Coursework
Stage 4
Task 3

Luka Triska

May 26, 2017

Description

1 Purpose

The purpose of the data that I have collected is to be able to see the dynamic of change between:

1. Desired routes that user inputs
2. The *weather* on those routes
3. The particular *time of year* (and day) when it is the most safe and the most dangerous to travel

I have successfully used this programme for estimating the Ukrainian road and weather conditions.

2 Input

If user runs `data_collecting.py` or `weather_ADT_test.py` he will be prompted to enter the start of his journey and the end of his journey. Then he has to enter departure year, month, day, hour and minute.

3 Output

3.1 `data_collecting.py`

Module outputs into txt files only the most popular summary for each month, one day of the month (15th) and one certain hour.

3.2 `weather_ADT_test.py`

Module prints out a number of segments (depending on the route length), and in each segment there is the following information:

- **Temperature** - temperature
- **Time** - time when that temperature occurs
- **From, To** - Where the segment begins and where it ends
- **Distance** - How long the segment is
- **Duration** - How much time it will supposedly take you
- **Summary** - Weather summary

4 Programme structure

The programme consists of five modules:

1. `weather_ADT.py`
   It contains the WeatherAPI ADT (python class), which has the following methods:
   - `__init__()` - accepts address, time, and other markers, based on those it creates a weather forecast URL
   - `address()` - returns the entered address
   - `time()` - returns the entered time
   - `weather_url()` - returns the data from the URL, using `get_data_from_url()` imported from `secondary_functions.py` (look below)
2. `secondary_functions.py`
   As you may have noticed, this module contains help functions to the above mentioned module, such as:
   - `create_destination_url()` - creates the destination URL for the maps part, used in the programme
   - `get_data_from_url()` - loads the JSON file from URL into a dictionary
   - `address_to_coors() and coors_to_address()` - convert geographical coordinates to normal address and otherwise (this is where the Google Maps API is used)
3. `weather_ADT_test.py`
   A test function for `weather_ADT.py` - input requirements listed here, output - here.
4. `data_collecting.py`
   Collects the data, using this input, and gives this output
5. `data_processing.py`
   Processes the output from `data_collecting.py`, writing to a .txt file what summaries are the most popular for certain months.

5 Manual

To collect data run `data_collecting.py` and to test the weather API run `weather_ADT_test.py`.

6 Test examples description

To see it go here.