Project 5 – Earthquakes

CPE 101

Updates

- None so far!

Credit

This project adapted from an iOS programming lab developed by Dr. John Bellardo. Many thanks!

Objectives

- More practice working with lists
- More practice searching with lists
- Introduction to file input and output in Python
- Introduction to objects in Python
- Introduction to sorting in Python
- Introduction to dictionaries in Python
- Exposure to JSON data and reading data from a webpage

Programming environment. This is a solo programming project. You are responsible for all the work related to the program development, testing, and submission.

Given Files

You will need various input and output files to test your program. These files are available to copy from the command line by typing:

```
cp ~graderjw/www/101/Projects/p5/files/* .
```

Program Description

In this project, you will implement a program to store and display real earthquake data. Your program will begin by reading earthquake data from a file and displaying the data in a nicely formatted table. Then you will give your user the option to sort the earthquakes in different ways and re-display the data. As a last step you will implement a feature to read new earthquake data from the usgs.gov website to update your program with live earthquake data!

A sample run of the program is shown here. User input is in bold for clarity:

```
Earthquakes:
------------
(1.05)             9km E of Running Springs, CA at 2017-02-27 00:51:05 (-117.008, 34.204)
(2.30)             16km SSW of Big Lake, Alaska at 2017-02-27 18:23:02 (-150.074, 61.381)
```

```
(2.22)          28km SW of Rio Dell, California at 2017-02-27 00:31:18 (-124.335, 40.307)
(2.19)              5km S of Gilroy, California at 2017-02-26 21:32:18 (-121.580, 36.958)
(2.06)                6km SW of Volcano, Hawaii at 2017-02-27 10:55:59 (-155.280, 19.383)
(4.50)        122km SSE of Chignik Lake, Alaska at 2017-02-27 08:07:28 (-157.822, 55.289)
(1.00)              4km NNW of Lake Henshaw, CA at 2017-02-27 07:43:04 (-116.775, 33.281)
(2.58)          62km WSW of Ferndale, California at 2017-02-27 19:04:52 (-124.934, 40.349)
(0.98)          2km N of The Geysers, California at 2017-02-27 01:03:53 (-122.757, 38.797)
(2.80)                12km W of Harper, Kansas at 2017-02-27 00:36:40 ( -98.161, 37.287)
(2.67)               25km SSE of Waimea, Hawaii at 2017-02-27 18:34:27 (-155.544, 19.832)

Options:
  (s)ort
  (f)ilter
  (n)ew quakes
  (q)uit

Choice: s
Sort by (m)agnitude, (t)ime, (l)ongitude, or l(a)titude? m

Earthquakes:
------------
(4.50)        122km SSE of Chignik Lake, Alaska at 2017-02-27 08:07:28 (-157.822, 55.289)
(2.80)                12km W of Harper, Kansas at 2017-02-27 00:36:40 ( -98.161, 37.287)
(2.67)               25km SSE of Waimea, Hawaii at 2017-02-27 18:34:27 (-155.544, 19.832)
(2.58)          62km WSW of Ferndale, California at 2017-02-27 19:04:52 (-124.934, 40.349)
(2.30)             16km SSW of Big Lake, Alaska at 2017-02-27 18:23:02 (-150.074, 61.381)
(2.22)          28km SW of Rio Dell, California at 2017-02-27 00:31:18 (-124.335, 40.307)
