CSSE1001

Semester 1, 2015

Assignment 2

10 marks

Due Thursday 21 May, 2015, 9:30am

A GUI for plotting yearly average temperature data

1 Introduction

In assignment 1 you wrote a simple tool for loading and displaying daily temperature data. For this assignment you will be using yearly average data and writing a GUI for plotting this data. The user will be able to open up to 8 data files and plot the data for all the files. Unlike in assignment 1, different station data files can start and end at different years.

2 Assignment Tasks

For each class and method that you write you need to provide a suitable comment giving a description and where necessary the type and any preconditions. You should use the triple-quote commenting style.

2.1 Download file

The file `assign2.py` is for your assignment. Add your name and student number in the space provided. When you have completed your assignment you will submit the file `assign2.py` containing your solution to the assignment.

The file already contains some code (in two parts). Do not modify any of this code! The first part should be left at the beginning of the file and consists of some import statements and support code. The second part should appear at the end of your code. **NOTE:** You will lose marks if you don't follow these instructions.

2.2 Write the code

Finally, write your solution to the assignment making sure you have included suitable comments. Your solution should include at least the following classes.

2.2.1 TemperatureData Class

This class is used to hold the temperature data for each loaded station. You are required to use a dictionary to store the data with the station names as keys and `Station` objects (containing the temperature data) as values. Because the dictionary does not preserve the order in which the station data is entered then you need to keep track of the list of station names in the order in which they are loaded. Also the user can choose if a given stations data set is to be displayed and so you need to keep track of which data sets are to be displayed. This class must define at least the following methods.

- `__init__(self)` that initializes the internal data.
- `load_data(self, filename)` loads in data from the given filename (of the form *StationName*`.txt`).
- `get_data(self)` that returns the dictionary of station data
- `toggle_selected(self, i)` that toggles the flag for displaying the station at index `i`.
- `is_selected(self, i)` that returns a boolean used to determine if the data for the station at index `i` is to be displayed.
- `get_ranges(self)` that returns a 4-tuple of the form `(min_year, max_year, min_temp, max_temp)`.
  This information is used to translate between data values and canvas coordinates.

Example:

```
>>> data = TemperatureData()
>>> data.load_data('Brisbane.txt')
>>> data.get_data()
{'Brisbane': Station(Brisbane)}
>>> data.get_stations()
['Brisbane']
>>> data.get_ranges()
(1950, 2014, 24.267, 25.947)
>>> data.load_data('Adelaide.txt')
>>> data.get_stations()
['Brisbane', 'Adelaide']
>>> data.get_data()
{'Brisbane': Station(Brisbane), 'Adelaide': Station(Adelaide)}
>>> data.get_ranges()
(1910, 2014, 21.021, 25.947)
>>> data.is_selected(0)
True
>>> data.is_selected(1)
True
>>> data.toggle_selected(0)
>>> data.is_selected(0)
False
>>> data.is_selected(1)
True
>>>
```

2.2.2 Plotter Class

This class is responsible for doing the plotting and should inherit from `Canvas`.

2.2.3 SelectionFrame Class

This class is the 'widget' used for selecting which stations data is to be displayed and should inherit from `Frame`. It consists of a label and a `Checkbutton`

for each loaded station (in the order loaded).

2.2.4 DataFrame Class

This class is the 'widget' used for displaying the temperatures for a chosen year for each selected station.

2.2.5 TemperaturePlotApp Class

This is the top-level class for the GUI. It is responsible for creating and managing instances of the above classes.

At the top of the application there is a file menu to be used for opening station data files. Below that is a canvas for drawing the data. Below that is the widget for displaying temperatures for the selected year and below that is the widget for selecting which station data to display.

Note that, if the user enters an invalid data file (one that is not readable, one that does not end in '.txt' or one with invalid data) then an exception will be raised. You need to catch exceptions and display them in a pop-up `messagebox` window.

When the user enters a valid data file, the data is plotted in the canvas window using the required colour and a checkbox (also using that colour) is added to the selection frame. When the user enters another stations data it is also plotted (and a check box added) but the old data may need to be replotted as the year and temperature ranges may have increased. Similarly, if the user resizes the application then the canvas will change size and the data will need to be replotted.

The user can interact by clicking the left mouse button - this will draw a vertical line and display the data for the selected year in data frame. If the button is held down and dragged then the vertical line will move with the mouse pointer and the data for that year will be displayed.

Details of the required GUI layout is shown in the examples listed below.

NOTE: You must use pack (rather than grid) to do your GUI layout.

CSSE7030 students only

If the user does a left mouse button click to select a year (as above) and then the user presses any key on the keyboard then it selects a "starting year". If the user repeats the process then an "end year" is selected and then a "best line fit" is drawn between the start year and end year for each selected station. Make the line width 2 so that it stands out more.

2.3 Examples

The course web page contains the following examples.

- Screenshots showing the GUI in use
- Screencast showing the application in action.

2.4 Hints

For redrawing, either because the window has been resized or because the data to be displayed has changed, it's easiest to delete everything on the canvas (`delete(ALL)`) and delete all the data displayed in the data frame and draw all the required lines.

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 12. 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 `assign2.py` (use this name - all lower case).

You may submit your assignment multiple times before the deadline - only the last submission will be marked. After each submission please use Blackboard to check that the file you submitted was the one you intended to submit. Make sure the file is called `assign2.py` and not, for example, `assign2.py.py`

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 may submit a request for an extension.

Requests for extensions should be made as soon as possible, and preferably before the assignment due date. All requests for extension must be submitted on the UQ Application for Extension of Progressive Assessment form:
`http://www.uq.edu.au/myadvisor/forms/exams/`
`progressive-assessment-extension.pdf`
no later than 48 hours prior to the submission deadline. The application and supporting documentation (e.g. medical certificate) must be submitted to the ITEE Coursework Studies office (78-425) or by email to `enquiries@itee.uq.edu.au`. If submitted electronically, you must retain the original documentation for a minimum period of six months to provide as verification should you be requested to do so.