CMSC 114 Computer Science I Spring 2004

1. Instructors

Fawzi Emad	James Maybury
010x sections	020x sections
1109 AV Williams Bldg.	1119 AV Williams Bldg.
301-405-8411	301-405-2672
fpe@cs.umd.edu	jmaybury@umd5.umd.edu

2. Description

The topics covered by this course are the following: file I/O, dynamic memory allocation, object-based programming, recursion, searching algorithms, sorting algorithms, and data abstraction. You will learn the essentials of the C++ Programming language required to study the above topics. For further details, see the last page of this document.

3. Pre/Co requisites

Prerequisite: CMSC 106 (Introduction to C Programming) Corequisite: MATH 141

4. Text

Text: *C++: How to Program*, Deitel & Deitel, Fourth Edition, Prentice Hall

5. Posting Accounts / Class Webpage

Course materials (e.g., projects, assignments, office hours, etc.) will be made available in the public course posting accounts. Directions for obtaining course materials from the posting accounts will be discussed in class. Many course materials will be posted on the class webpage, which is: `http://www.cs.umd.edu/class/spring2004/cmsc114/`

6. Office Hours

The Instructors' and TAs' office hours will be provided in lecture and available in the class posting accounts.

7. E-mails

The Instructors' emails are supplied so you can contact us when necessary in emergency situations, but we regret that we have insuffiencent resources to debug students' projects via email. Additionally, email is a bad medium for holding a discussion. If you have any problems with your projects then stop by during the instructors' office hours or the TAs' office hours. Office hours or personal communication during or after class (if time permits), are the preferred means for asking us questions about the course material.

8. Course Requirements

Quizzes (announced):	Approximately seven	10%
Midterms:	two midterms	30% (15% of the total course grade each)
Programming projects:	seven coding assignments	35%
	Project #0 – 1% (of the total course grade)	
	Project #1 – 3%	
	Project #2 – 4%	
	Project #3 – 4%	
	Project #4 – 4%	
	Project #5 – 8%	
	Project #6 (Required)– 8%	
	Project #7 – 3%	
Final exam:	will be comprehensive	25%

The final course grades may be curved, if necessary, based on each student's total numeric score for all coursework at the end of the semester.

You must provide a working version of your Project #6 in order to pass the course. A working version of a project is one that satisfies the primary output as specified in the project description. ANY STUDENT WITHOUT A WORKING VERSION OF PROJECT #6 WILL RECEIVE AN F IN THE COURSE. For example, if a student has a 100 in every project, exam, and quiz, but does not have a working version of Project #6, then the student will receive an F in the course.

9. Grading Policy

All projects will be due at the time and date indicated on each project assignment. All projects will be graded out of 100 points. After the deadline for a project, you have two extra days to submit a working version of your project. Only the last project submission you provide will be graded; it will supercede any earlier submissions you may have made. For example, if a student has both an on-time as well as a two-day late submission then only the last (two-day late) submission will be graded. The penalties for each day late are as follows:

```
1 day late  -> -15 points (you can get a maximum of 85 points)
2 days late -> -25 points (you can get a maximum of 75 points)
```

If after the third day you have not provided a working version of your project (except for Project #6) then you will receive 0 points for the project. Project #6 is the only exception to this rule. We will accept Project #6 submissions after the second day late, and until 11:59 PM on Friday May 7th. Versions of Project #6 submitted after second day late and before 11:59 PM on Friday May 7th will receive a maximum of 10 points.

Under **no circumstances** will any emailed project, or a project submitted by any means other than that specified on the assignment be considered. There are no extensions on project deadlines unless there are extraordinary circumstances.

10. Missed Assignments

There will be no make-ups for missed hourly exams or quizzes. In the case of a documented medical excuse or documented family or personal emergency the grade for a missed exam/quiz will be calculated from the student's scores for other exams and/or quizzes. Please note that "Verification of Visit" forms from the University Health Center do not constitute a valid excuse – an authorization form to release medical information must be signed for any Health Center documentation to be considered valid (this form is provided by the Health Center.) If it is known in advance that a student will miss an assignment, the student must notify the instructor prior to the due date of the assignment.

11. Important Dates

Exams will be held on the dates below unless a change is necessary to accommodate lecture topics. In that case the revised date will be announced in lecture. The final exam date, however, is fixed. IF YOU HAVE A SITUATION THAT PREVENTS YOU FROM TAKING THE FINAL EXAM AT THE SCHEDULED DATE IT IS YOUR OBLIGATION TO INFORM YOUR INSTRUCTOR DURING THE FIRST TWO WEEKS OF CLASS, IN ORDER TO MAKE THE APPROPRIATE ARRANGEMENTS.