(2.19)              5km S of Gilroy, California at 2017-02-26 21:32:18 (-121.580, 36.958)
(2.06)                6km SW of Volcano, Hawaii at 2017-02-27 10:55:59 (-155.280, 19.383)
(1.05)             9km E of Running Springs, CA at 2017-02-27 00:51:05 (-117.008, 34.204)
(1.00)              4km NNW of Lake Henshaw, CA at 2017-02-27 07:43:04 (-116.775, 33.281)
(0.98)          2km N of The Geysers, California at 2017-02-27 01:03:53 (-122.757, 38.797)

Options:
  (s)ort
  (f)ilter
  (n)ew quakes
  (q)uit

Choice: f
Filter by (m)agnitude or (p)lace? p
Search for what string? ca

Earthquakes:
------------
(2.58)          62km WSW of Ferndale, California at 2017-02-27 19:04:52 (-124.934, 40.349)
(2.22)          28km SW of Rio Dell, California at 2017-02-27 00:31:18 (-124.335, 40.307)
(2.19)              5km S of Gilroy, California at 2017-02-26 21:32:18 (-121.580, 36.958)
(2.06)                6km SW of Volcano, Hawaii at 2017-02-27 10:55:59 (-155.280, 19.383)
(1.05)             9km E of Running Springs, CA at 2017-02-27 00:51:05 (-117.008, 34.204)
(1.00)              4km NNW of Lake Henshaw, CA at 2017-02-27 07:43:04 (-116.775, 33.281)
(0.98)          2km N of The Geysers, California at 2017-02-27 01:03:53 (-122.757, 38.797)

Options:
  (s)ort
  (f)ilter
  (n)ew quakes
  (q)uit

Choice: n

New quakes found!!!

Earthquakes:
------------
(4.50)        122km SSE of Chignik Lake, Alaska at 2017-02-27 08:07:28 (-157.822, 55.289)
(2.80)                12km W of Harper, Kansas at 2017-02-27 00:36:40 ( -98.161, 37.287)
(2.67)               25km SSE of Waimea, Hawaii at 2017-02-27 18:34:27 (-155.544, 19.832)
(2.58)          62km WSW of Ferndale, California at 2017-02-27 19:04:52 (-124.934, 40.349)
(2.30)             16km SSW of Big Lake, Alaska at 2017-02-27 18:23:02 (-150.074, 61.381)
(2.22)          28km SW of Rio Dell, California at 2017-02-27 00:31:18 (-124.335, 40.307)
```

```
(2.19)              5km S of Gilroy, California at 2017-02-26 21:32:18 (-121.580, 36.958)
(2.06)                6km SW of Volcano, Hawaii at 2017-02-27 10:55:59 (-155.280, 19.383)
(1.05)             9km E of Running Springs, CA at 2017-02-27 00:51:05 (-117.008, 34.204)
(1.00)              4km NNW of Lake Henshaw, CA at 2017-02-27 07:43:04 (-116.775, 33.281)
(0.98)         2km N of The Geysers, California at 2017-02-27 01:03:53 (-122.757, 38.797)
(2.30)               23km NW of Nikiski, Alaska at 2017-02-27 19:11:38 (-151.592, 60.840)
(1.09)         24km WSW of Coalinga, California at 2017-02-27 19:09:56 (-120.603, 36.034)


Options:
  (s)ort
  (f)ilter
  (n)ew quakes
  (q)uit

Choice: f
Filter by (m)agnitude or (p)lace? m
Lower bound: 2.5
Upper bound: 10

Earthquakes:
------------
(4.50)        122km SSE of Chignik Lake, Alaska at 2017-02-27 08:07:28 (-157.822, 55.289)
(2.80)                 12km W of Harper, Kansas at 2017-02-27 00:36:40 ( -98.161, 37.287)
(2.67)              25km SSE of Waimea, Hawaii at 2017-02-27 18:34:27 (-155.544, 19.832)
(2.58)          62km WSW of Ferndale, California at 2017-02-27 19:04:52 (-124.934, 40.349)

Options:
  (s)ort
  (f)ilter
  (n)ew quakes
  (q)uit

Choice: q
```

