

CPSC 221: BASIC ALGORITHMS AND DATA STRUCTURES

Last Updated: September 19, 2008

Course Description

This course is an introduction to the design and analysis of basic data structures. It integrates topics in discrete mathematics and data structures, thereby providing a theoretical and practical foundation for many of our 3rd and 4th year computer science courses. The topics in this course are listed on the last page of this document.

Prerequisites CPSC 211, and one of CPSC 121 or MATH 220.

MATH 101 (or equivalent) is a corequisite, meaning that you either need to have credit for MATH 101 (or equivalent) or you have to take it concurrently. The department will enforce these prerequisites and corequisites. Equivalent credit for older versions of these courses (or appropriate transfer credits) will likely be acceptable.

Instructor William Evans (will@cs.ubc.ca), office: ICCS X841¹

Please limit e-mail to questions of a personal/confidential nature (e.g., sickness). The lectures, tutorials/labs, and bulletin board (also known as “Discussions” or “Discussion Group” on WebCT) are the right places to ask general questions. The bulletin board is read by all students in our lecture section, plus the TA’s, and the instructor; so, a lot of us can respond to your questions. Students are encouraged to ask questions, and to respond to other students’ questions, as long as the questions don’t ask for answers to homework (theory or programming assignments).

TAs Landon Boyd (blandon@cs.ubc.ca) and Ashiqur KhudaBukhsh (ashiqur@cs.ubc.ca)

Plus there will be several markers assigned to this course.

Office Hours

Will Monday 13:00-14:00 & Thursday 09:00-10:00 in ICCS X841

Landon Wednesday 15:30-16:30 & Friday 10:00-11:00 in ICCS X150

Ashiqur Tuesday 14:30-15:30 in ICCS X150

Lab/Tutorial Times (and a few comments about the programming in this course)

Labs/tutorials start during the second week of the course (Sep. 9th and after). You must be registered in one of the following lab/tutorial sections. Please pay attention to the handin dates for the programming labs – these are based on your registered lab section.

Lab	Date & Time	Location	TA	TA Helper	size
L1A	Tue 11:00–13:00	ICCS X350	Ashique	TBA	26
L1C	Thu 15:30–17:30	ICCS X251	Landon	TBA	22
L1D	Thu 10:30–12:30	ICCS X251	Ashique	TBA	19

¹Note that “ICCS” stands for the regular CS dept.’s building, sometimes called “CICSR” or “ICICS” or “ICICS/CS”.

The labs and tutorials for this course are combined into a single 2-hour block. Normally, the “tutorial” part will be the first half. This is the structured part of the 2-hour block, where the TAs will be presenting supplementary material to complement the lectures. Examples will be stressed, and there will be some group thinking questions. There will also be C++ programming topics that are relevant to your assignments. Please give the TAs your attention during the tutorials.

The “lab” part will be the second half of the 2-hour block. This time is less structured, and is used as a consulting hour in the lab. If you are struggling with C++ programming, and if there is space in the lab, you are welcome to attend as many of these in-lab consulting hours as you like. There will be situations where more than one hour of a given 2-hour block will be devoted to tutorials, and other situations where more time will be devoted to labs. This flexibility explains why some CPSC courses no longer have separate lab and tutorial sections.

Since you are now in second year, you’ll need to pick up a lot of the programming concepts and syntax on your own; however, the TAs are there to help. There are also online C++ learning resources that you may find useful, such as the C++ programming pages on WebCT. The Nyhoff textbook has an online site that contains some downloadable code for the examples which appear in the textbook (e.g., code that appears in the figures in the textbook). The URL is `http://cs.calvin.edu/books/c++/ds`.

PRS Clickers

We will not be using PRS clickers this term. If you got a PRS clicker rebate coupon when purchasing the textbook, then you can hang on to the coupon for possible future use.

Textbooks

You can get these at the *UBC bookstore*, *Discount Textbooks* in the University Village, online, etc.

- Larry Nyhoff. *ADTs, Data Structures, and Problem Solving with C++*, 2nd edition, Prentice-Hall, 2004.
- Susanna Epp. *Discrete Mathematics with Applications*, 3rd edition, Thomson/Brooks-Cole, 2004.

This is the same textbook that you used for CPSC 121. If you used the book by Rosen instead (see below), then that’s fine; you can continue to use it in CPSC 221.

There is a *Student Solutions Manual/Study Guide* for this text that is optional for this course. In Appendix B of the Epp textbook, you’ll find answers to selected exercises. The Student Solutions Manual/Study Guide “contains complete solutions to every exercise that is not completely answered in Appendix B and whose number is divisible by 3. The guide also includes alternative explanations for some of the concepts, and review questions for each chapter.”

Course Reserves

We have requested that 1-2 copies of the textbooks (Epp and Nyhoff) be on reserve for short-term loan (e.g., 2 hours) at both: (a) the Ike Barber Library/Learning Centre Reserve room, and (b) the ICCS Reading Room (located near the CPSC main office). The Reading Room’s Web page at <http://www.cs.ubc.ca/local/reading/> has links to direct you to the course reserves for (a) and

(b); however, the textbooks will likely be of value to you in years to come, so we strongly encourage you to buy the books.

