UW ECE XXX, Fall 2023

This course will introduce techniques for solving programming problems with fundamental algorithms and data structures. We will explore and make use of core data structures such as linked lists, stacks, queues, graphs and trees, evaluating the tradeoffs of different data structures and algorithms. Specifically, the course covers algorithms for sorting and searching over several of these data structures. This course will be taught with the Python programming language.

When programmers start writing code to solve specific problems they will face many implementation decisions. These decisions will have consequences on the final system, either in terms of utilizing resources (computation time, runtime), the robustness and stability as it grows, and how easy it is to add potential new features. In this course, we'll see this by focusing on learning data structures as *implementations* of **abstract data types**. An abstract data type describes what you can do with a data type, but not how the data type is implemented; a data structure provides both a description of functionality and a specific implementation of functionality.

This 10-week course is organized around four 2-week modules and a final portfolio.

Topics

This course is organized around 4 core content modules, of roughly 2-week scope each. The last module will focus on topics related to the application of this content in the current technology landscape, current student projects, and

- Module 1: Data Types and Algorithm Analysis
  - Intro to Abstract Data Types and Implementations
  - Algorithms and Alogorithm Analysis
    - Sorting algorithms
    - Choosing different data structures gives us different runtimes (and tradeoffs)
- Module 2: Searching and Sorting
  - Trees and Searching
  - Improving on Comparison Sorts
- Module 3: Priority Queues
  - Binary Heaps
  - Hash Tables
- Module 4: Graphs
  - Traversals
  - Applications
  - Dynamic Programming
- Module 5: Bringing it all together
  - Relevant student project topics
  - Current applications
    - This could include:
      - Computer Vision
      - Microcontrollers and Physical Computing
      - Personal/Voice Assistants
      - Large Language Models/ChatGPT

Course Structure

- 2x2-hr meetings/week

    - One will always be in-person; please attend if possible
    - One will be optionally virtual

Grading

I have not finalized grading, but here is the starting point:

- 60%: 4 mini-projects/assignments, based on module topics
- 30%: 1 course project/portfolio
- 10%: "Participation" (to be specified fully at the start of the course)