CSSE1001/7030

Semester 2, 2014
Assignment 1
10 marks

Due Thursday 4 September, 2014, 9:30am

To PV or not to PV

1 Introduction

The University of Queensland has several arrays of photovoltaic panels on roofs of buildings over several campuses. Data on power output, sunlight, temperature and other weather information is being collected on a minute-by-minute basis. This data is available for everyone to use. You may have noticed, for example, that one of the screens on the wall on level 2 outside the lecture theatre cycles through various information and this includes the PV data.

This assignment is about processing this data. To prevent overloading the server that provides this information we copy this information to our server. The support file for the assignment provides access to this copied data in such a way as to reduce the chances of students accidentally overloading our server - for example by going into an infinite loop.

The support file provides a function `get_data_for_date` which takes a date string of the form `'DD-MM-YYYY'` (for example `'03-08-2014'`) and returns essentially the text of a CSV file. Each line represents the data for a given minute of the day. Below are two consecutive lines of the data on `'03-08-2014'` at times `14:00` and `14:01`.

```
14:00,18.7,383.5,266405,5480,212500,183750,52380,6804,57150,17431,65567
14:01,18.7,383.5,226430,6600,210850,206700,51870,11868,69850,18486,59222
```

The first entry is the time of day, the second entry is the temperature in Celsius, the third entry is the amount of sunlight in W/m^2 and the remaining entries are the power outputs in Watts produced by the different arrays (in the order of the entries of the `ARRAYS` list in the support file).

For this assignment you will write a program to carry out some simple calculations on the data and display the results. With your text-based interactive program you will be able to enter a date of interest and have data displayed as in the example interaction below.

```
Welcome to PV calculator


Command: date 03-08-2014

Statistics for 03-08-2014

Maximum Temperature: 19.8C at times 14:04, 14:05, 14:06

Total Energy Production: 4218.2kWh

Maximum Power Outputs:

UQ Centre, St Lucia                 351.2kW
Concentrating Array                   7.0kW
Multi Level Car Park #1             275.9kW
Multi Level Car Park #2             269.1kW
Sir Llew Edwards Bld.                64.2kW
Prentice Building                    13.6kW
Advanced Engineering Bld.            86.2kW
Learning Innovation Bld.             26.9kW
Global Change Institute              95.5kW

Command: date
Unknown command: date

Command: date 02-08-2014 03-08-2014
Unknown command: date 02-08-2014 03-08-2014

Command: xxx
Unknown command: xxx

Command: date 04-08-2014
```

```
Statistics for 04-08-2014

Maximum Temperature: 19.0C at times 13:52, 13:53

Total Energy Production: 3393.3kWh

Maximum Power Outputs:

UQ Centre, St Lucia                338.1kW
Concentrating Array                  6.4kW
Multi Level Car Park #1            288.6kW
Multi Level Car Park #2            311.9kW
Sir Llew Edwards Bld.               67.7kW
Prentice Building                   13.0kW
Advanced Engineering Bld.           94.2kW
Learning Innovation Bld.            28.0kW
Global Change Institute            102.8kW

Command: q
```

The response to the first command given (`date 04-08-2014`) is to display a collection of statistics for that date. The second, third and fourth examples show the response to an illegal command. The `q` command causes the program to terminate. The only legal commands are `date` followed by a date (we assume for this assignment that the date is in the correct format), and `q` to quit the program.

2 Assignment Tasks

For each function that you write you need to provide a suitable comment giving a description, the type and any preconditions. You should use the triple-quote commenting style. CSSE7030 students are required to add extra features as described at the end of this section.

2.1 Download files

The first task is to download `assign1.py` and `assign1_support.py`.
Note: When you download these files your name and student number will be automatically added in the comments at the beginning of `assign1.py` and a key will be added to `assign1_support.py`.

We suggest you create a folder in which to write your solution and put these files in that folder.

The file `assign1.py` is for your assignment. **Do not modify the support file or your assignment file outside the area provided for you to write your code**. When you have completed your assignment you will submit the file `assign1.py` containing your solution to the assignment (see Section 4 for submission details).

2.2 Write the code

Finally, write your solution to the assignment making sure you have included suitable comments. There are several functions you need to write and these are described below. **Do not use global variables in your code.** We are not expecting you to deal with exceptions for this assignment and so we will not be testing your code with things like invalid date strings or for dates that have no information about the PV arrays (such as future dates or very old dates).

2.2.1 Load PV Data

`load_data(dateStr)` takes a string representing a date in the correct format and returns a list of data for each minute of the day in time order. This function gets the data from the server (using the appropriate support function) as a string in CSV format and converts this to the required data structure. In the example below we show just the two time entries as in the example of the text version. **Note:** When you test this in the interpreter it will print the result on one (very long) line. The example below has been formatted so it will fit on the page. Each element of the list is a 4-tuple consisting of the time, temperature, sunlight and a tuple of the power output for each array.

```
>>> load_data('03-08-2014')
[
.....
('14:00', 18.7, 383.5,
    (266405, 5480, 212500, 183750, 52380, 6804, 57150, 17431, 65567)),
('14:01', 18.7, 383.5,
    (226430, 6600, 210850, 206700, 51870, 11868, 69850, 18486, 59222)),
.....
]
```

2.2.2 Maximum Temperature

`max_temperature(data)` takes data as produced by `load_data` and returns a pair consisting of the maximum temperature and the list of all the times at which the temperature was maximum. For example:

```
>>> data = load_data('03-08-2014')
>>> max_temperature(data)
(19.8, ['14:04', '14:05', '14:06'])
>>>
```

