CIS 41B - Lab assignment 2: data analysis / visualization, GUI

Write a program that works with college tuition data in the US. The program lets the user view the tuition trend, the room and board trend, and the total cost of 4 years of college for a range of years.

**Input file**

The input text file is tuition.csv. It is a cleaned version (with only numeric data) of the file tuition\_original.xlsx.  
The xlsx file is from the [College Board](https://trends.collegeboard.org/college-pricing/figures-tables/published-prices-national), and it has column headers to explain the data and each field of data. Before working with the tuition.csv file, you should open the xlsx file to get an overview of how the data is organized.

The dataset contains the average yearly tuition and room-and-board cost for colleges in the US, in 2018 dollars and in original dollars, for the academic years starting in 1971-72 to 2018-19, this current academic year.

Each line of the file contains data for one year. There are 10 fields in each line and they are

- tuition for private 4-years and the % change from previous year (2 fields)

- tuition for public 4-years and the % change from previous year (2 fields)

- tuition for public 2-years and the % change from previous year (2 fields)

- tuition + room and board for private 4-years and the % change from previous year (2 fields)

- tuition + room and board for public 4-years and the % change from previous year (2 fields)

The file contains 2 different datasets: the first set is the cost in 2018 dollars for the years 1971 to 2018, the second set is the same cost but in original dollars (the actual dollar amount for a particular year) for the years 1971 to 2018.

For example: in 1971, the average yearly tuition for a 2-year community college is $190 back in 1971, but converting it to today's (2018) dollar, it's $1,180.

You can assume the input file will always be in the same format, with the same 10 fields and the same 2 datasets. The only difference between input files will be the number of years. The file you work with is 1971-2018. Another input file might have fewer years or more years.

**Program requirements**

The application is made of 2 source files: lab2.py and college.py

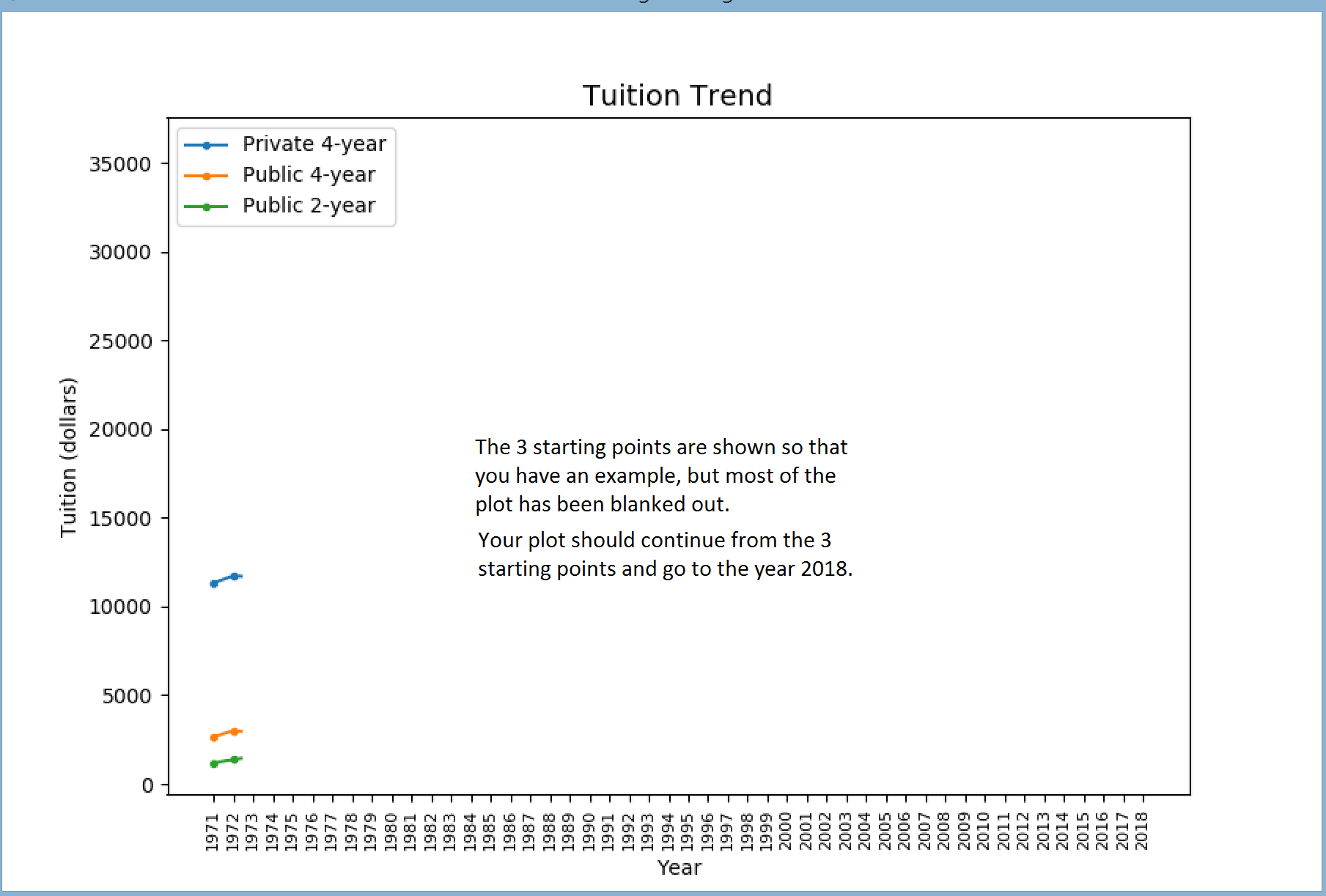
lab2.py is the user interface, and college.py is the data analysis engine.