Note: Some of our textbooks are cross-listed with CPSC 121 and 260, so you may have to tell the librarian to check under those courses, if he/she can't find the book in question or, if our copy is currently out.

Additional Reference Material (optional)

- Kenneth Rosen. *Discrete Mathematics and Its Applications*, 5th (or 6th) edition, McGraw-Hill, 2003 (or 2007).

This is also very good book. In fact, we used it as a textbook a year or two ago. It is similar in style and coverage to the Epp textbook.

There is also a *Student's Solutions Manual* for this book. In the back of the Rosen textbook, you'll find answers to the odd numbered questions, but in the Student's Solutions Manual, you'll find more detailed answers for the odd numbered questions. A copy of this book is available on 2-hour reserve in the Reading Room.

- Walter Savitch. *Absolute C++*, Addison-Wesley, 2003 or 2nd edition, 2005.

This is a good book for learning C++. We used it for some of our first year CPSC courses several years ago, and it remains an excellent reference book. Two copies are on one-day reserve in the Reading Room. This book, or others like it in the Reading Room and UBC Library, may be valuable if you find that the Nyhoff textbook, the online WebCT C++ notes, etc., aren't detailed enough for you.

- Harley Hahn. *A Student's Guide to Unix*, McGraw-Hill, 1994 (or 2nd edition, 1996).

This is a good book for learning Unix, and for use as a reference. This is probably a worthwhile investment. A copy is in the Reading Room for 1-day loan.

- Lastly, there are many other books on discrete math and data structures in the Reading Room and UBC Library.

Use of WebCT

WebCT² (www.vista.ubc.ca) hosts the course bulletin board (discussion group), and is required reading for this course. Please read it at least once per day, and preferably more often. The TAs are also being asked to read it several times per day to check for messages. WebCT should be accessible to you by the second lecture, so don't be alarmed if you don't have access for the first couple of days. Also on WebCT are online tutorial notes for C++ programming topics and data structures topics, and all kinds of PDF documents containing practice questions, programming labs, theory-based assignments, etc. Furthermore, many of the lecture notes are available as PDF documents for downloading. This should save you a lot of handwriting in class, and will allow you to better focus on lecture topics. We will try to post the current set of lecture slides by 8pm the night before the start of a new lecture topic. Once they're posted, please go ahead and download them.

²We'll use the new version of WebCT, called Vista, in this course, but I'll continue to refer to it as WebCT to avoid confusion with a popular operating system.

Unless your memory is great, you'll probably need to take your own notes in class. Although we may post some handouts from time to time on WebCT, you are responsible for taking your own class notes. Please be advised that you are responsible for all material presented in the lectures, tutorials, labs, and the relevant parts of the textbooks.

If you don't already have a Campus-Wide Login (CWL) account, you should visit www.cwl.ubc.ca to get one. All registered students for CPSC 221 automatically have their CWL ID linked to the CPSC 221 WebCT pages. If you are registered in the course, and your CWL/WebCT link hasn't been set up, then a guest account will be available temporarily. If you are taking other courses that use WebCT, then your same CWL ID will be linked with those courses.

We will use WebCT Vista Edition. To get to WebCT, use URL www.vista.ubc.ca and you'll see a list of all the courses that you're taking which use WebCT Vista.

Questions and Technical Support

Questions about course content (e.g., lecture, textbook, assignments, WebCT content) can be posted on the bulletin board, *but please check to make sure that your question hasn't already been asked or answered*. In previous courses, students tend to ask the same questions over and over, and the TAs become progressively slower in answering questions that have already been asked. You can always re-read postings (i.e., read postings that you've already read) by using the "Show All" messages option. The TAs will be monitoring and responding to questions regularly (with less frequency on the weekends). Your classmates are also welcome (and encouraged) to respond to your postings. In fact, WebCT calls the bulletin board a "discussion group", implying that it's for class wide participation.

Please *do not post code* on the bulletin board (other than perhaps very small fragments, if necessary).

Problems with WebCT (rather than course content) should be directed to help@itservices.ubc.ca

Problems with undergrad accounts and servers should be directed to help@ugrad.cs.ubc.ca. Hardware problems should be reported to hardware@ugrad.cs.ubc.ca.

Programming Environment

We will use GNU C++ as the compiler, and the CPSC department's Unix undergrad computing facilities (Solaris). You are welcome to use Microsoft Visual C++ or another C++ compiler, but you should be sure that your code works on the department machines, since the markers will be running/marking it there. This course uses the automated "handin" facilities to submit your code for marking. A standard text editor (e.g., `emacs` or `vi` on Unix) should be sufficient. Many of you will continue to use Unix in your upper-level courses; therefore, time spent learning `emacs` or `vi` should pay off. The handin facility gives you an acknowledgement upon submitting your work; therefore, pay careful attention to it, to make sure that you've submitted your work correctly.

