Intro to Python

SheHacks II

Fatima I @sugaroverflow

Hi I'm Fatima

Tech Lead | Digital Echidna

BS Computer Science | NYU Engr

@sugaroverflow

Python Essentials

dynamic, interpreted language

- no type declarations

case sensitive

object oriented

Syntax

```
hackathon_name = 'SheHacks II'
print hackathon_name # SheHacks II

my_string = 'My favourite hackathon is '
my_string += hackathon_name
new_string = my_string + hackathon_name

# 'My favourite hackathon is SheHacks II'
```

Strings

```
print("Name: %s\
\nDate: %s\
\nWorkshop: %s" % ( 'Fatima', '3/25', 'Intro to Python'))

"""
Name: Fatima
Date: 3/25
Workshop: Intro to Python
"""
```

Data Types

```
# primitives
var_string = 5
var_int = 4.5
var_bool = True
```

Data Types

```
# primitives
var_string = 5
var_int = 4.5
var_bool = True
```

```
# lists
list_empty = []
list_one = [3.0]
list_num = [1, 2, 3, 4, 5]
list_str = ['hello', 'world']
list_misc = ['first', [], 'second', [1, 2, 3, 4], 'third']
print list_str[1] # world
```

Data Types

```
# tuples
var_tuple = (1, 2, 3, 4)
var_tuple[2] # 3
```

Data Types

```
# tuples
var_tuple = (1, 2, 3, 4)
var_tuple[2] # 3
```

```
# dictionaries
var_dictionary = {'A': 'apple', 'B': 'banana'}
print var_dictionary['A'] # apple
```

Conditionals

```
# if
my_name = 'Fatima'
if my_name == 'Fatima':
  return TRUE
```

Conditionals

```
# if
my_name = 'Fatima'
if my_name == 'Fatima':
  return TRUE
```

```
# for
num_list = [1, 2, 3, 4, 5, 6]
for num in num_list:
  if num % 2 == 0:
    print num
    # 2 4 6
```

Conditionals

```
# while
some_list = ['hello', 'world', 'bye']
counter = 0
while counter < len(some_list):
  print counter
  counter += 1
  if some_list[counter] == 'bye': print 'yes!'
  else: print 'nope!'
# nope! yes!
```

Functions

```
# function with parameter
def happyBirthday(person):
    """
    Returns a string using the name given.
    """
    return("Happy Birthday to " + person + "!")
```

Functions

```
# function with parameter
def happyBirthday(person):
    """
    Returns a string using the name given.
    """
    return("Happy Birthday to " + person + "!")
```

```
def main():
  print happyBirthday('Emily') # Happy Birthday to Emily!
```

understanding an interpreted language

```
def main():
    if name == 'Fatima':
        print haaaappyBirthday(name) + '!!!'
    else:
        print happyBirthday(name)
```

Workshop

Getting data from the Wunderground API

- using the `urllib` library

Visualizing that data with Seaborn

- using `Seaborn` viz library based on `matplotlib`

Wunderground API

worldwide weather data, weather reports,

maps & tropical weather conditions

Register for API key

`wunderground.com/weather/api` `aac66ae63af63f1d`

API documentation

`wunderground.com/weather/api/d/docs`

Getting started with Python

create a `weatherdata.py` file in your project:

```
#!/usr/bin/env python

# import modules used here
import sys
```

Getting started with Python

create a `weatherdata.py` file in your project:

```
#!/usr/bin/env python

# import modules used here
import sys
```

```
# Gather our code in a main() function
def main():
    print "Intro to Python Workshop"
```

Getting started with Python

create a `weatherdata.py` file in your project:

```
#!/usr/bin/env python

# import modules used here
import sys
```

```
# Gather our code in a main() function
def main():
    print "Intro to Python Workshop"
```

```
# call the main() function to begin
if __name__ == '__main__':
    main()
```

urllib

a Python module that can be used for fetching URLs.

functions and classes to help with URL actions

- basic authentication,
- redirections,
- cookies, etc

Creating a function

```
def get_weather_json(city):
    """
    Gets the 10 day forecast of a city in Canada.
    And writes it to a CSV file.
    """
    return "data"
```

Creating a function

```
def get_weather_json(city):
    """
    Gets the 10 day forecast of a city in Canada.
    And writes it to a CSV file.
    """
    return "data"
```

```
def main():
  get_weather_json('London')
```

Getting the data

```
def get_weather_json(city):

    response = urlopen('http://api.wunderground.com/api/'
     + WAPI_KEY
     + '/forecast10day/q/canada/'
     + city
     + '.json')

    data = json.load(response)
```

Writing to a CSV

```
with open('%s.csv' % city, 'w') as outfile:
  writer = csv.writer(outfile)
  writer.writerow(["Day", "Low Temp"]) # header row

  for day in data['forecast']['simpleforecast']['forecastday']:
    row = []
    row.append(str(day['date']['weekday']))
    row.append(day['low']['celsius'])
    writer.writerow(row)
```

- `python weatherdata.py`

Pandas

`pandas.pydata.org`

a data analysis library

takes data and creates a DataFrame

- a python object with rows and columns

Seaborn

`seaborn.pydata.org`

high-level interface to Matplotlib

default themes, custom colour palettes

visualizing information from matrices

Plotting our temperature data

create a `plotweather.py`

```
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
```