Exam 1: Monday March 8 during your Lab session.
Exam 2: **NEW Wednesday April 14th** during your Lab session (and not the 12th as originally posted).
Final Exam: Friday, May 14th from 4:00 to 6:00. Location to be announced.
Submission of Project #6 working version: 11:59 PM, Friday May 7.

12. Learning Assistance Service

If you are experiencing difficulties in keeping up with the academic demands of this course, contact the Learning Assistance Service, 2201 Shoemaker Building, 301-314-7693. Their educational counselors can help with time management, reading, note-taking and exam preparation skills.

13. Accommodations for Students with Disabilities

Students with disabilities must inform the instructor of their needs at the beginning of the semester.

14. Academic Integrity Statement

Many students in this course believe that because the size of the class is so large, we will not check that each project has been implemented individually. We want to inform you that we do check. We have software tools that allow us to compare every student project against every project across all the sections of the course. Some students believe that by changing variable names, adding/removing comments, etc., they can transform one program into a different one; this is not true. Even with those modifications we are able to recognize one project as a modified version of another project and we will report this as an academic integrity situation. Our advice to you is to do all your projects by yourself.

Every semester we have taught we have caught students cheating on project assignments. Student's academic careers have been ruined by their decision to cheat.

Please carefully read and consider the Computer Science Department policy regarding the use of computer systems, and the Academic Instructional Technology Services' policy regarding acceptable use of computer accounts provided for instructional use, in the Fall 2001 issue of the departmental newsletter, CS Tid-Bits, handed out with this syllabus. Note in particular the penalties for impermissible cooperation on programming projects, which is a violation of the University's Code of Academic Integrity. **Any evidence of unacceptable use of computer accounts or unauthorized cooperation on exams or projects will be submitted to the appropriate authorities or to the Student Honor Council, which could result in an XF for the course, suspension, or expulsion from the University.**

NOTE SPECIFICALLY that projects are to be done INDIVIDUALLY. **For academic honesty purposes, projects are to be considered comparable to an extended take-home exam. That is, any cooperation or exchange of ideas which would be prohibited on an exam is <u>also</u> prohibited on a project assignment.** Working IN ANY WAY with another student will be considered a violation and WILL BE REPORTED to the Honor Council. Students may not discuss pseudocode, design of a program or of individual functions, coding of individual functions, or any high–level code with **anyone** except the instructors and teaching assistants. Should you have difficulty with a programming assignment you should **see your instructor or the teaching assistants in office hours**, NOT solicit help from anyone else in violation of these rules. Students are welcome to study together or to receive help in learning the course concepts from any others, but quizzes, exams and projects are to be solely and entirely each student's own work.

VIOLATIONS OF ACADEMIC HONESTY INCLUDE:

- failing to do all or any of the work on a project by yourself, other than assistance from the instructional staff.
- using any ideas or any part of another student's project, or copying any other individual's work in any way.
- giving any parts or ideas from your project, including test data, to another student.
- having programs on an open account or on a PC that other students can access.
- transferring any part of a project to or from another student or individual by any means, electronic or otherwise.
- hardcoding project output (instead of following the instructions specified in the project description).

IT IS THE RESPONSIBILITY, UNDER THE UNIVERSITY'S HONOR POLICY, OF ANY STUDENT WHO LEARNS OF AN INCIDENT OF ACADEMIC DISHONESTY TO REPORT IT TO THEIR INSTRUCTOR.

15. Course Topics (SUBJECT TO CHANGE)

This list may vary according to the pace of lecture topics.

	Topics
Week 1	Introduction, review, intro to I/O, reference parameters, passing arguments
Week 2	Review, namespaces, continue basic I/O
Week 3	File I/O, BNF, review pointers
Week 4	Dynamic memory allocation
Week 5	Program organization, makefiles, preliminaries for learning about "classes"
Week 6	Intro to classes
Week 7	Continue classes
Week 8	Continue classes
Week 9	Spring break
Week 10	Finish classes, including discussion of "objects inside objects"
Week 11	Linked Lists and recursion
Week 12	Continue linked lists, iterator model
Week 13	Operator overloading
Week 14	Searching and sorting
Week 15	Other topics, as time permits