It is recommended that you write code for college.py in the first week, then write code for lab2.py in the second week.

college.py

Write a class that handles all the data analysis and visualization.

* Create class attributes as needed. The minimum 3 class attributes are:
  + the default filename, set to tuition.csv
  + the start year, set to 1971
  + the end year, set to 2018
* Throughout your application, do not hard code tuition.csv or the year 1971 or 2018. Use the class attributes instead. This way your application will work next year, with only one change to the end year, when the College Board data goes up to the 2019-20 academic year.
* Use one line of code, read data from the file and store the data in a numpy array. This is the only numpy array in your code.
* Take care of file open error if needed.
* There are 2 datasets in the file (2018 dollar and original dollar), the code should work with 2018 dollar costs only.
* Write code to plot the tuition cost over time for private 4-years, public 4-years, and public 2-years:
  + All 3 tuition trends should be in the same plot so it's easy to visually compare them.
  + The ticks for cost in dollars and for the years should be clearly labeled on the 2 axes.
  + There should be a title, a legend, and axis labels so the plot is self-explanatory.
  + Here's a sample plot:



* + To get data for plotting, use a loop that runs 3 times, one time for each tuition trend. There should be no other loop and no container used to get data from the numpy array, because you can take advantage of numpy array features.
* Write code to plot the room and board cost over time for private 4-years and public 4-years. The dataset assumes that most 2-year college students go to local colleges and don't have room and board cost away from home.
  + Both room-and-board trends should be in the same plot so it's easy to visually compare them.
  + Note that you're plotting the room-and-board cost only. Don't count the tuition in your data.
  + The ticks for cost in dollars and for the years should be clearly labeled on the 2 axes.
  + There should be a title, a legend, and axis labels so the plot is self-explanatory.
  + The plot should be in the same format as the example output shown above.
  + Because there are only 2 trends to plot, you can call the plot function twice as 2 lines of code (one call for each line). This means there should be no loop and no container needed to get data from the numpy array to plot. Take advantage of numpy array features.
* Suppose you're the data analyst for a college advising company. The students and their parents (your customers) want to know the most cost effective way for the student to get a 4-year college degree. The paths for the student are:

- Go to a 4-year private college

- Go to a 4-year public college

- Go to a 2-year public college, then transfer to a 4-year private college for the last 2 years

- Go to a 2-year public college, then transfer to a 4-year public college for the last 2 years

Gather data from the numpy array and prepare a plot that can visually explain to your customers the difference in the cost of each choice.

