Class Project – Weather Processing App

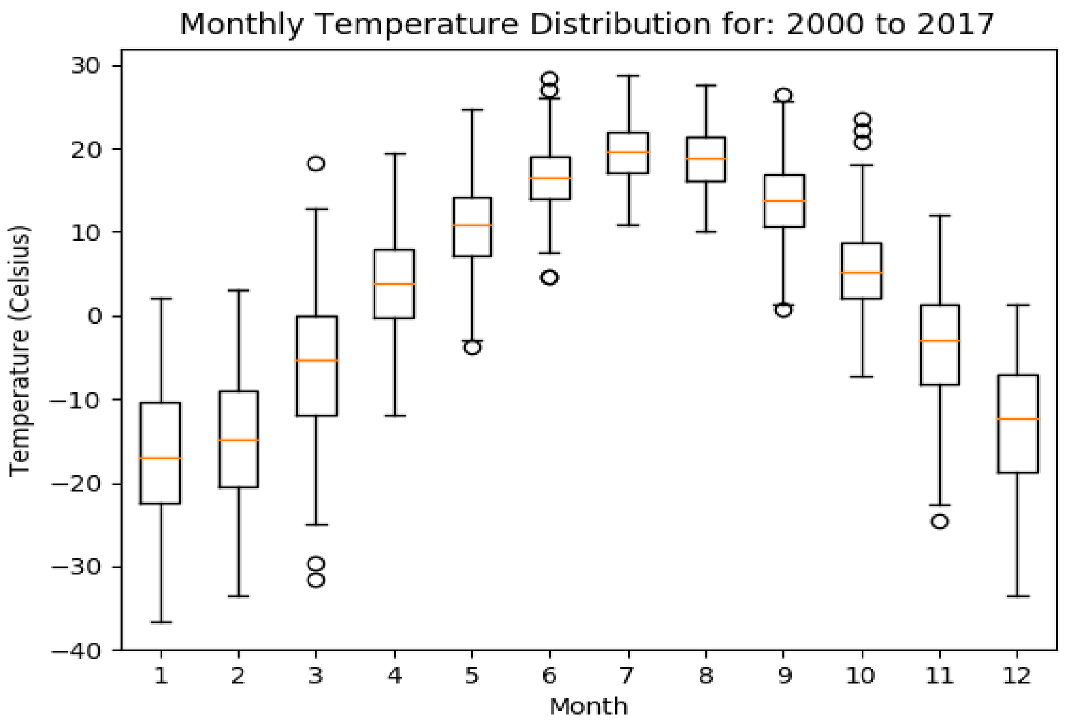
This project will be marked out of 100 points, and is worth 40% of your final grade.

Develop an application with the following features:

Part 1 – Scraping	
Tasks	• Create a scrape_weather.py module with a WeatherScraper class inside. • Use the Python HTMLParser class to scrape Winnipeg weather data (mean temperatures) from the Environment Canada website, from the current date, as far back in time as is available. ○ http://climate.weather.gc.ca/climate_data/daily_data_e.html?StationID=27174&timeframe=2&StartYear=1840&EndYear=2018&Day=1&Year=2018&Month=5# ○ Notice the year and month is encoded directly in the URL.
Input	The URL to scrape.
Output	Be creative. One way could be a dictionary of dictionaries. For example: • daily_temps = {“Max”: 12.0, “Min”: 5.6, “Mean”: 7.1} • weather = {“2018-06-01”: daily_temps, “2018-06-02”: daily_temps}
Bonus	Scrape the min and max temperatures as well, for future processing. (3 points)
Grading	30 points

Part 2 - Database	
Tasks	• Create a db_operations.py module with a DBOperations class inside. • Use the Python sqlite3 module to store the weather data in an SQLite database in the specified format. SQL queries to create and query the DB can be provided if required. The DB format for your reference: ○ id -> integer, primary key, autoincrement ○ sample_date -> text ○ location -> text ○ min_temp -> real ○ max_temp -> real ○ avg_temp -> real
Input	Dictionary of dictionaries from WeatherScraper class.
Output	Whatever data is required to complete the tasks below. Typically a rows tuple.
Bonus	Create a context manager to manage the database connections. (2 points)
Grading	20 points

Part 3 - Plotting	
Tasks	• Create a plot_operations.py module with a PlotOperations class inside. • Use the Python matplotlib to create a basic boxplot: ○ https://matplotlib.org/examples/pylab_examples/boxplot_demo.html
Input	Be creative. One way is a dictionary of lists. For example: • weather_data = {1: [1.1, 5.5, 6.2, 7.1], 2: [8.1, 5.4, 9.6, 4.7]} • The dictionary key is the month: January = 1, February = 2 etc... • The data is all the mean temperatures for each day of that month.
Output	A boxplot displaying one box per month, so it shows all 12 months of the year on one plot. Labels are automatically created from user input. Example:
Bonus	In addition to the above box plot, display a line plot of a particular months mean temperature data, based on user input. For example, display all the mean temperatures from January, with the x axis being the day, and the y axis being temperature. (2 points) • https://matplotlib.org/tutorials/introductory/pyplot.html#sphx-glr-tutorials-introductory-pyplot-py
Grading	15 points



Part 4 – User Interaction	
Tasks	• Create a weather_processor.py module with a WeatherProcessor class inside. • When the program starts, prompt the user to download a full set of weather data, or to update it (optional). • Then prompt the user for a year range of interest (from year, to year). • Use this class to launch and manage all the other tasks.
Input	User supplies input.
Output	Call the correct class methods to accomplish the tasks.
Bonus	When the program starts, prompt user to check for new weather data, and update the database up to today's date. (3 points)
Grading	20 points

Part 5 - Packaging	
Tasks	• Create a Windows package installer using Inno Setup, that allows a user to install your weather app on a Windows 10 computer.
Input	Binary distribution created with the Python pyinstaller module.
Output	Standalone exe installer package for Windows 10.
Bonus	Include your own logo and license agreement as part of the installation process. (3 points)
Grading	10 points

Part 6 – Additional Requirements	
Tasks	• Code must adhere to the PEP8 standard, and will be checked with pytest. ○ To install pytest: pip install pytest pytest-pep8 ○ To use pytest: py.test --pep8 mypythonfile.py • Code must be documented well for easy review and grading.
Bonus	Score a perfect rating on ALL your code, with no errors or warnings. (2 points)
Grading	5 points

Part 7 – Super Bonus	
Tasks	• Create a nice user interface with wxPython for all user interaction. • Label and align everything properly so it looks nice. • The matplotlib charts can open in their own window, they don't need to be integrated into the UI.
Bonus	5 points

Total		
Points	Bonus	Maximum Possible Points
100	20	120/100