COMS W3134: Data Structures in Java

Syllabus, Fall 2015, Columbia University

COMS W3134 introduces fundamental ways of algorithmic problem solving by organizing and processing information efficiently. The course covers basic data types and structures (arrays, linked lists, stacks, queues, trees, sets, and graphs) as well as programming techniques and algorithms that operate on them (sorting, searching, hashing, finding shortests paths,...). We will examine implementations of data structures and algorithms in Java. The course also covers the rudiments of analyzing space and time requirements of algorithms.

Time and Location:
Section 1 (Bauer): Mondays and Wednesdays, 5:40pm-6:55pm, 301 Pupin
Section 2 (Stead): Tuesdays and Thursdays, 5:40pm-6:55pm, 614 Schermerhorn

Instructors:
Daniel Bauer (`bauer@cs.columbia.edu`)
Office hours: Time/Date TBD. Location: 7LW3 Shapiro/CEPSR ("SpeechLab", 7th floor).

Larry Stead (`lss2168@columbia.edu`)
Office hours: Time/Date/Location TBD.

Important Links:
Class website: `http://www.cs.columbia.edu/~bauer/cs3134`
Piazza forum for class discussions (post questions to Piazza before emailing the teaching staff:
`https://piazza.com/columbia/fall2015/comsw3134`

Textbook:
Weiss, Mark Allen (2012). *Data Structures and Algorithm Analysis in Java*. 3rd ed. Prentice Hall. ISBN: 9780132576277
Available through the Columbia Bookstore or your favorite book seller. On reserve at the Columbia Library. You do not have to buy the textbook before class starts.

Teaching Assistants/Recitations:
There will be recitation sessions conducted by the TAs. Attendance is optional but highly encouraged. The recitations are complementary to the lecture and focus on Java programming and homework review. The lecture will emphasizes the theory of data structures and algorithms. In addition, there will be TA hours and TAs will respond to questions on Piazza.

Course Goals:

- Be familiar with basic data structures (arrays, lists, stacks, queues, priority queues, search trees, maps, and graphs).
- Be able to implement these data structures and corresponding algorithms (sorting, searching, hashing, combinatorial optimization) efficiently in Java.
- Be able to use the Java Collections framework.
- Be able to understand and write recursive algorithms.
- Be able to apply basic techniques for analysing time and space requirements of algorithms.

Prerequisites: Knowledge of Java (for example COMS W1004). **If you do not know Java, please contact the instructors before registering for this class!**

Topics and Tentative Schedule:
The following table is meant to provide an overview of the class content, which will be covered roughly in the order specified. The precise schedule is subject to change.

Week of	**Topic**
9/7	Overview. Abstract Data Types. Arrays and Linked Lists.
9/14	Java Review.
9/21	Math Review: Proofs, Series. Intro to Algorithm Analysis and Recursion.
9/28	Stacks and Queues.
10/5	Introducton to Trees. Binary Search Trees.
10/12	AVL Trees, B-Trees.
10/19	Sets and Maps. Hashing.
10/26	Midterm Review. **Midterm on Wed/Thu**.
11/3	Priority Queues (Heaps).
10/9	Sorting Algorithms.
11/16	Graphs. Graph Search. Shortest Paths.
11/23	Spanning Trees. Applications of DFS.
11/30	Algorithm Design Techniques.
12/5	NP-Completeness. Final Review.

Grading and Exams:

- 50% - Homework.
- 20% - Midterm exam (in-class, tentative date: October 28/29th).
- 25% - Final exam (date and room TBD; will be assigned by the registrar).
- 5% - Participation (class attendance, activity on Piazza).

Homework Assignments: There will be seven homework assignments consisting of programming problems and a few theoretical questions. The lowest-scoring assignment will not factor into your final grade. You will have approximately 10 days to complete each assignment. Homework assignments will be submitted through GitHub. Homework 1 will explain the homework submission process in detail. This class uses Java 8. If you use Eclipse or another IDE, you **must** make sure that your code compiles on the command line (with `javac`). Code that does not compile, and undocumented or badly formatted code will result in lower scores. The theory portion of your assignments must be in plain text or PDF format.

Please be aware that this is a work intensive course and that you might spend 10h or more on each homework assignment.

Late policy: You will lose 1% of the total score for every 6 minutes your homework is late. The latest submitted version of your homework will be graded. Homework submitted later than 10h after the deadline will receive no credit.

Academic Honesty Policy (Please Read!)

It is important that you read and understand this section. Any form of academic misconduct will result in a homework or exam grade of zero and will be reported to the Office of Judicial Affairs.

Interaction With Other Students: All homework assignments must be solved individually. You are encouraged to discuss problems with others and to work them out on the whiteboard, but when you sit down to write or code up your solution you must work on your own, without any further interaction. You are not allowed to share your solutions (literal code and theory solutions) with other students.

Online Material: Treat coding problems like paper assignments: You are not permitted to copy any part of other people's work without attribution. This applies to code produced by other students and to material found on the internet. Sometimes online sources (for instance Stackoverflow) can be useful as a reference. If you have to use code snippets found online you must attribute your source in a comment (complete link). You are not allowed to copy *non-trivial* code fragments from these sources. Non trivial code is defined as:

- Any code you do not fully understand.
- Code longer than three lines or complete class or method definitions that directly relate to the homework problem.

Example: You may copy a one-liner that opens and reads from a text file, if you attribute your source, but it is not okay to copy an entire method for inserting into a balanced search tree.

In addition to this policy, the CS department's academic honesty policy applies to this course[1].

[1] `http://www.cs.columbia.edu/education/honesty`