Plotting our temperature data

create a `plotweather.py`

```
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
```

```
def main():
  # do fancy things here.

if __name__ == '__main__':
    main()
```

Plotting our temperature data

```
temps = pd.read_csv('London.csv') # read data with pandas
```

Plotting our temperature data

```
temps = pd.read_csv('London.csv') # read data with pandas
```

```
# plot with seaborn
sns.pointplot(x = "Day", y = "Low Temp", data = temps)
```

Plotting our temperature data

```
temps = pd.read_csv('London.csv') # read data with pandas
```

```
# plot with seaborn
sns.pointplot(x = "Day", y = "Low Temp", data = temps)
```

```
plt.title("Graph of weekly Low temperature in London, Ontario")
plt.xlabel("Day")
plt.ylabel("Low Temp");
```

Plotting our temperature data

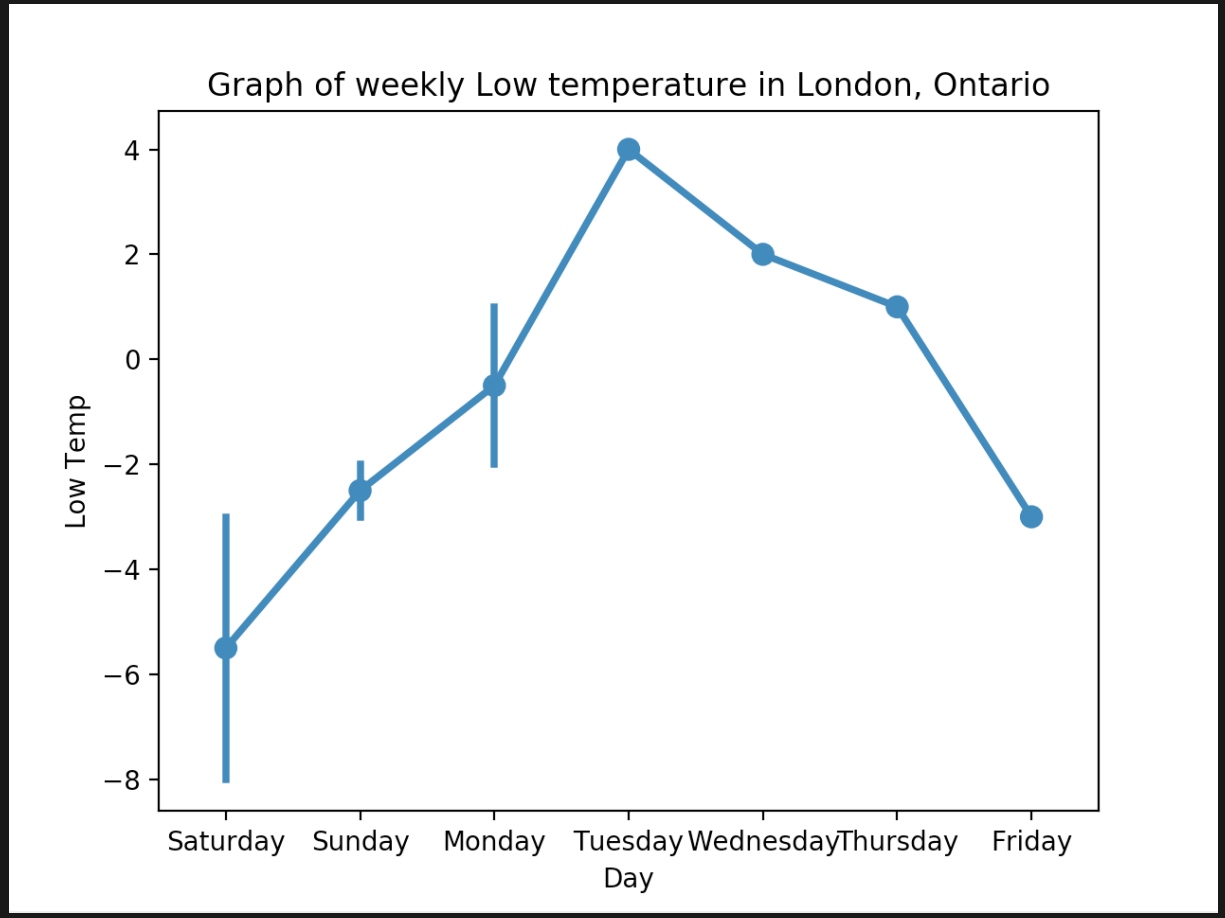
```
temps = pd.read_csv('London.csv') # read data with pandas
```

```
# plot with seaborn
sns.pointplot(x = "Day", y = "Low Temp", data = temps)
```

```
plt.title("Graph of weekly Low temperature in London, Ontario")
plt.xlabel("Day")
plt.ylabel("Low Temp");
```

```
plt.show() # show the plot
```

Our first plot!



Challenge!

Plotting High & Low temperatures for London

Median temps for Feb 2018

Winter Snowfall

Other cool libraries

NumPy

Beautiful Soup

Scikit-Learn

Resources and cool projects

Analyze Taylor Lyrics using Python

Humble Intro to Analysis with Pandas and Seaborn | Kaggle

Flask

Face Recognition