Data Structures II

Prerequisites: A grade of C$^-$ or better in CPSC 281, and CPSC 340; or permission of instructor.

Instructor: David Casperson; **Office:** TL 10-2080; **Phone:** 960-6672; **Departmental Administrative Assistant:** Marva Byfield (960-6490); **e-mail:** `casper@unbc.ca` ; **web:** `http://web.unbc.ca/~casper` .

Lecture times: M W F 10:30–11:20. **Room**: 5-173. There are *no* assigned lab or tutorial times.

Grading Scheme:

Homework:	20%	
Midterm 1:	20%	Wed, Oct 14
Midterm 2:	20%	Fri, Nov 13
Final Exam:	40%	3h in 7–18 Dec

I reserve the right to change the weight of any portion of this marking scheme. If changes are made, your grade will be calculated using the original weighting and the new weighting, and you will be given the higher of the two.

Programming Assigments: There will be approximately four medium-sized programming assignments during the semester.

Text Book: [3] is *required* for this course.

References

[1] Donald E. Knuth. The art of computer programming.

[2] Chris Okasaki. *Purely Functional Data Structures*. Cambridge University Press, 1998.

[3] Mark Allen Weiss. *Data Structures and Algorithm Analysis in Java*. Addison-Wesley, second edition, 2007. for CPSC 482.

Syllabus: The calendar says:

> External sorting and merging, best case, worst case, and average case estimates, time and space estimates for algorithms studied in CPSC 200-3 and 281-3.

, which goes to show that the calendar is out of date.

Goals: a student who successfully completes this course can successfully reason about data structures.
In particular she (or he)

- can articulate appropriate criteria for choosing a data structure;
- can choose appropriate data structures based to solve a higher level problem;
- is familiar with classical data structures; ¡em¿and¡/em¿
- can design and implement new data structures.

Much of the material is from [3]. In particular, this includes: • Chapter 2 • Chapter 3 • Chapter 6 • Chapter 8 • Chapter 10 and • Chapter 11.

Topics include:

- Algorithm Analysis.
- Collection libraries in C++ and Java.
- Amortized complexity.
- Review of lists, trees, and hash-tables.
- Heaps.
- Union find structres.
- Tries.
- Purely functional versus imperative programming.
- Persistent verus ephemeral data structures.
- Strict versus non-strict evaluation. Laziness and thunks.

Cheating: First offenses result in a grade of $-100\,\%$ on the assignment in question and formal notification of the College Dean. Allowing someone to copy your work is cheating. The UNBC Calendar describes academic offenses and possible penalties in more detail.