CS 359 – Programming Paradigms

Python Homework

Due: Taps, Lesson M16, Tuesday, 25 September

Help Policy:

>**AUTHORIZED RESOURCES:** Any, except another cadet's program.
>**NOTE:**
>- Never copy another person's work and submit it as your own.
>- Do not jointly create a program unless explicitly allowed.
>- You must document all help received from sources other than your instructor or instructor-provided course materials (including your textbook).
>- **DFCS will recommend a course grade of F for any cadet who egregiously violates this Help Policy or contributes to a violation by others.**

Documentation Policy:

- You must document all help received from any source other than your instructor.
- The documentation statement must explicitly describe <u>WHAT assistance was provided</u>, <u>WHERE on the assignment the assistance was provided</u>, and <u>WHO provided the assistance</u>.
- If no help was received on this assignment, the documentation statement must state "NONE."
- If you checked answers with anyone, you must document with whom on which problems. You must document whether or not you made any changes, and if you did make changes you must document the problems you changed and the reasons why.
- **Vague documentation statements must be corrected before the assignment will be graded and will result in a 5% deduction on the assignment.**

Turn-in Policies:

- On-time turn-in is at taps on the M-day of the lesson indicated on the course schedule.
- Late penalties accrue at a rate of 25% per 24-hour period past the on-time turn-in date and time. The late penalty is a cap on the maximum grade that may be awarded for late work.
- There is no early turn-in bonus or extra credit for this assignment.

---

1. OBJECTIVES

To complete this homework you will use Python to

- obtain web resources in XML format
- parse a Document Object Model
- read and write data files
- interface with the operating system

2. BACKGROUND

Your task for this assignment is to utilize the Python programming language as a bridge between multiple programs and technologies. This is sometimes referred to as *glueware*.

More specifically, you will use Python to retrieve location and weather information from the Internet for a given zip code, create a custom KML file suitable for rendering by Google Earth, and finally launch Google Earth to display the placemark contained in the KML file.

3. EXERCISE

Your application must have a function named **weather** that takes a zip code as its parameter. When testing, I will expect to be able to run your code by entering the following at the Python prompt:

```
>>> weather( '80840' ) # Quotes necessary for zip codes such as 01507
>>>
```

This function does not return a value, but will launch Google Earth and display a placemark at the given zip code with a balloon containing the current weather forecast.

Your program should follow this general outline:

1. **Obtain city name, latitude, and longitude information for a given zip code.**

   Yahoo! provides a location API that returns an XML file when given a zip code. To obtain location information about USAFA, the following URL would be used:

   > http://where.yahooapis.com/geocode?q=80840

   Note: The raw XML returned from this query can be displayed directly in Internet Explorer or Google Chrome by typing the URL into the address bar of your web browser and pressing enter.

2. **Obtain weather forecast information for a given zip code.**

   Yahoo! also provides a weather API that returns an XML file when given a zip code. To obtain weather information about USAFA, the following URL would be used:

   > http://weather.yahooapis.com/forecastrss?q=80840

   Note: The XML returned from this query contains information about an rss feed that is used to display the data. Thus, typing the query into the address bar of a web browser will not display the raw XML. In fact, Google Chrome will prompt you to select a service to use to view the data. You may do this if you wish, but I suggest you use Internet Explorer to view the raw XML.

   To see the raw XML, type the URL into the address bar of Internet Explorer, press enter, right-click on the page that is displayed, and select View Source. This will display the raw XML by launching Notepad or whatever other text editor you may have installed on your system.

3. **Create a custom KML file with an appropriate Placemark tag**

KML is an XML –based language used to display geographic data in an Earth browser such as Google Earth or Google Maps. For this project, you will need a relatively simple KML file with a Placemark tag in it. In fact, the first example of a Placemark on the [KML Tutorial page](#) is very close to what you will need.

To create this custom KML file, your program will need to parse the node values for city name, latitude, and longitude from the XML containing location information (obtained in step 1) and also parse the day, text description, high temperature, and low temperature attributes from the XML containing the weather forecast (obtained in step 2).

4. **Render the KML Placemark in Google Earth**

Finally, your program will interact with the operating system to launch Google Earth and display the contents of the KML file you have created. Before doing this, you will obviously need to [download Google Earth](#) (requires agreeing to the terms of service).

Google Earth will likely install to a path similar to this:
```
C:\Program Files (x86)\Google\Google Earth\client\googleearth.exe
```

Thus, typing this command at a command prompt would launch Google Earth and have it immediately load the contents of `usafa.kml` file:
```
C:\Program Files (x86)\Google\Google Earth\client\googleearth.exe usafa.kml
```

4. HELPFUL HINTS

- Refer the [currency.py](#) example from class frequently – it contains examples of all necessary tasks.
- Latitude and longitude are reversed in the coordinates tag of a Placemark. Latitude and longitude are normally expressed in that order, but the **<coordinates>** tag of a KML Placemark reverses them so longitude comes before latitude. Thus, the following would be used for the coordinates of USAFA:
  ```
  <coordinates>-104.850226,38.989671</coordinates>
  ```
- Python uses escape characters within strings in the same manner as Java and C. Thus, the path to the Google Earth executable shown above would need to be typed in one of the following two ways:
  ```
  'C:/Program Files (x86)/Google/Google Earth/client/googleearth.exe'
  ```
  or
  ```
  'C:\\Program Files (x86)\\Google\\Google Earth\\client\\googleearth.exe'
  ```
- Sadly, balloons in Google Earth cannot be made to open by default … the user must click on the pin.

5. SUBMISSION REQUIREMENTS

- The file name of your submission must be **Lastname.py** (using your own last name, of course).
- Your name and documentation statement must be included as a comment at the top of your file.
- Use the course website to submit your single file, **Lastname.py**. **Do not zip your source file!**

CS 359 – Python HW – Grade Sheet   Name:_____

Criteria	Points Earned	Points Available
Obtains location information from where.yahooapis.com.		**5**
Parses the XML location information to retrieve necessary data		**5**
Obtains weather information from weather.yahooapis.com		**5**
Parses the XML weather information to retrieve necessary data		**5**
Creates the custom KML file with all necessary Placemark data		**5**
Launches Google Earth and displays the Placemark		**5**
Overall quality of software design		**10**
Subtotal:		**40**

Adjustments			
	All code meets specified standards:		**− 4**
	Vague/Missing Documentation:		**− 2**
Adjustments	**Submission Requirements Not Followed:**		**− 2**
	Late Penalties:		**25/50/75%**
	Total w/adjustments:		

Comments from Instructor: