CS201 Data Structures and Algorithms
AY2025/26 Term 1

Project Description

In this project, students work in their project groups to solve an algorithmic problem:

- Using appropriate data structures/algorithm to solve a problem
- Running experiments and discussing the experimental outcomes
- Analyzing the trade-offs of space and time

Project Objective

The objective of this assignment is to implement and experiment with a variety of data structures, covered in class and others explored independently by applying them to solve a real-world computational problem. Real-world problems are often inspired by interesting datasets. Students will identify a computational problem with the given dataset https://www.kaggle.com/datasets/mexwell/skytrax-user-reviews-dataset and design an algorithmic solution to address it.

The Skytrax User Reviews Dataset is a scraped collection of user reviews from Skytrax (airlinequality.com) covering multiple categories such as airlines, airports, seats, and lounges. Records include free-text review content plus rich metadata—titles, dates, authors, traveler/cabin/route or aircraft details when available, star ratings across aspects, and a recommended flag. You are not required to perform data cleansing, but you may do so if it helps. However, data cleansing efforts will not be considered in grading.

The key component of this assignment is to analyze how different data structures and configurations affect algorithm performance, particularly in terms of time and space trade-offs. Students are expected to:

- Compare multiple data structures within the same algorithmic context.
- Evaluate theoretical complexities versus practical performance.
- Conduct experiments to validate assumptions and observe thresholds where theoretical expectations may diverge from real-world behavior.

The emphasis is not on achieving optimal performance, but on the learning process, reflection on design choices, and insights gained through experimentation. For example:

While insertion sort has a theoretical worst-case time complexity of $O(n^2)$, it plays a role in TimSort, a hybrid sorting algorithm that combines insertion sort and merge sort techniques. TimSort is widely adopted in languages like Python and Java due to its practical efficiency on real-world data. This demonstrates that algorithms with higher theoretical complexity can be valuable in specific contexts, especially when optimized for typical data patterns.

CS201 Data Structures and Algorithms
AY2025/26 Term 1

Grading Criteria

The project will account for 20% of the overall grade for the course.

Your project is graded on the following components:

- Problem (10%)
  - Interestingness of problem
- Contents (50%)
  - Novelty of Data structures explored
  - Rigour of experiments and evaluation metrics
  - Discussions of 3 most interesting experiments/learning highlights
  - Potential extensions
- Presentation (30%) – 15 minutes (including Q & A)
  - Clarity of presentation
  - Appropriate choice of visual aids in conveying information
  - Efforts reflecting thoughtful planning in presentation quality
- X-factor (10%)
  - What makes the project outstanding. Be creative!
  - Justify your own X-factors in your presentation. Implementing a fancy UI does not qualify as a X-factor.

Peer Evaluation

(a) Each group will perform evaluation of the other teams during the Week 13 project presentations. Each team is expected to pose questions to another presenting group.

(b) Each group member will perform self-evaluation and peer evaluation of their group members that will influence the individual class participation component.

The evaluation is mandatory, and non-submissions will be penalized.

If there are issues within the group, please inform the teaching team for intervention latest by end of Week 10 and not leaving it till the peer evaluation phase.

CS201 Data Structures and Algorithms
AY2025/26 Term 1

Deliverables

- Week 12 Sunday, 11pm : Submission of deliverables via eLearn
- Week 13 : Submission of Peer Evaluation via eLearn in class

The following to be submitted via eLearn

(A) Source Codes

- The implementation must be coded using Java. You are allowed to use other 3rd party Java libraries but do include them in the submission and include the readme file to run the source codes.

G<X>T<X>.zip
- Contain ALL source codes for the experiments discussed in presentation
- Contain proper folder structure for each experiment, labeled meaningfully
- Include a readme file at the root folder
    - describe your folder structure
    - instructions to run your source codes

- Multiple submissions are allowed but only most recent submission kept

(B) Presentation slides