General Notes

You should have a total of three files in your submission:

- `quakeFuncs.py` This file must include the actual functions definitions/implementation, as well as the definition of your `Earthquake` structure (see below). You are given this file to start with and must add to it. (Do not change existing code.)
- `funcsTests.py` This file must include unit test functions, which contain the assert statements testing the functions developed in `quakeFuncs.py`. You are given this file to start with and must add to it. (Do not change the existing tests.)
- `quakes.py` This file must include the main function, which calls all the functions developed within `quakeFuncs.py`.

Program Specification

Earthquake Object:

- You must store each line of data in an object whose type is a class called `Earthquake`. The `Earthquake` class should have the following attributes: `place` (a string), `mag` (a float), `longitude` (a float), `latitude` (a float), and `time` (an int). Put the definition for this class in your `quakeFuncs.py` module.

- Add an `__eq__` function to your `Earthquake` class so that you can test two `Earthquake` objects for equality. Two earthquakes are equal if all of their attributes are equivalent. *You need to add this function in order to pass my test cases.*

I give you tests in funcsTest.py to test creating earthquake objects and to test them for equality.

Input File:

Your program will read initial earthquake data from a file. You are required to write a function `read_quakes_from_file(filename)` function that takes a string filename as input and returns a list of `Earthquake` objects as output. Make sure this function works with any filename you give it. However, in your `main` you will hardcode the filename `"quakes.txt"` as the input file to your program. An initial `"quakes.txt"` file is given to you in the following format:

```
2.19 -121.5801697 36.9580002 1488173538 5km S of Gilroy, California
1.24 -117.4906667 33.9131667 1488179250 5km NE of Home Gardens, CA
0.95 -122.8063354 38.8243332 1488178712 6km NW of The Geysers, California
```

The magnitude, longitude, and latitude for each earthquake are given first. Read these as floats. The time of the earthquake (in seconds since the Unix Epoch - January 1, 1970) comes next. Read this as an integer. The rest of the line is the place of the earthquake. Store this as a string.

Hint: Recall the `split()` and `join()` methods of strings.

I give you a test in funcsTests.py to test reading earthquakes in from a file.

Your function may assume any file given to it exists and contains valid earthquake data. I will not test with non-existent or invalid files.

Output:

Your program should format and output all the earthquake data to the screen in a table. (See above for sample output.) The magnitude should be printed to 2 decimal places, the "place" of the earthquake in a column of 40 spaces, then the date and time of the earthquake, followed by the longitude and latitude printed to 3 decimal places each.

I *give you a function* to convert from an integer time to a formatted string displaying the time in local time. Use this function to display the time in your table. The function is called `time_to_str` and is located in the given `quakeFuncs.py` module.

I suggest writing a function to display the earthquake data.

Program Options:

After displaying the earthquake data to the user, display a list of program options. (See above for formatting.)

- Should the user select any of the sorts, sort the data according to the format chosen and re-display the data. If the user choses magnitude or time, the earthquakes should be sorted in descending order. If the user choses longitude or latitude, the earthquakes should be sorted in ascending order.

- If the user choses to filter the quakes, get additional information regarding how the quakes should be filtered (see sample above). Display the filtered list of quakes to the user, but *do not* alter the original list of quakes.

    - Do not alter the order of the quakes. For example, if the original list of quakes is sorted by time, then the filtered list should also be ordered by time. Note that this shouldn't be any extra work on your behalf. Just don't reorder the quakes.

    - You must write and use two additional functions to help with this task:

      `filter_by_mag(quakes, low, high)` takes a list of quakes, a low value, and a high value. The function returns a new list of quakes consisting of all the quakes with a magnitude between the low and high values (inclusive). Note that this function does no printing.

      `filter_by_place(quakes, word)` takes a list of quakes and a string to search for as input. The function returns a new list of quakes consisting of all the quakes with the search word contained in the place string. The search should be case insensitive. Note that this function also does no printing.

      I provide sample test cases for both functions in the test file. Provide additional tests.

- If the user choses to find new earthquakes, you must parse JSON data from the usgs.gov website to add the latest earthquakes to your program (see below for more details). This feature is worth the final 20 points of your program.

- If the user choses to quit, then write the earthquake data back out to the "quakes.txt" file to assure the earthquakes are written in the order they were last sorted, and to assure that any new earthquake data is saved. I suggest writing a function to do this. Note that you must write the data to the file in the same *format* that it was read in.

Additional menu details:

- Your menu must work regardless of whether the user enters upper or lowercase characters. For example, if the user presses 'M', then your program should sort the earthquakes by magnitude.
- Your program will not be tested with invalid input.

Reading New Quakes

Should the user chose to find new earthquake data, your program needs to read JSON (JavaScript Object Notation) data from the usgs.gov website. I provide a function for you to get and return this data as a Python dictionary, `get_json(url)`. See the last page of this document for a sample of what the data looks like. Call my function with the following url string:

```
'http://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/1.0_hour.geojson'
```

Then take the returned dictionary and get the "features" array from it.

For each feature in the features array, create a new `Earthquake` from the feature. You must write and use a function `quake_from_feature(feature)` in your solution. The function takes a feature dictionary as input and returns an `Earthquake` object as output. Simply read each piece of data from the appropriate place in the given feature dictionary and pass it to appropriate part of the `Earthquake` constructor. For example, to read the "place" of the earthquake you would access the `feature["properties"]["place"]`.

