CptS 111 Introduction to Algorithmic Problem Solving (http://piazza.com/wsu/fall2016/cpts111/home)

Washington State University (https://wsu.edu)

Gina Sprint (http://eecs.wsu.edu/~gsprint/)

PA2 Functions (60 pts)

Due Monday, September 26 at Midnight.

Learner Objectives

At the conclusion of this programming assignment, participants should be able to:

- Call functions
- Define functions

Prerequisites

Before starting this programming assignment, participants should be able to:

- Use variables, `print()` statements, and collect input from the user with `input()`
- Convert values from one type to another (type casting)
- Apply basic arithmetic

Acknowledgments

Content used in this assignment is based upon information in the following sources:

- Google Maps API (https://developers.google.com/maps/web-services/overview)
- Michigan State University Computer Science Department's PA11 (http://www.cse.msu.edu/~cse231/PracticeOfComputingUsingPython/08_ClassDesign/GoogleMap/Project11

Overview and Requirements

Write a program (distances.py) that computes distances between pairs of U.S. cities. To do this, we are going to use the Google Maps API (https://developers.google.com/maps/web-services/overview). API stands for application programming interface. An API provides functions for programmers to use, without needing to know the details of the implementation. For example, Google Maps is a complex, well-tested program. We can use Google Maps via its API without knowing Google Map's details. This is also the beauty of functions. As long as we know:

1. What function(s) to call
2. What parameters the function(s) expect
3. What the function(s) return

Then we can use Google Maps!

Program Details

Write a program that prompts the user to enter 7 cities:

1. The user's city of residence
2. 3 *pairs of cities*.

A city is a string of the form: "city, state abbreviation". For example: `New York, NY`

The output of the program has two parts:

1. Displays the *distance in miles* between the each pair of cities (3 pairs).
2. Displays the *distance in miles* between each city (all 6 cities entered by the user: 3 pairs -> 6 cities) and the user's city of residence.

All distances reported should be in miles and with two decimal places.

Starter Code

At this point in the course, we haven't learned the skills necessary to query the Google Maps API and extract the distance from the response. Consequently, I am providing code for you to **copy and paste** into your program that does this for you:

In [1]:

```python
import urllib.request

def format_city_string(city_str):
    '''
    Students: no need to call this function
    To prepare the city string for the query:
    1. remove comma
    2. replace spaces with +
    '''
    city_str = city_str.replace(",", "")
    city_str = city_str.replace(" ", "+")
    return city_str

def build_query(origin, dest):
    '''
    Students: no need to call this function
    Builds the query string for the Google Distance Matrix API according to
 this website:
    https://developers.google.com/maps/documentation/distance-matrix/start
    '''
    query_base = "http://maps.googleapis.com/maps/api/distancematrix/json?
igins="
    query = query_base + origin
    query += "&destinations="
    query += dest
    query += "&mode=driving&sensor=false"
    return query

def extract_distance(results_str):
    '''
    Students: no need to call this function
    Extracts the distance in meters from the JSON response.
    '''
    index = results_str.find("distance")
    results_str = results_str[index:]
    index = results_str.find("value")
    results_str = results_str[index:]
    index = results_str.find(":")
    results_str = results_str[index + 2:]
    index = results_str.find(r"\n")
    results_str = results_str[:index]
    dist = int(results_str)
    return dist

def get_distance(city1, city2):
    '''
    STUDENTS: THIS IS THE FUNCTION YOU WILL CALL
    Accepts 2 strings representing cities in the U.S.
    Returns the integer distance in meters between city1 and city2
    '''
    city1 = format_city_string(city1)
    city2 = format_city_string(city2)

    query = build_query(city1, city2)
    web_obj = urllib.request.urlopen(query)
    # web_obj.read() returns an array of bytes, need to convert to a string
    results_str = str(web_obj.read())
    web_obj.close()

    dist = extract_distance(results_str)
    return dist
```

The only function you need to interact with is `get_distance(<city1 string>, <city2 string>)`. Here is an example of how to use `get_distance()`:

```
In [2]: dist = get_distance("Seattle, WA", "Pullman, WA")
        print("The distance between Seattle, WA and Pullman, WA is %d meters" %
        (dist))

        The distance between Seattle, WA and Pullman, WA is 457923 meters
```

Functions to Define

For this program, define the following functions:

1. `display_instructions()`: Accepts no arguments. Prints instructions describing the use of the program to the user.
2. `get_city_string(pair_num, city_ordering)`: Accepts 2 arguments: `pair_num` an integer representing the pair number [1-3] and `city_ordering` a string representing whether this is the "first" or "second" city of the pair. Prompts the user to enter the city. Use `pair_num` and `city_ordering` to provide the user with information regarding the progress of entering city names. Returns a string representing the city.
3. `meters_to_miles()`: Accepts 1 argument: `meters` an integer representing the distance in meters. Converts `meters` to miles and returns the result as a floating point number.
4. `display_city_distance(city1, city2, distance)`: Accepts 3 arguments: `city1` a string representing the first city of a pair, `city2` a string representing the second city of a pair, and `distance` an integer representing the distance in meters. Calls `meters_to_miles()` to convert `distance` from meters to miles and displays the result.
5. `main()` function that drives the program.

Feel free to define more functions as you see fit!

Example Run

Here is an example run of the program:

```
Welcome to the distance calculator program, utilizing Google Maps!
You will be prompted to enter 3 pairs of cities.

Please enter cities in the form "city, state abbreviation".
For example: New York, NY

The program will tell you the distances between each pair of cities. Enjoy!

Please enter your currenty city of residence:
Pullman, WA
Please enter the first city for pair #1:
Seattle, WA
Please enter the second city for pair #1:
Spokane, WA
Please enter the first city for pair #2:
Pullman, WA
Please enter the second city for pair #2:
Spokane, WA
Please enter the first city for pair #3:
New York, NY
Please enter the second city for pair #3:
Seattle, WA

The distance between Seattle, WA and Spokane, WA is 278.93 miles
The distance between Pullman, WA and Spokane, WA is 74.61 miles
The distance between New York, NY and Seattle, WA is 2831.12 miles

The distance between Seattle, WA and Pullman, WA is 284.54 miles
The distance between Spokane, WA and Pullman, WA is 74.67 miles
The distance between Pullman, WA and Pullman, WA is 0.00 miles
The distance between Spokane, WA and Pullman, WA is 74.67 miles
The distance between New York, NY and Pullman, WA is 2609.58 miles
The distance between Seattle, WA and Pullman, WA is 284.54 miles
```

Bonus (5 pts)

`matplotlib`(http://matplotlib.org/index.html) is a Python library that produces beautiful publication-quality visualizations. To use `matplotlib`, we simply need to import it: `import matplotlib` I've written a function, `plot_city_distances()` that accepts 13 arguments, 7 strings representing the cities entered by the user and 6 distances between the cities and the user's current city of residence:

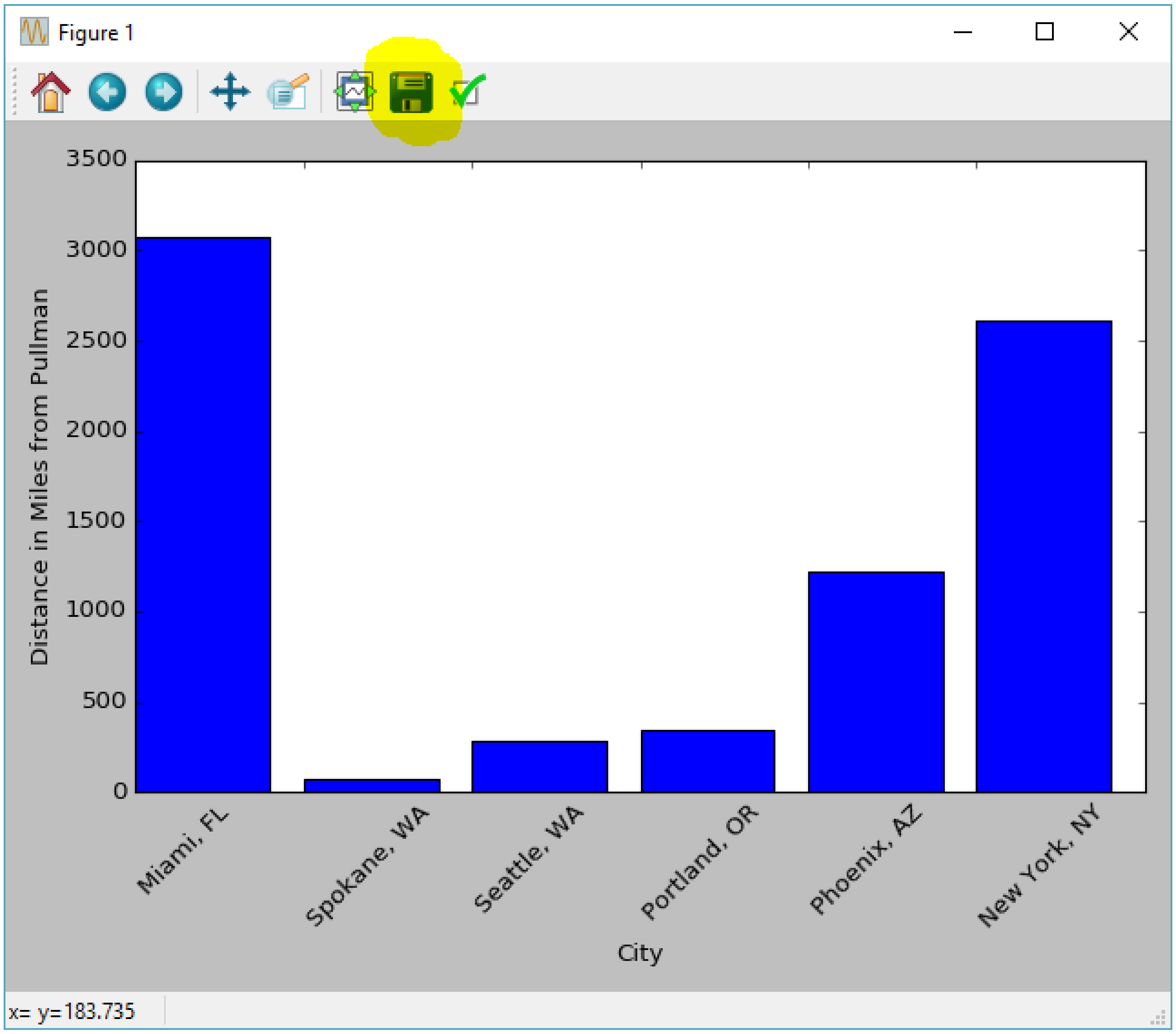
In [1]:
```
import matplotlib.pyplot as plt

def plot_city_distances(current_city, city1, dist1, city2, dist2, city3, dis
t3, city4, dist4, city5, dist5, city6, dist6):
    '''
    111 STUDENTS: THIS IS THE FUNCTION YOU WILL CALL FOR THE **BONUS** TASK
    Accepts 6 strings representing cities in the U.S. and 6 distances in met
ers representing
    the distance between the city and the user's current city of residence
    Ordering of the parameters is 6 pairs of city string, distance value

    Uses matplotlib functions to plot a bar graph of the city distances.
    Save the plot by clicking on the save button on the toolbar of the plot
 window.
    Press the X to close the window when you are done.

    This function does not return anything.
    '''
    x = [0, 1, 2, 3, 4, 5]
    x_labels = [city1, city2, city3, city4, city5, city6]
    y = [meters_to_miles(dist1), meters_to_miles(dist2), meters_to_miles(dis
t3), meters_to_miles(dist4), \
         meters_to_miles(dist5), meters_to_miles(dist6)]
    plt.bar(x, y)
    plt.xticks(x, x_labels, rotation=45, ha='left')
    plt.xlabel("City")
    plt.ylabel("Distance in Miles from %s" %(current_city))
    plt.tight_layout()
    # show the window
    plt.show()
```

Copy and paste the above code into your distances.py files. Call `plot_city_distances()` passing in the city strings and distances to the user's current city of residence in meters. The function will show a window that looks like the following:



Save the plot as a .png image file by clicking on the save icon in the toolbar (highlighted in the screen shot above). Turn in your saved city distances plot image file in with your distances.py file in your zip file.

Submitting Assignments

1. Use the Blackboard tool https://learn.wsu.edu (https://learn.wsu.edu) to submit your assignment to your TA. You will submit your code to the corresponding programming assignment under the "Content" tab. You must upload your solutions as `<your last name>_pa2.zip` by the due date and time.
2. Your .zip file should contain your .py file and your .png file *if you attempted the bonus task*.