* The customer can enter the graduation year, and your plot will show the cost of 4 years of college for each the 4 paths above.   
  For example, if the customer chooses 2000 as the graduation year, then the plot shows the costs of the 4 paths, using data for the years 1997, 1998, 1999, and 2000.
* It is up to you to choose the correct type of plot to visually show the costs and their differences.
* You should not have to use any loop to get the data from the numpy array in order to plot. Take advantage of numpy array features.
* After you've calculated the 4 costs for the 4 years, return the 4 costs.
* Copy your decorator from lab1 into college.py. Use it to decorate the method to calculate the 4 costs so the decorator will print the 4 costs to screen / console.
* Write a main function as a test driver to test your code above. The main function should have 4 lines of code:
  + Create an instance of the data analysis class
  + Call the method to plot the tuition trend
  + Call the method to plot the room and board trend
  + Call the method to plot the 4-year cost for the 4 college paths, use any valid year you like

1pt Extra Credit:

Demo your code of all the above to me by Mon 4/29, end of office hour (2:30pm). You can make an appointment to see me starting at 8:00am if the rest of your Mon is busy.

I will ask you to briefly show me your code and then run your code, and the 3 plots should show up, one by one.

==================================================================================================

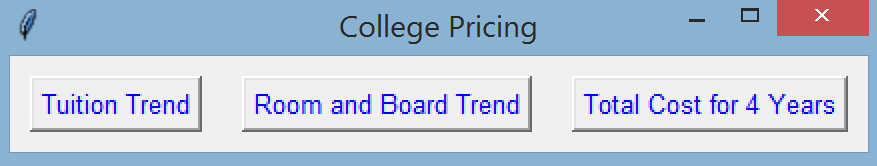
After you've tested and see that college.py fully works, then prepare it to work with the GUI of lab2.py:

* Comment out the call to main() so that the test driver main of college.py doesn't run any more.  
  (If you use the optional: if \_\_name\_\_ ... then you can skip this step. If you don't know what that is, don't worry, we'll get there soon.)
* Comment out all the plt.show() from your plotting method. matplotlib will no longer be showing the plot. In lab2.py we'll "connect" matplotib with tkinter, and tkinter will show the plot.

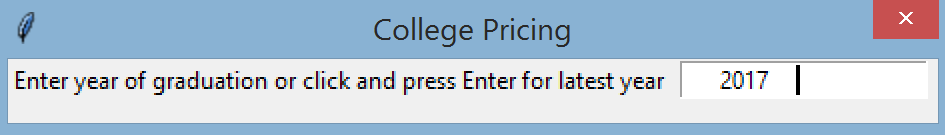
lab2.py

lab2.py is the "front end" or the user interface for the data analysis class. It handles all interaction with the user (input, output, error) through a GUI.

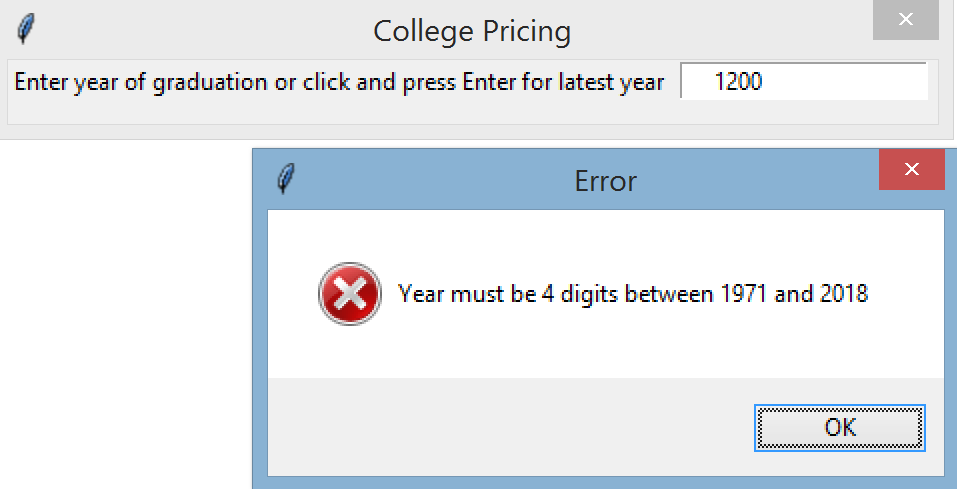
* The GUI must be implemented with each GUI window as a tkinter derived class.
* There are 3 GUI windows, so there are 3 tkinter derived classes in lab2.py
* The main window has:
  + An appropriate title
  + 3 buttons: plot tuition trend, plot room and board trend, plot college cost
  + Sample main window (feel free to use your own wording for the text strings):



