

# CMSC 420: Data Structures

## Spring 2008

**Instructor:** Carl Kingsford, Office: AVW 3223. Email: `carlkcs.umd.edu`. Office hours: Tues 3:15-4:15. If you cannot attend office hours, email me about scheduling a different time.

### Teaching assistants:

- Radu Dondera, `rdondera@cs.umd.edu`, Office Hours: Monday 12:30-2:30.
- Subhojit Basu, `subhojit@cs.umd.edu`, Office Hours: Wed 2-4pm in AVW 1105.

**Class time:** Tues/Thurs 2:00-3:15 in CSIC 1121.

**Course objectives:** Introduce methods for organizing data so that the data can be accessed and updated efficiently within a computer program. Particular emphasis will be placed on handling data that represent geometric items such as points and lines.

### Textbooks:

- Practical Introduction to Data Structures and Algorithm Analysis, C++ Edition, 2nd Edition by Clifford A. Shaffer. Prentice Hall, 2000. ISBN: 0-13-028-446-7.
- (optional) Foundations of Multidimensional and Metric Data Structures by Hanan Samet. Morgan Kaufmann, 2006. ISBN: 0-12-369-446-9.

**Course work:** Assignments will be a mix of programming projects and problem sets. Written problem sets are due at the start of class and **no late homework will be accepted** — turn in what you have completed. If you will miss class, turn in the homework early. Answers to homework problems should be written concisely and clearly. Homework problems that ask for an algorithm should present either a clear English description or pseudocode (*not* full C++ code).

Programming assignments will be due at 11:59pm on the given due date. **Late programming assignments will be assessed a late penalty:** 5% of the total grade for up to 6 hours late, 10% for up to 24 hours late, 20% for each additional late day.

There will be an in-class midterm and a comprehensive final. Approximate grading weight: 50% for homeworks & projects; 30% for final; 20% for midterm.

**Excused absences.** Students claiming an excused absence must apply in writing and furnish documentary support (such as from a health care professional who treated the student) for any assertion that the absence qualifies as an excused absence. The support should explicitly indicate the dates or times the student was incapacitated due to illness. Self-documentation of illness is not sufficient to excuse the absence. Absences for religious observances must be submitted in writing to the instructor within two weeks of the start of the semester. The instructor is not under obligation to offer a substitute assignment or to give a student a make-up assessment unless the failure to perform was due to an excused absence. An excused absence for an individual typically does not translate into an extension

for team deliverables on a project.

**Academic accommodations.** Any student eligible for and requesting reasonable academic accommodations due to a disability is requested to provide, to the instructor in office hours, a letter of accommodation from the Office of Disability Support Services (DSS) within the first two weeks of the semester.

**Academic honesty.** All class work should be done independently unless explicitly indicated on the assignment handout. You may *discuss* general strategies for homework problems and programming assignments with classmates, but must write and/or program your solution by yourself. If you do discuss assignments with other classmates, you must supply their names at the top of your homework / source code. No excuses will be accepted for copying others work (from the current or past semesters), and violations will be dealt with harshly. (Getting a bad grade is much preferable to cheating.)

## Tentative Schedule

### *Basics and Review:*

1/29: **Introduction & background.** Goals, administrivia, big-O notation.

1/31, 2/5: **Basic data structures.** Lists, queues, dequeues, stacks, graphs, trees.

### *Trees and Balanced Trees:*

2/7, 2/12: **Trees.** Definitions, properties, traversals, implementations.

2/14: **Binary search trees.** Dictionary ADT, insertion, deletion, findmin.

2/19: **AVL trees.** Balanced trees, insertion, deletion.

2/21: **Splay trees.** Amortization, splaying.

2/26: **B-trees.**  $k$ -ary search trees, insertion, key rotation, deletion, RB-trees.

2/28: **Heaps.** Leftist, skew, binomial, and fibonacci heaps; priority queues.

3/4: **Sorting.** Heap, insertion, shell, radix, and quick sort; ordering space.

3/6: **Minimum spanning tree.** Prim's algorithm.

### *Midterm:*

3/11: **Review for midterm.** Questions & answers, etc.

3/13: **MIDTERM.** In class exam.

### *Geometric data structures:*

3/25: **Geometric preliminaries.** General position, convexity, intersections, . . . .

3/27: **Range trees.** 1- and 2-dimensional range trees.

4/1: **Interval trees.** Stabbing queries.

4/3, 4/8, 4/10: **Quad-trees & kd-trees.** Geometric queries, insertion, findmin, deletion, range-searching, mesh-generation.

4/15: **Point location.** Trapezoidal map, randomized construction.

4/17: **BSP Trees.** Binary space partitions, hidden-surface removal.

4/22: **Doubly connected edge lists.** Voronoi diagrams, representing 3D data.

### *Other data structures and applications:*

4/24: **Union-Find.** Equivalence classes, path compression, analysis.

4/29: **Skip lists.** Randomization, implementation, analysis.

5/1: **Memory management.** First fit, best fit, buddy system, GC, mark-and-sweep.

5/6: **Hashing.** Linear probing, open addressing, separate chaining, hash functions.

5/8: **Tries.** Digital search trees, suffix trees, de la Brandais trees.

### *Wrap-up:*

5/13: **Review for final.** Questions and answers, etc..