Final Project Sheet

Project Goals

- Summarize all of your course knowledge in one significant coding project.
- Work individually or as a team (upto 3 members) to implement an interesting system.
- Research a new data structure that we won't cover during this semester.

Project Details

A significant portion of your final course grade (20%) is the project component. There are two tracks designed for the course project.

Project Track 1: Cryptography and Cryptanalysis

Explore a topic in the fields that make up the "art and science of sending and decoding secret messages". This project invites you to implement, test, and evaluate several cryptography and/or cryptanalysis systems. To start, you should investigate, implement, and test ciphers such as the Caesar and Vigenere ciphers. Then, you should use your ciphers to demonstrate that you can successfully send secret messages through, for instance, an email server. In addition to creating and testing these Python programs, your report should include a detailed explanation of how your implementation work. Finally, you should conduct and carefully report on doubling experiments (time function with varied data sizes - roughly double) that evaluate the efficiency of your tool that performs cryptography and/or cryptanalysis. Along with studying content about any other necessary data structures or algorithms, students who pick this project should review the material about arrays and ArrayLists in Chapters 3 and 7.

Project Track 2: Searching and Sorting

One example of sorting technique is Selection Sort. You can implement and test exisiting sorting techniques. Students who had selected this project could implement and then do an empirically study of the performance of your code (using time function with varied data sizes). Or, you could investigate a different algorithm that searches through a data structure (e.g., a Tree or a hashtable) and looks for data that matches a specific pattern. Students who pick this project should review Section 3.1.2 and further study Chapter 12 of the textbook. Next, this project invites you to conduct doubling experiments (time function with varied data sizes - roughly double) to assess the performance of your implementation. Along with including the source code of the doubling experiment framework and your chosen sorting technique, this project invites students to write a performance evaluation report.

Project Track 3: System Implementation

Now that you understand the key topics of this course, you are ready to design, implement, and test your own Python program. Students who had picked this project should first identify a problem domain in which they will work. For instance, you might decide that you want to implement your own command-line client for the Twitter social network by investigating the Twitter4J library. Then, you will need to decide on the data structures and techniques that you will create to store and manipulate your data. Finally, you should conduct experiments to assess the efficiency of your implementation. If you had chosen this project, then you should review Chapter 2 and the chapter(s) for your chosen data structures. For instance, if you decide to store Twitter data in a list (e.g., a SinglyLinkedList or a List), then you should review the relevant details in Chapters 3 and 7.

Project Track 4: Card Game

In this project option, you will implement some card game such that a user can play against the computer. The sole requirement of this project is to implement a Card class, so that each Card has a value property (2, 3, 4, 5, ..., 'J', 'Q', 'K', 'A') and a suit property ("CLUB","HEART", "DIAMOND", "SPADE"). You will need to create a deck of Cards stored in a data structure (a Queue or a Stack would be the most natural options), so that the Cards can be shuffled, dealt in a specified order, and that each Card only exists once. Aside from this requirement, the game and how you choose to implement it is up to you.

The two easiest card games to implement are poker and blackjack, though you may certainly choose to implement another, so long as a user is competing against the computer. You will need to research the rules of each of these games to ensure that your implementation is correct. Using the poker game as an example, you will create your Card data structure, and write a function to deal 5 Cards to the player and the computer. The user then will have to option to keep or discard any of his/her Cards, so you will need to write a function to handle this. Similarly, the computer will have the option to keep or discard any set of Cards. You can make this AI as simple or as detailed as you would like. Finally, you will need a function to determine what kind of hand the user and the computer have (royal flush, flush, straight, etc.), noting that precedence is important because a user with a full house also has a set of 3 and a set of 2. You will then need a function to decide if the user or the computer has won, based off of the same precedence. If both the user and the computer have matching precedence levels (both have a full house), then you'll need to write a tiebreaker function to determine that a queens and 8s full house beats a jacks and 9s full house.

Project Track 5: Conduct Research on Advanced Data Structures

If you are tired of writing code and want to do something different, you have the option of researching a data structure or algorithm that we will not have time to discuss during this semester. Some proposed topics are listed at the end of this section, or you may propose your own topic.