Important, the time in the JSON data is stored as integer milliseconds since the epoch. You must convert from milliseconds to seconds when you construct your new earthquake object.

Finally, once you have a new `Earthquake` constructed and returned from your `quake_from_feature` function, check to see if it is already *in* your list of earthquakes. If not, append the new quake to the list. *Hint*: If your `__eq__` method is written properly, you may use the `in` operator. Additionally, keep track of whether or not your program found any new quakes.

When finished adding new quakes to the list, print "New quakes found!!!" if one or more quakes were added to the list. Then (regardless of whether or not new quakes were found) print the list of quakes and menu options again.

I provide a test in funcsTests.py for your `quake_from_feature()` function.

Unit Testing – For Your Functions

You are given a sample `funcsTests.py` file to work with. Make sure your code passes all these tests and add more of your own. To run the file, make sure `test0.txt` is in the same directory as your `funcsTests.py` file. Then type:

```
python3 < funcsTests.py
```

I suggest you write more tests for these required functions and add tests for any additional functions you write. However, you are not required to do so. Know that I will unit test your required functions with many more tests than those provided.

Diff Testing – For Your Main

Your "main" code is required to produce the same output as mine. Like previous projects, you can check by diffing your output with mine given certain user inputs. The difficulty is that your code both reads from a file (quakes.txt) and gets user input. Additionally, it has output to the

screen and writes output to the quakes.txt file. So, to `diff` test your code with mine, follow these steps carefully. First, copy all the needed diff testing files:

```
cp ~graderjw/www/101/Projects/p5/difftests/* .
```

Each test case has four files associated with it. For example, Test 0 has: `quakes0.txt`, `quakesFinal0.txt`, `in0.txt`, `out0.txt`

The following steps are for doing Test 0. Replace the 0 in each file with a 1 for Test1, etc.

1. Copy `quakes0.txt` to `quakes.txt` so that your program uses the quakes inside as the initial list of quakes.

   ```
   cp quakes0.txt quakes.txt
   ```

2. Run your program with the user input provided in `in0.txt` and save the output to `my_out0.txt`

   ```
   python3 quakes.py < in0.txt > my_out0.txt
   ```

3. Diff your `my_out0.txt` file with the instructor `out0.txt` file.

   ```
   diff -wB my_out0.txt out0.txt
   ```

4. Diff the `quakes.txt` file (which should have been altered by your program) with `quakesFinal0.txt`

   ```
   diff -wB quakes.txt quakesFinal0.txt
   ```

As always, if you find differences (or your program seems to be behaving wildly), I suggest looking at the user input file (`in0.txt`) and running your code by hand to see what it does when you type in that input.

IMPORTANT: If you want to re-run a test, you must start back at Step 1 and re-copy the quakes0.txt file over quakes.txt so you start with a fresh copy of initial quakes.

IMPORTANT 2: Please be careful copying the various `quakes.txt` files around so that you do not accidentally overwrite your **`quakes.py`** file!!! I regret giving them similar names, but it's a little late to change them now.

Note that none of the diff tests test the New Quakes feature of your program. The diff tests will help assure you that you are reading, sorting, filtering, displaying, and saving properly. If you confirm that your program is able to get new quakes from the website, then you are probably good to go!

Automatic Diff Testing

Automatic test script to run all of the procedures shown in the previous section is now available on UNIX. Use the following command to copy it over to your working directory:

```
cp ~graderjw/www/101/Projects/p5/test_all.sh .
```

Once copied over, you can run all diff tests simply by running the script with some arguments:

```
test_all.sh 0 2
```

Above command will run all instructor tests (0 through 2). You can change the two integer arguments to run different range of tests (two numbers are inclusive upper and lower bounds). In addition, you can also supply up to two diff options as you see fit. For example:

```
test_all.sh 1 2 -y --suppress-common-lines
```

The command above will run instructor diff tests 1 and 2, and show any differences from the diff output side-by-side while hiding the lines with no difference between your output and the instructor's.

