Project Final Deliverable

1. Project title

Student-Friendly Schedule of Classes

2. Team members and roles

Ady Weng | Project Manager | ady.weng@gmail.com

Michael Pulley | Analyst | mjpulley@terpmail.umd.edu

Tristan Clark | Researcher | tristansclark@gmail.com

Lorenzo Regala | Tester | enzo.regala4@gmail.com

Andrew Birgin | Researcher | acbirgin@yahoo.com

3. Summary of accomplishments

In this project, we were able to work collaboratively to create a program that displays, sorts by, and searches for specific class information and criteria that students likely find the most pressing. For example, the program current accepts a CSV file as input and sorts via one of the many categories (Average Semester Grade, Pass Rate, Professor Rating, and Format, to name a few), returning a tabular representation by displaying the resulting data frame. It also searches with specific categories; for example, after selecting Name, users are able to search for the name of a course and return a tabular representation of all classes that fulfill that criteria. The program also allows users to save courses for later by registering their row value; the classes are printed out in an orderly list following the conclusion of the program. Finally, we were also able to make many quality-of-life tweaks including but not limited to not terminating the program until the user specifically demands it, in addition to using else and try-except conditions to prevent errors that break the program.

4. Summary of learning

In working as a team, we learned the importance of communication and being available to our partners; while this skillset seems inherent, it is something that we all improved in over the course of this project. Beyond this soft skill, we have also learned more about the concept of abstraction, which involves breaking apart the code into several distinct portions that can be approached from many different perspectives. For example, instead of containing the sort and search capabilities in one function, we split them up into two, and delegated their coding to groups of group members. This ensured that we remained logically sound, focused, and working efficiently.

With regards to the coding portion, we successfully completed research regarding the try-except condition in Python and implemented it in our code (hopefully) correctly. It allowed us to check for ValueErrors and ensure that our program never broke to a significant extent. Our greatest challenge was most likely dealing with incorrect syntactical errors concerning the user input, but we were able to manage it through strategic if-statements and the aforementioned try-except. This was one of many quality-of-life fixes that we made - this project has really taught us about how painful it is to troubleshoot programs with many responses when the tester messes up on one line and has to restart.

5. Next steps

- Add more than the INST major into the database of classes;
- Subsequently, use the Matplotlib Python plotting library to visually represent data from each individual major in a substantial and beneficial manner;
- Allow the user to delete a class from the saved courses list;
- Save the saved_courses list to a text file that can be retrieved following the conclusion of the program;
- Improve the graphical representation of the saved_courses list beyond the default formatting;
- Introduce a “student comments” section where students who have taken classes before can leave specific and appropriate reviews.

6. Individual contributions

- Ady Weng
    - Coding Contributions: sort_courses, save_courses, function_select functions; more specifically, the parts asking for user input and many of the relevant if/elif-statements for sort_courses and search_courses, and implementing the while-loop in the save_courses and function_select functions. Worked on the try-except for search_courses.
    - For those aforementioned functions, wrote much of the in-line comments and documentation.
    - Primary updater of the README file.
    - Managed documentation and Github organization.
- Michael Pulley
    - Coding Contributions: sort_courses, search_courses, save_courses functions; more specifically, if/elif-statements for sort_courses and search_courses, and the logic (and brunt of the code) for save_courses.
    - Worked on the flowcharts for the aforementioned functions.
    - Continued editing the flowcharts wth updated logic to reflect algorithmic developments; most specifically, with the try-catch conditions.
- Tristan Clark
    - Coding Contributions: search_courses, function_select functions; more specifically, the try-except portion of search_courses, and the logic (and brunt of the code) for the function_select function.
    - Wrote relevant documentation comments.
    - Edited the README file for grammatical errors.
    - Researched and formatted the first CSV file.
    - Compiled relevant data from PlanetTerp into the CSV file.
- Lorenzo Regala
    - Coding Contributions: sort_courses, search_courses functions; more specifically, the elif statements for sort_courses and assisting with the try-except used in the course_searches section.
    - Tested all iterations for our code in search of minor bug fixes and issues encountered with the implementation of more data.
    - Edited flowcharts with updated logic.
    - Worked on considering and researching the quality-of-life improvements for the project, including the try-except used in the search_courses function.
- Andrew Birgin

- Coding Contributions: search_courses, function_select functions; more specifically, working through the try-except portion of search_courses (in addition to its user input functionality) and assisting with logical implementation of the select_function.
- Alongside Michael Pulley, contributed to developing flowcharts for our functions.
- Edited the README file for grammatical errors.
- Researched sources to pull class information from to compile our CSV file.
- Subsequently helped compile relevant data into the CSV file – did the brunt of this work, and contributed greatly in developing our method of figuring out a “pass rate."

7. Include everything **for your project in the ZIP file (including previous updates/code-snapshots, where available). Make sure you follow the rubric in the Team Project information doc.**