2.2.3 Total Energy

`total_energy(data)` takes data as produced by `load_data` and returns the total power produced in kilowatt hours (kWh) of all the arrays over the entire day. For example:

```
>>> data = load_data('03-08-2014')
>>> total_energy(data)
4218.23895
>>>
```

2.2.4 Maximum Power

`max_power(data)` takes data as produced by `load_data` and returns the list of pairs of array names and maximum power for the day for that array in kilowatts. (Again this is formatted to fit on the page.) For example:

```
>>> data = load_data('03-08-2014')
>>> max_power(data)
[('UQ Centre, St Lucia', 351.19),
 ('Concentrating Array', 7.0),
 ('Multi Level Car Park #1', 275.85),
 ('Multi Level Car Park #2', 269.05),
 ('Sir Llew Edwards Bld.', 64.17),
 ('Prentice Building', 13.56),
 ('Advanced Engineering Bld.', 86.15),
 ('Learning Innovation Bld.', 26.945),
 ('Global Change Institute', 95.463)]
>>>
```

2.2.5 Display the Statistics

`display_stats(date)` takes a date in the correct format and prints out the required statistics. No value is returned.

For example:

```
>>> display_stats('03-08-2014')

Statistics for 03-08-2014

Maximum Temperature: 19.8C at times 14:04, 14:05, 14:06

Total Energy Production: 4218.2kWh

Maximum Power Outputs:

UQ Centre, St Lucia                 351.2kW
Concentrating Array                   7.0kW
Multi Level Car Park #1             275.9kW
Multi Level Car Park #2             269.1kW
Sir Llew Edwards Bld.                64.2kW
Prentice Building                    13.6kW
Advanced Engineering Bld.            86.2kW
Learning Innovation Bld.             26.9kW
Global Change Institute              95.5kW
```

```
>>>
```

2.2.6 The Top-Level Interface

`interact()` is the top-level function that defines the text-base user interface as described in the introduction.

2.2.7 Hints

For string processing: `strip, split, partition`
For user interaction: `raw_input`

2.3 Extra Task for CSSE7030 Students Only

The CSSE7030 students are required to add extra features to the program. (**Note:** CSSE1001 students will not gain credit for attempting the CSSE7030 task.)

The added features consists of an extra command in interact and an extra function.

The extra command is of the form `week DD-MM-YYYY` as in the example below.

```
Command: week 03-08-2014

Date                Temp        Sunlight        Power
-----------------------------------------------------
03-08-2014:      19.8 C     841.0 W/m^2    1189.4 kW
04-08-2014:      19.0 C     742.5 W/m^2    1250.7 kW
05-08-2014:      20.4 C     901.0 W/m^2    1175.5 kW
06-08-2014:      20.9 C     824.0 W/m^2    1137.9 kW
07-08-2014:      22.5 C     686.5 W/m^2     921.9 kW
08-08-2014:      21.9 C     874.5 W/m^2    1232.7 kW
09-08-2014:      20.2 C     799.5 W/m^2    1124.0 kW
```

The extra function is `display_weekly_stats(start_date)` which takes a start date and prints out the information as in the above example for the 7 days from the start date. For each date the maximum temperature, the

maximum sunlight and the sum of the maximum power outputs for the arrays are displayed in the table.

In order to do this you will need to look up the documentation for `datetime` and in particular `strptime`, `strftime` and `timedelta`.

3 Assessment and Marking Criteria

In addition to providing a working solution to the assignment problem, the assessment will involve discussing your code submission with a tutor. This discussion will take place in the practical session you have signed up to in week 7. You **must** attend that session in order to obtain marks for the assignment.

In preparation for your discussion with a tutor you may wish to consider:

- any parts of the assignment that you found particularly difficult, and how you overcame them to arrive at a solution;
- whether you considered any alternative ways of implementing a given function;
- where you have known errors in your code, their cause and possible solutions (if known).

It is also important that you can explain to the tutor how each of the functions that you have written operates (for example, if you have used a for loop or a while loop in a function, why this was the right choice).

Marks will be awarded based on a combination of the correctness of your code and on your understanding of the code that you have written. A technically correct solution will not elicit a pass mark unless you can demonstrate that you understand its operation.

We will mark your assignment according to the following criteria.

Criteria	Mark
Your code is mostly complete, correct, clear, succinct and well commented. You are able to explain your code.	8 - 10
Your code has some problems OR you have some problems explaining your code.	4 - 7
Your code is clearly incomplete, incorrect, too complex or hard to understand OR you have major problems explaining your code.	1 - 3
Your work has little or no academic merit.	0

A partial solution will be marked. If your partial solution causes problems in the Python interpreter please comment out that code and we will mark that.

Please read the section in the course profile about plagiarism.

4 Assignment Submission

You must submit your completed assignment electronically through Blackboard.

Please read
`http://www.library.uq.edu.au/ask-it/blackboard-assessment`
for information on submitting through Blackboard.

You should electronically submit your copy of the file `assign1.py` (use this name - all lower case).

You may submit your assignment multiple times before the deadline - only the last submission will be marked.

Late submission of the assignment will not be accepted. In the event of exceptional personal or medical circumstances that prevent you from handing in the assignment on-time, you should contact the lecturer in charge and be prepared to supply appropriate documentary evidence. You should be prepared to submit whatever work you have completed at the deadline, if required. Requests for extensions should be made as soon as possible, and preferably before the assignment due date.