1** | 0 | 1 |
| **1** | **1** | 1 | 1 |

| $x$ | $\neg x$ |
|---|---|
| **0** | 1 |
| **1** | 0 |

- Loops allow you to execute a block of code several times using **while** or **for..in**
- Else condition in loops are executed when condition is false
- Stop a loop using break
- **Watch out for infinite loops!**

## Checkpoint

- There is a shorter version for an if statement with one else:

```
a,b=5,3
if a>b :
    x=10
else:
    x=11
print(x)
```

**Is the same as:**

```
a,b=5,3
x = 10 if a > b else 11
print(x)
```

# Questions?

# Thank You

GSA

D2D

DATA TO DECISIONS