Intro to CS II with Intensive C++
CAS CS-113 – Spring 2002
http://www.cs.bu.edu/fac/gkollios/cs113/cs113.html

TR 11:00 AM - 12:30 PM, CAS 213

Instructor: Prof. George Kollios, *gkollios@cs.bu.edu*, phone 617-358-1835.
Office Hours: Monday 12:00am - 1:30pm and Thursday 1:45pm - 3:15pm in MCS 288, or by appointment (the best way to reach me is via email.)

Teaching Fellow: Huan Luo, *huanl@cs.bu.edu.*
Office Hours: TBA

Course Description: This course begins with an intensive introduction to C++ and then covers all the materials in CS 112. You will learn advanced programming techniques involving dynamic memory allocation, pointers, linked lists, stacks, recursion, trees, and some searching and sorting. All of this will be embedded into the highly-disciplined structure provided by the object-oriented programming language C++.

Prerequisites: This course is designed for students who already program proficiently in PASCAL, C, FORTRAN, or some other high-level programming language. Note: If you do not have such previous exposure to programming, then you are requested to transfer to CS 111. Please consult the instructor if you are uncertain about your preparation.

You are expected to be familiar with UNIX and EMACS. Some help will be available in the section, but if you have not used UNIX or EMACS before, then you should attend the appropriate tutorials provided by B.U. Office of Information Technology: http://www.bu.edu/cc/tutorials/.

Required Textbooks: The required textbooks are:

1. C++ How to Program, 3rd edition, Deitel and Deitel, Prentice Hall, 2001, ISBN 0-13-089571-7.
2. Data Structures and Program Design in C++, Robert Kruse and Alexander J. Ryba, Prentice Hall, 1998, ISBN 0-13-768995-0.

Workload: Be forewarned - the workload in this course will be heavy. To master the conceptual material covered in lecture and to become a strong C++ programmer, there will be substantial programming assignments due approximately every other week. This is an intensive course that covers the topics of CS 111 and CS 112 in one semester.

Grading: The course grade will break down as follows:

- 50% programming projects
- 15% midterm
- 25% comprehensive final
- 10% labs, attendance and class participation

Incompletes will not be granted.

Exams: There will be a ninety minute in-class midterm held during the middle of the semester, likely Thursday, February 28. The final will be held during the normal final exam slot. Please make your end-of-semester travel plans accordingly. In the event of serious illness documented by a doctor's note, makeup examinations will be given orally.

Program Submissions: Programming assignments will be submitted via the gsubmit program, usage of which is documented on the course homepage and will be discussed in class. All assignments will be tested for originality by an automated software tool.

Attendance: It is expected that you will attend lecture and the laboratory section for this course and I will take attendance at the beginning of the lecture. Note that, when students are at a borderline between grades, I will check the attendance records before making a final determination.

Late Policy: Programming assignments are typically due Monday at midnight. During the course, you will have **two** opportunities to turn in an assignment up to 24 hours late with no penalty. Under no circumstances will additional time be granted, nor will additional late submissions be granted.

CAS Academic Conduct Code: Academic standards and the code of academic conduct are taken very seriously at our university, the College of Arts and Sciences, and the Department of Computer Science. Course participants must adhere to the CAS Academic Conduct Code - please take the time to review this document if you are unfamiliar with its contents. All instances of academic dishonesty will be reported to the academic conduct committee; first time violators are routinely suspended for a semester or more.

Collaboration Policy: The work that you submit must be your own original work and it is an act of plagiarism to represent the work of another as your own. You are encouraged to discuss the general nature of solutions with other students in the course, but it is not acceptable to collaborate in writing lines of code, nor to share or copy code. Any discussion or collaboration with other students in the course must also be acknowledged in your submission. If you are uncertain whether an action constitutes a violation of the collaboration policy, I will be glad to discuss the matter with you.

Course Outline (subject to change)

Week of		Lecture	Readings	Section
Jan	15	Introduction	**DD** 1	no lab
	22	Objects, flow of control, iteration	**DD** 2	unix, compilation
	29	Functions, recursion	**DD** 3	functions
Feb	5	Classes and methods	**DD** 6	recursion
	12	Vectors, arrays, streams	**DD** 4, 11	classes
	20	Streams, running time analysis	**DD** 11	debugging
	26	Running time analysis (cont.)	**handout**	writing make files
	28	**Midterm** (in class)		
Mar	2	**Spring Break**		
	12	Containers, pointers, dynamic memory	**DD** 5, **KR** 4	O-Notation
	19	Templates, linked lists	**DD** 12, **KR** 4, 6	linked lists
	26	Stacks, queues	**KR** 2, 3	queues, templates
Apr	2	Trees, recursion	**KR** 5, 10	trees, recursion
	9	Searching, Hashing	**KR** 7, 9	inheritance
	16	Sorting	**KR** 8	templates review, sorting
	23	Graphs	**KR** 12	hash tables
	30	Review		Discussion: review

DD = Deitel & Deitel text.
KR = Kruse & Ryba text.