Final Project Specifications

CSCI 136: Spring 2024

Learning Objectives

1. Learn and implement a data structure (that we did not cover in lecture or lab).
2. Build an application that uses the data structure you implemented.
3. Practice developing code in a group setting.

Overview

This final project is your chance to demonstrate your ability to learn and implement data structures (the major learning goal of CS 136). You will then use a new data structure in an application of your choice. We recommend choosing an application that you are excited about and allows you to be creative.

Workload. Since this final project is worth 15% of your CS 136 grade, we expect the amount of work you put into the final project to be at least **two labs worth of work for each person**. Use this heuristic to calibrate how extensive your application needs to be.

Tasks

1. **Storage & Back-up.** Create a private GitHub repository. Add both project partners and Katie (GitHub name: kakeith) to the repository.
2. **Data structure choice.** Choose a data structure from this Wikipedia list. You must choose one that we did *not* cover in lecture or lab.
3. **ADT.** Decide on the ADT your chosen data structure will implement and and create in interface in Java for that ADT.
4. **Data structure implementation.** Implement (and test) your chosen data structure from scratch (it should *implement* the interface from the step above).
5. **Application.** Build (and test) an application that uses your chosen data structure. Feel free to find real-world datasets online to ingest into your application. Your application should also use at least one other data structure that we have used in lectures or lab (for example, using `java.util.ArrayList` or a Queue implemented with a Doubly Linked List from Lab 4).
6. **Project management.** Allocate work to each project partner. You must have at least 5 git commits from each person (see rubric below). Also, see our tips for developing with multiple people.

Permissible Imports

All imported libraries must be approved by Katie, *except* java.util, java.io, and Java graphics libraries used in Lab 2 (you are welcome to use these; no need for approval).

Deliverables and Dates

1. **Partner preference form (Due: Fri, April 19).** Fill out your preference for a project partner.
2. **Proposal (Due: Mon April 29).** Submit a one-page PDF document to Gradescope that outlines your plan for all the tasks above. Please write in full sentences and justify (1) why you think what you're proposing is feasible and (2) why you are excited about what you're proposing to work on.
3. **Presentation & Demo (May 6-10).** You will present to the class for about 10 minutes and explain your chosen data structure to the class and give a demo of what you have implemented thus far.
4. **Final code submission (Due: May 14 at 4pm).** Double-check that Katie can see your GitHub private repository and submit your code (and data if applicable) as a .zip file to Gradescope. Note: This is a **hard deadline** as the College does not allow work submitted after this date for classes (like ours) that have a final exam.

Grading Rubric

Here's is a general overview of the point breakdown for different components of the final project.

Deliverable	**Component**	**Points**
Proposal	PDF submitted	5
Presentation	Data structure description	10
	Code demo	10
Final Code	Design/organization	10
	New data structure	20
	Additional data structure	5
	Application implementation	20
	Commenting	10
	Documentation in a README.md	10
	> 5 git commits for each person	5
	Testing	10
	Creativity & Mastery	10

Tips for Developing Code with Multiple People

1. **Communicate regularly.** Make sure both project partners are regularly communicate about the pieces of code they are working on.
2. **Developer vs. Tester.** Have one person develop a chunk of code. Have the other person write unit tests (asserts and print statements) for the code to verity its correctness.
3. **Pair program.** Sit down at one machine together. Have one person be the "driver" (person actively writing code) and one person be the "navigator" (reviews each line of code type and thinks about overall design and strategic direction).
4. **Version control.** Work on separate pieces of the codebase when developing. Regularly add, commit and push to GitHub. Before you start developing locally, remember to git pull (your partner's work) and manually resolve any merge conflicts that exist.

Example Project Proposal Sketch

Here's an example project outline and a "sketch" of a project proposal. Note, this is just a "sketch" and your project proposal should be a bit longer. In the project proposal you submit you will need to write full sentences and justify (1) why you think what you're proposing is feasible and (2) why you are excited about what you're proposing to work on.

1. **Storage & Back-up.** Created a private GitHub repository (URL LINK) and Katie has been invited.
2. **Data structure choice.** Bloom filter: A probabilistic data structure that uses hash functions and is used to test whether an element is a member of a set. A query returns either "possibly in set" or "definitely not in set".
3. **ADT.** A Set (operations isElementOf, isEmpty, size, iterate, enumerate).
4. **Data structure implementation.** We'll have the Bloom filter in it's own .java file. We'll have to build the required hash function and test it.
5. **Application.** Databases use Bloom filters to optimize complex queries. Before performing disk-intensive join operations, a Bloom filter can be used to check if the joining keys exist in a particular data set. We'll implement a database join and join separate large files from the NYC Taxi and Limousine Commission (TLC) Trip Record Data (a very large real-world dataset). We'll write one file to inject the data. We'll write another file that implements the join using the Bloom filter.
6. **Project management.** Alice will implement the hash function and Bob will implement the isElementOf and other operations in the Bloom Filter. Alice will write the code that does the database join operation. Bob will write the code that injests the real-world data.