If you select this project option, your writing deliverables is expected to be more significant than the other project options. You can use the textbook as a primary resource for your research, but you must also include at least 4 other resources, 2 of which must not be Web resources.

Suggested topics: Skip List, Red-Black Tree, Boyer-Moore Algorithm, Knuth-Morris-Pratt Algorithm, Splay Tree, Bellman-Ford Algorithm.

Project Track 6: Propose your own

If you choose to implement your own proposed project idea, you should select something with real-life applications or that tests some series of widely-used data structures. You have a significant amount of flexibility with what you can do with your project, so use it wisely! Is there something you wanted to create but never had time? Is there a project from other courses that you always wanted to extend? Juniors: is there an idea you have for your thesis that you want to try out first before committing to it? Seniors: can you supplement your thesis with a programming aspect? It is always an option to discuss and brainstorm ideas with the Professor and do a course project based on that. Setting up a platform to collaborate with the Professor will be particularly useful for you to receive a strong recommendation letter from the Professor in the future and/or work on some future projects such as summer research, thesis, and independent study.

Timeline and Deliverables

1. Team - Fill the team list document by end of Wednesday (04/21) lab time and write out one paragraph with a title and project idea using a file named `project-idea.md`.

2. Proposal – 1-2 pages, Deadline April 26th (5:00 PM EST). Develop the ideas into a proposal document using a file named `proposal.pdf`. Write a 1-2 page technical description of what you propose to do for your project and submit a PDF copy of your proposal through the GitHub link shared. Your proposal does not need to be very detailed at this point. It should describe what project you are going to pursue, what you want to do (the real problem you will tackle, and at least a couple of references to indicate that you have done some research about the problem.

3. Progress Report – 3-4 pages, Deadline May 5th. By this point, you should have made a tremendous amount of progress towards implementing your project and submit the report using a file named `progress-report.pdf`. Were there any unexpected challenges? Did you have to change your initial model/framework or the project skeleton code? You should have also finished the core implementation of your proposed methodology by now. Include everything you have done so far in your progress report, even if it is incomplete. No need to include the actual code (unless you want my help with it), just describe what progress you have made with it. Submit a PDF copy of your progress report using the GitHub link shared.

4. Presentation – The deadline for this part is May 12th 5:00 PM EST. All team should record a 10 minutes video (with screen recording) to virtually present their course project. I expect every team member to participate in the presentation. Please present individually, if your project is done individually by you. It is expected that students use Slides during their presentation. The link to upload will be shared at a later point. By the presentation session, you should have finished implementation, run some preliminary experiments, and done some basic analysis. In the presentation, you should describe the motivation, problem definition, challenges, approaches, and results and analysis. Use diagrams and a few bullet points rather than long sentences and equations. The goal of the presentation is to communicate the important high-level ideas and give intuition rather than be a formal specification of everything you did. Design at least 6 to 10 slides, including a slide with the title of your project and your name. Also, you need to show a short demo of your tool at the end of the presentation.

5. Final Report – 6-8 pages, Deadline May 18th EOD. Incorporate any feedback from the progress reports and the presentation session and submit the final report using a file named `final-report.pdf`. Your final report should be clear and well written, which includes no typos or grammatical errors. Your report should be written in a professional and technical manner. Your report should include the following:

    - The motivation for your project. Why is the problem you decided to solve important or useful?
    - Background for the proposed problem. What have others done for it already? Include references.
    - Detailed data structure overview, description of the proposed implementation, and the complexity analysis. Include pseudocode, diagrams, and examples if appropriate. If you are extending existing work, briefly describe previous work and include references to it.
    - Description of your results. Make graphs, tables, and anything else that can help me understand your results.
    - Conclusion. Give a short overview of your project and its results. Describe what you learned, what were the biggest challenges and the biggest rewards.

Grading Rubric

For each deliverable, you need to submit a PDF with your report (or presentation slides). For your final report, you need to submit any supplementary material (code, data, a README file documenting what everything is, and how to run your program) to the git repository using the link shared:

1. Proposal – 15 points
2. Progress report – 15 points
3. Presentation – 35 points
4. Final report and implementation – 35 points
5. Please make sure to sign the honor code statement in the submission folder.