Homework

There will be 3-4 programming assignments, and 4-5 theory/paper-based assignments (e.g., written answers to discrete math and data structures questions). You should not assume that the assignments are equally weighted in terms of marks or workload.

Course Policy on Plagiarism

Don't forget that homework in this course is to be done alone. Although you may discuss problem solving approaches with other students, and get help from the TAs, you must write up, and do, your own work. Plagiarism occurs when you submit someone else's work as though it were your own. Examples of plagiarism include: copying some or all of someone else's code; sending someone your code via e-mail, Microsoft Messenger, etc. (even if it's just for "reference purposes"); hiring a tutor to write some of the code for you; and so on. The penalty for plagiarism, in most cases, is suspension from the University for 4-12 months and a grade of zero in the course. So, please don't turn in someone else's work. For more information please see the Department's Policy on Plagiarism at: <http://www.cs.ubc.ca/ugrad/info/current/Plagiarism.html>

Examinations

If you cannot write an exam (e.g., due to illness), you must obtain suitable documentation (e.g., doctor's note) and inform the instructor as soon as possible—not after you've written the exam or gotten your mark back. With appropriate medical documentation, and at the discretion of the instructor, you will either write a different midterm at a future date determined by the instructor, or your final exam mark will be applied to your midterm mark. (The latter is the more likely situation.)

The final exam will cover the whole course, and its time and date will be determined by the Registrar's office. If you are sick for the final exam, then you must get a doctor's note, and deliver it to your home faculty's office (e.g., if you are in the Faculty of Science, then you should go to the Science Advising office). You will write a different exam at a future date determined by the Registrar's office (probably at the end of the next semester).

The midterms will be held during normal class times. Since lectures are only 50 minutes long, and since a single 45-50 minute midterm will be too time-compressed for most students, there will be 2 midterms, and they are tentatively scheduled for **Wednesday, October 8** and **Wednesday, November 12**.

Grading

Although the instructor reserves the right to modify the grading scheme under unusual circumstances, we intend to adhere to the following policy:

- 25% for the homework (10% for the paper-based assignments, and 15% for the programming)
- 25% for the midterms (12.5% each)
- 50% for the final exam

To pass the course, you must obtain at least a 50% overall course mark (as per the above formula) and you must pass the final exam. In keeping with department policy, students who fail the final exam receive a course grade of $\min\{45, x\}$ where x is the overall course mark. The final exam will cover the whole course, and not just the last part of the course.

Tentative Course Outline

The weeks are approximations only, and will vary depending on when the course starts (e.g., partway into a week), statutory holidays, the timing of midterms, etc. The page numbers listed are *starting* page numbers.

week	description	hours	Epp(3rd)	Nyhoff(2nd)
1	Introduction (handout)	0.5		
	Introductory C++ Programming (self-taught)	0		1.3–2.4 p.16 Apps C,D,E
	Review of Sets (self-taught) Review of Functions (self-taught) Review of Induction and Recursion Loop Invariants and Program Correctness	2.5	5.1–2 p.255 7.1–2 p.389 4.2–3 p.215 4.5 p.245	10.1 p.522 10.6 p.573
2	Big-O, Big-Omega, Big-Theta	2	9.2 p.518	10.4 p.551
	Complexity: Time and Space	1	9.3 p.531	10.4 p.551
3	Linked Lists Stacks Queues and Deques	1.5		6.4–6 p.287 7.1–5 p.315 8.1–3 p.389
	MergeSort	1	9.5 p.564	13.4 p.762
4	QuickSort	1		13.3 p.753
	Counting: Product Rule, Sum Rule, Inclusion-Exclusion, Tree Diagrams	2	6.1–3 p.297	
5	Combinations Permutations	2	6.2 p.306 6.4–5 p.334	
	Binomial Theorem Pascal's Identity	1.5	6.7 p.362 6.6 p.356	
6	Generalized Permutations and Combinations (maybe)	0–1	6.2 p.306 6.4–5 p.334	
	Introduction to Trees, including Implementation and Applications	2	11.5 p.705	12.2 p.651
7	Tree Traversal	1.5		12.3 p.660
	Binary Search Trees	1		12.4 p.667
8	Priority Queues, Heaps, Heapsort	2.5		13.2 p.735
	Pigeonhole Principle	1	7.3 p.420	
9	Implementation of Hash Tables	2		12.7 p.707
10	B+-trees	2		
	Graph Theory: Introduction, Terminology	2	11.1 p.649	16.3 p.911
11	Graph Representation and Isomorphism	1	11.3–4 p.683	
	Connectivity, Euler and Hamilton Paths & Cycles	2	11.2 p.665	
12	Planar Graphs	2		
13	Contingency	0–2		
	Time for midterms	2		
	Total Hours (excl. statutory holidays, etc.)	36–39		

Note: Various topics will include, as subtopics, program correctness and recursion.