# Data Structures (CS 1520) Fall 2012

**Time and Place:** 11 AM - 12:15 Tuesday and Thursday in ITTC 322 **and** 10 - 11:50 AM Wednesday in **Wright 112**

**Web-site:** **http://www.cs.uni.edu/~fienup/cs1520f12/**

**Class Email List:** Send messages to Google group for the course at CS-1520-02-fall@uni.edu

**Instructor:** Mark Fienup (fienup@cs.uni.edu)
Office: ITTC 313
Phone: 273-5918 (Home 266-5379)
Office Hours: M 9-11:45, 1:10-3; T 9:30-10:45; W 9-9:45, 1:10-3; Th 9:30-10:45; F 9-11:45

**Prerequisite: Intro. to Computing (CS 1510) and co-requisite Discrete Structures (CS 1800)**

**Goals:** After this course, you should be able to (1) write "medium" sized programs using algorithmic problem solving and functional decomposition in analysis, design, and implementation, (2) implement and understand the algorithms for manipulating the abstract data types (ADTs) stacks, queues, lists, strings, trees, and graphs, and (3) be able to select appropriate data structures when writing medium size programs.

**Text:** *Problem Solving with Algorithms and Data Structures Using Python*, 2nd edition, by Bradley N. Miller and David L. Ranum. Franklin, Beedle & Associates. ISBN: 978-1-59028-257-1 (http://www.pythonworks.org/pythonds)

**Assignments:** Assignments will consist of weekly laboratory exercises along with concurrent weekly or bi-weekly programming assignments.

**Pedagogic Approach:** In class, I'll tend to break up the lecture with active (and group) learning exercises to aid learning. While this is not formally graded, part (5%) of your grade will be based on your participation in (and attendence for) these in-class activities. Students benefit by (1) increased depth of understanding, (2) increased comfort and confidence, (3) increased motivation, and (4) being better prepared to work in groups on the job. This might sound great, but it will require you (and me) to work differently to prepare for class. Before the class, you must read the assigned reading, thought about what I asked you to think about, etc.; otherwise you won't be able to effectively participate during class.

**Grading policy:** There will be three tests (including the final). I'll announce tests at least one week in advance to allow you time to prepare. Tentative weighting of course components is:

| | | |
|---|---|---|
| In-class Work: | 5 % | |
| Labs: | 15 % | |
| Programming Assignments: | 20 % | |
| In-class Test 1: | 20 % | (about Sept. 27) |
| In-class Test 2: | 20 % | (about Nov. 1) |
| Final: | 20 % | (Thursday, December 13 from 10:00 to 11:50 AM in ITT 322) |

Grades will be assigned based on straight percentages off the top student score. If the top student's score is 92%, then the grading scale will be, i.e., 100-82 A, 81.9-72 B, 71.9-62 C, 61.9-52 D, and below 52 F. Plus and minus grades will be assigned for students near cutoff points.

**Scholastic Conduct:** You are responsible for being familiar with the University' Academic Ethics Policies (http://www.uni.edu/pres/policies/301.shtml). Copying from other students is expressly forbidden. Doing so on exams or assignments will be penalized every time it is discovered. The penalty can vary from zero credit for the copied items (first offense) up to a failing grade for the course. If an assignment makes you realize you don't understand the material, ask questions designed to improve your understanding, *not* ones designed to discover how another student solved the assignment. The solutions to assignments should be **individual, original** work unless otherwise specified. Remember: discussing assignments is good. Copying code or test-question answers is cheating.

Any substantive contribution to your assignment solution by another person or taken from a publication (**or the web**) should be properly acknowledged in writing. Failure to do so is plagiarism and will necessitate disciplinary action. In addition to the activities we can all agree are cheating (plagiarism, bringing notes to a closed book exam, texting during an exam, etc.), assisting or collaborating on cheating is cheating. Cheating can result in failing the course and/or more severe disciplinary actions.

**Special Notices:**

- In compliance with the University of Northern Iowa policy and equal access laws, I am available to discuss appropriate academic accommodations that may be required for students with disabilities. Requests for academic accommodations are to be made during the first three weeks of the semester, except for unusual circumstances, so arrangements can be made. Students are encouraged to register with Student Disability Services, 103 Student Health Center, to verify their eligibility for appropriate accommodations.

- I encourage you to utilize the Academic Learning Center's assistance with writing, math, science, reading, and learning strategies. There is no charge for currently-enrolled UNI students. UNI's Academic Learning Center is located in 007/008 ITTC. Visit the website at http://www.uni.edu/unialc/ or phone 319-273-2361 for more information.

## Data Structures Reading Schedule for Fall 2012

| Lect # | Tuesday | | Thursday | |
|---|---|---|---|---|
| 1 | 8/21 | Ch. 1: Python Review; version 2.x vs. 3.x; Classes | 8/23 | Ch. 2: Algorithm Analysis; Big-oh; timing |
| 3 | 8/28 | Ch. 2: Performance of Python Built-in data structures | 9/30 | Ch. 3: Linear data structures; stack implementations |
| 5 | 9/4 | Stack applications; Queue implementations | 9/6 | Queue applications; Deque |
| 7 | 9/11 | Ch. 6.6: Priority Queue: binary heap implementation | 9/13 | Unordered Lists implementations |
| 9 | 9/18 | Ordered List implementation | 9/20 | Ch. 4: Recursion; Run-time stack |
| 11 | 9/25 | Review for Test 1 | 9/27 | **Test 1** |
| 13 | 10/2 | Recursion examples: fibonacci, maze search & backtracking, coin-change problem | 10/4 | Dynamic programming: fibonacci, coin-change problem |
| 15 | 10/9 | Ch. 5: Searching: linear and binary search | 10/11 | Searching: hashing implementations: open vs. closed addressing implementations |
| 17 | 10/16 | Map ADT, performance analysis of hashing | 10/18 | Simple sorts: bubble, selecton, and insertion sorts; Advanced sorts: merge and quick sorts |
| 19 | 10/23 | Ch. 6: tree terminology, binary tree implementation, traversals, parse tree application | 10/25 | Binary Search Tree implementation |
| 21 | 10/30 | Review for Test 2 | 11/1 | **Test 2** |
| 23 | 11/6 | Binary Search Tree implementation and Map ADT | 11/8 | AVL trees |
| 25 | 11/13 | Ch. 7: Graph terminology, traversals (BFS and DFS) and implementations | 11/16 | Graph select algorithms: topological sort, Dijkstra's algorithm |
| | 11/20 | **Thanksgiving Break** | 11/22 | **Thanksgiving Break** |
| 27 | 11/27 | Concurrency Programming | 11/29 | Thread-safe data structures |
| 29 | 12/4 | Concurrency Examples | 12/6 | Review for Final |
| **Final**: Thursday, December 13 from 10:00 to 11:50 AM in ITT 322 | | | | |