Handin

```
handin graderjw 101project05 quakeFuncs.py funcsTests.py quakes.py
```

Sample JSON Earthquake Data


```
{
        "type": "FeatureCollection",
        "metadata": {
                "generated": 1488185761000,
                "url":
"http://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/1.0_hour.geojson",
                "title": "USGS Magnitude 1.0+ Earthquakes, Past Hour",
                "status": 200,
                "api": "1.5.4",
                "count": 3
        },
        "features": [
                {
                        "type": "Feature",
                        "properties": {
                                "mag": 1.05,
                                "place": "9km E of Running Springs, CA",
                                "time": 1488185465040,
                                "updated": 1488185688121,
                                "tz": -480,
                                "url":
"http://earthquake.usgs.gov/earthquakes/eventpage/ci37814024",
                                "detail":
"http://earthquake.usgs.gov/earthquakes/feed/v1.0/detail/ci37814024.geojson",
                                "felt": null,
                                "cdi": null,
                                "mmi": null,
```

```
                        "alert": null,
                        "status": "automatic",
                        "tsunami": 0,
                        "sig": 17,
                        "net": "ci",
                        "code": "37814024",
                        "ids": ",ci37814024,",
                        "sources": ",ci,",
                        "types": ",geoserve,nearby-cities,origin,phase-data,scitech-link,",
                        "nst": 27,
                        "dmin": 0.09331,
                        "rms": 0.23,
                        "gap": 83,
                        "magType": "ml",
                        "type": "earthquake",
                        "title": "M 1.1 - 9km E of Running Springs, CA"
                },
                "geometry": {
                        "type": "Point",
                        "coordinates": [
                                -117.0085,
                                34.2036667,
                                6.79
                        ]
                },
                "id": "ci37814024"
        },
        {
                "type": "Feature",
                "properties": {
                        "mag": 2.22,
                        "place": "28km SW of Rio Dell, California",
                        "time": 1488184278000,
                        "updated": 1488184563372,
                        "tz": -480,
                        "url":
"http://earthquake.usgs.gov/earthquakes/eventpage/nc72768481",
                        "detail":
"http://earthquake.usgs.gov/earthquakes/feed/v1.0/detail/nc72768481.geojson",
                        "felt": null,
                        "cdi": null,
                        "mmi": null,
                        "alert": null,
                        "status": "automatic",
                        "tsunami": 0,
                        "sig": 76,
                        "net": "nc",
                        "code": "72768481",
                        "ids": ",nc72768481,",
                        "sources": ",nc,",
                        "types": ",geoserve,nearby-cities,origin,phase-data,scitech-link,",
                        "nst": 5,
                        "dmin": 0.04125,
                        "rms": 0.01,
                        "gap": 283,
                        "magType": "md",
                        "type": "earthquake",
                        "title": "M 2.2 - 28km SW of Rio Dell, California"
                },
                "geometry": {
                        "type": "Point",
                        "coordinates": [
                                -124.3346634,
                                40.3071671,
                                7.13
                        ]
                },
                "id": "nc72768481"
        },
        {
                "type": "Feature",
```

```
                        "properties": {
                                "mag": 1.6,
                                "place": "36km W of Kenai, Alaska",
                                "time": 1488182473565,
                                "updated": 1488183500820,
                                "tz": -540,
                                "url":
"http://earthquake.usgs.gov/earthquakes/eventpage/ak15412705",
                                "detail":
"http://earthquake.usgs.gov/earthquakes/feed/v1.0/detail/ak15412705.geojson",
                                "felt": null,
                                "cdi": null,
                                "mmi": null,
                                "alert": null,
                                "status": "automatic",
                                "tsunami": 0,
                                "sig": 39,
                                "net": "ak",
                                "code": "15412705",
                                "ids": ",ak15412705,",
                                "sources": ",ak,",
                                "types": ",geoserve,origin,",
                                "nst": null,
                                "dmin": null,
                                "rms": 0.52,
                                "gap": null,
                                "magType": "ml",
                                "type": "earthquake",
                                "title": "M 1.6 - 36km W of Kenai, Alaska"
                        },
                        "geometry": {
                                "type": "Point",
                                "coordinates": [
                                        -151.923,
                                        60.5679,
                                        64.7
                                ]
                        },
                        "id": "ak15412705"
                }
        ],
        "bbox": [
                -151.923,
                34.2036667,
                6.79,
                -117.0085,
                60.5679,
                64.7
        ]
}
```