Summer 2018

COSI21a

Data Structures and Algorithms

Instructor: Antonella Di Lillo
E-Mail: dilant@cs.brandeis.edu
Phone: (781) 736-2736
Office: Volen 137
Office Hours: TBA

Overview

This course focuses on the design and analysis of algorithms and the use of data structures.

Through the introduction of the most widely used data structures employed in solving commonly encountered problems (e.g. lists, trees, and graphs), students will learn different ways to organize data for easy access and efficient manipulation. Algorithms to solve classic problems (e.g. searching, sorting, hashing, graph algorithms, etc.) will be presented, as well as classic algorithm design strategies (e.g. divide-and-conquer and greedy algorithms). Computational complexity theory will be introduced for studying the efficiency of the algorithms covered in the course.

Prerequisites

COSI 11a and either COSI 12b or permission from the Undergraduate Advising Head or Graduate Program Director.

Learning Objectives

Students who complete this course will be able to:

- Write programs that use data structures such as: arrays, linked lists, stacks, queues, trees, hash tables, and graphs.
- Compare and contrast the cost and benefits of dynamic and static structure implementations.
- Choose the appropriate data structure for modeling a given problem.
- Describe the concept of recursion and give examples of its use, identifying the base case and the general case of a recursively defined problem.
- Compare iterative and recursive solutions for elementary problems.
- Determine when a recursive solution is appropriate for a problem.
- Determine the time and space complexity of simple algorithms and recursively defined algorithms.
- Implement both a greedy and a divide-and-conquer algorithm to solve problems.
- Implement the most common sorting algorithms.
- Design and implement an appropriate hash function.
- Design and implement a collision-resolution algorithm for a hash table.
- Solve problems using the fundamental graph algorithms, including depth-first and breadth-first search, topological sort, minimum spanning tree algorithm, and single-source shortest path.

Textbook

- Data Structures and Algorithm Analysis in Java, by M. Weiss.
  - The textbook is optional, no assignments or required readings will be given directly from the textbook, so you may choose not to purchase it if you like. However, the book makes a useful supplement to the lecture presentations.

Grading

The final grades for the course will be determined using the following weights:

- Homework: 35% NOTE: Late homework will not receive credit.
- Exams: 60%
- Participation: 5%

Academic Honesty

As stated in the Rights and Responsibilities handbook, "*Every member of the University community is expected to maintain the highest standards of academic honesty. A student shall not receive credit for work that is not the product of the student's own effort*."

Programming assignments must be completed individually (unless specified otherwise by the instructor); all code you submit must be your own work. You may discuss general ideas of how to approach an assignment, but never specific details about the code to write. Any help you receive from or provide to classmates should be limited and should never involve details of how to code a solution.

As a student of this course you are agreeing to the following rules:

- You may not work as a partner with another student on an assignment.
- You may not get code from online sources.
- You may not show another student your solution to an assignment, nor look at his/her solution, for any reason.
- You may not have another person "walk you through" an assignment, describe in detail how to solve it, or sit with you as you write it. You also may not provide such help to another student. This includes current or former students, tutors, friends, TAs, web site forums, or anyone else.
- You may not post your homework solution code online or ask others for online help. This includes public message boards, forums, file sharing sites and services, or any other online system.

Under our policy, a student who gives inappropriate help is equally guilty with one who receives it. Instead of providing such help to someone who does not understand an assignment, point him or her to other class resources such as lecture examples, the textbook, or emailing a TA or instructor. You must not share your solution and ideas with others. You must also ensure that your work is not copied by others, such as making sure to log out of shared computers, not leaving printouts of your code in public places, and not emailing your code to other students or posting it on the web. We enforce this policy by running similarity detection software over all submitted student programs.

Course Outline *

Tentative Topics
▪ Pseudocode, Algorithm analysis ▪ Stacks, Queues, Lists ▪ Recursion and Recurrences
▪ Trees, Tree Representation, Tree Traversal Algorithms ▪ Binary Trees
▪ Binary Search Trees (BST), BST operations, Rotation operations to balance BST ▪ Self-adjusting BST
▪ Heaps, Operations on heaps, Heapsort, Priority Queues ▪ Sorting Algorithms ▪ Hash Tables/Hashing
▪ Graphs, Depth-first search, Breath-first search, Topological Sort, ▪ Single-source shortest paths algorithms ▪ Minimum Spanning Trees

* Note: topics are tentative