* + When the user clicks on one of the plot trend buttons, the plot window appears and shows the appropriate plot.
  + When the user clicks on the total cost button, the dialog window appears to get the graduation year from the user.
  + The main window is the start of the GUI, and when the user clicks X, the window closes and the GUI ends.  
    This means when the user clicks X on the main window, all GUI windows should close.
  + The main window instantiates the data analysis object so that the GUI can interact with it. There should only be one data analysis object being created.
  + If there is any exception from the data analysis class, pop up an error window (discussed below) to explain what the error is. Then close the GUI main window to end the application.
* The plot window is used for all 3 plots (tuition trend, room and board trend, total costs).
  + It displays the plot that the data analysis object creates. It doesn't do the plotting itself.
  + It should be large enough to show all of the plot.
  + The sample plot output shown on page 2 is a plot window.
  + The plot window stays opened on screen until the user clicks X on it to close, or until the user clicks X on the main window to close the entire application.
  + There can be multiple plot windows on screen if the user chooses to plot multiple times and doesn't close the plot windows.
  + While the plot window is opened, the user can still access the main window to select choices.
* The dialog window is used to get the year from the user. It has
  + A prompt string to ask the user for a year or press Enter for the latest year
  + An entry box for the user to enter the year
  + Sample dialog window (feel free to use your own wording for the text strings):



* + If the user presses Enter only, the code fills in the year with the end year of the data analysis object.
  + If the user enters a year, check that it's a 4-digit number, spaces in front or after the 4 digits are okay, and that the value is between the start year and end year of the data analysis object.   
    In the example window above, there are spaces in front and after 2017 (the vertical bar is the cursor), and this value should be considered valid.
  + If the user input is valid, close the dialog window.
  + If the user input is not valid, pop up an error window (described below) and keep the dialog window open until you get a valid input.
  + While the dialog window is opened, the user cannot select the main window to make more choices. The user must either enter a valid choice so that the dialog window is closed automatically, or the user must click X on the dialog window to close it without entering a year.
  + If the user enters a valid year, then the main window will create the plot window to plot the total cost based on the input year.
  + If the user closes the window by clicking X on the dialog window, then there is no plot showing up.
* The error window is used to print any error message to the user
  + It is a tkinter pop up window for error
  + In the pop up window, display clearly what the error message is
  + Sample error window when the user input is not valid



* At the end of lab2.py, write 2 lines of code to start the GUI application:
  + Create an instance of the main window class
  + Call the mainloop() method of the main window object

These should be the only 2 lines of code to start the GUI.

* 1pt Extra Credit for the data scientist in you:

Add a comment block at the end of all your code in lab2.py. In the comment block, write 2-3 sentences to explain to the customers of your college advising business which is the most cost effective way to earn a 4-year degree. Rank the cost in order of least cost to most cost.   
To convince your customers that your reasoning is solid, you need to show the 4 costs for 4 or 5 years (use the decorator output for the 4 costs). Then explain how you chose your 4 or 5 sample years. (The 4 or 5 years don't have to be in sequence, one after another. Why did you choose the particular years in your sample?)