CS 15-123
Effective Programming in C and UNIX
Spring 2011
Course Syllabus

Instructor Ananda Gunawardena (guna)
Office Location Gates 6005
Phone (412) 268-1559
E-mail guna@cs.cmu.edu
Office Hours T,TR 10:30AM-12:00PM or whenever my office door is open

Section	CA's	Email	Office Hrs
A	TBA		
E	Emily Grove		
F	Kee Young Lee		
G	Sylvia Han		

Course Description: 15-123 Effective Programming in C and UNIX
All Semesters: 9 units
This course is designed to provide a substantial exposure to the C programming language and the Unix programming environment for students with some prior programming experience but minimal exposure to C. Features of the C language that are emphasized include arrays, structs and unions, dynamic memory allocation (malloc and free), pointers, pointer arithmetic, and casting. Data structures that are emphasized include dynamic lists and hash tables. Students will develop a sense of proper programming style in the C idiom, and will be exposed to cross-platform portability issues. Students will learn to use tools such as emacs/vi, make, and gdb to assist them in the design, testing and debugging of their programs. Students will also learn about regular expressions and will be able to use scripting languages such as Perl and Shell scripting to solve simple problems. This course serves as the prerequisite for 15-213.
Prerequisites: 15-110

Course Objectives: This primary objective of this course is to prepare students on how to use C, Unix, scripting tools to manage development projects. The course also emphasizes the preparation for 15-213, Systems Programming course, which in turn prepares them for operating systems and distributed systems and other systems courses. We will use the C language (old fashioned C not C++) to teach the basics of pointers, memory addressing, copying and moving memory, and other fundamental system tasks. We move off the Windows platform to the AFS (Andrew File System) . On AFS we will use the gcc compiler and it's debugger gbd. The use of simple make files will also be emphasized. We assume that you have taken at least one programming course and you are comfortable working with a compiler and also debugging programs. As such the depth and rigor of the course will be more than a typical intro course. You are expected to devote a considerable amount of time to plan, develop, test and debug your code.

Primary Course Text Books:

All course textbooks are optional. Lecture notes are available from

(1) http://www.cs.cmu.edu/~guna/15-123S11/lectures

AND YOU MAY WANT TO CONSIDER PURCHASING THE TEXTBOOK

(2) C Programming Language (2nd Edition) – good one to keep
by Brian W. Kernighan (Author), Dennis Ritchie (Author)

Other Recommended Text Books are:

(3) "C for Java Programmers" by Thomasz Muldner" ISBN: 0-201-70279-7 - Addison Wesley Longman 2000

(4) ANSI C on UNIX by Paul Wang http://www.sofpower.com/pub_bk01.html

(5) **Learning Perl, Fourth Edition** by Randal L. Schwartz, Tom Phoenix, brian d foy
Fourth Edition July 2005 http://www.oreilly.com/catalog/learnperl4/

(6) The UNIX programming Environment by Kernighan and Pike
http://cm.bell-labs.com/cm/cs/upe/

Assignments and Exams: Assignments will be challenging and you will have to solve them independently. Although you are encouraged to discuss assignments with your colleagues, TA's and instructor, you are **not allowed** to use any part of others code (except code used as demo code in class) for completing your assignments. All students must read, type and submit our **cheating policy as your lab assignment 0** (see the end of this document). **The penalties for any detection of dishonesty are severe**.

Debugging C programs are much harder than debugging Java programs. Therefore you may need more than few hours (more like few days working few hours a day) to complete the assignments. You need to start early and ask questions. Some assignments may occasionally contain elements of mathematics. There will be a C programming midterm, script programming midterm and a written midterm, covering concepts discussed in the lectures.

Programming Assignments are usually released on Fridays and are due the following Sunday at 11:59pm. No late work is accepted without a valid medical excuse. If you are not done, hand in what you have. You may get partial credit for what you have completed. You can use 3 total late days for the entire semester for ANY reason. No questions asked. You *cannot use more than one late day per assignment*. When using late days, we will grade with a penalty as you are late. Section TA will keep track of late days and you can trade late days to offset the penalty. **Don't ask for extensions** unless you have a valid documented excuse. **Use one of the late days, instead**. To make the management of the course grading easy, your programs will be graded as usual (max full

credit if submitted on time, 50% penalty - if 0-24 hours late. 80% penalty if 25-48 hours late. No grading after 48 hours). At the end of the course you can trade late days for late penalties. Only one late day can be used per assignment.

Skills Labs: There will be skills labs discussed during recitations as listed in the course schedule. The skills labs are short assignments that often emphasize things related to larger assignments or other important concepts. These labs need to be completed on wednesday during recitation and must be submitted no later than 12 midnight Thursday. Therefore attendance at recitations is mandatory.

Quizzes or Salons: There will be approximately 8 take home quizzes or take home Salons given during the course of the semester. The concept of a salon will be discussed in both lectures and students may have the option to choose your assessment method (quizzes or salons). We will do 2 salons on prior knowledge and midterm preparation (and perhaps final exam preparation)

Attendance: class and recitation attendance is required. TA's will be taking attendance during recitations. Be sure to sign attendance sheet during recitations. You will also be required to come see instructor during office hours (or schedule a time) to introduce yourself. TA's will also have evening office hours to help with course assignments.

Ungraded Homework: We may assign some homework each day following a lecture. We will not be grading these activities. But you are encouraged to meet with instructor or TA to discuss homeworks.

handin/handback System: All assignments are submitted through the AFS handing system. You must attend the first recitation to learn how to submit the labs through AFS handin system. You submit your assignments to handin folder. The path is

/afs/andrew/course/15/123/handin/lab#/YourAndrewID

After grading, we will drop a txt file with comments in the handback folder.

/afs/andrew/course/15/123/handback/lab#/YourAndrewID

Handin folder will be closed after the deadline, and a late program sub-folder will be open for all submissions. If for any reason, you cannot submit your assignment to handin system, you must email assignment directly to your section TA. DO NOT send emails to instructor (you can always copy instructor though). Check your section and see the TA assigned for the section on the syllabus.

Grades: Your grades will be available through AFS path

/afs/andrew/course/15/123/grades/yourID

Typically all assignments are graded within one week from the submission date. We will notify the class when grades are available for a submitted assignment. You can always contact the section TA to inquire about the grade. You have ONE WEEK to discuss your grade with TA. No grade will be reconsidered one week after the grade is released. Please note that this is a strict deadline.

Exam Dates: There will be three exams given on the 8th week (written test during lecture – 80 mins) and 9th week (C programming test – 50 mins) and 13th week (Script programming test – 50 mins) of classes. In the programming exam, you need to complete a given task or two within 50 minutes. You will be given starter code and will be asked to complete the code, compile, run, debug and submit within 50 minutes. The written exam will test your understanding of basic C concepts and questions will be similar to what is discussed during lectures. Final exam is a 3 hour common written exam. No alternative make up dates can be scheduled without a valid medical excuse, school sanctioned activities (i.e. you are a member of the football team and you have a game that day), or other extenuating circumstances. If you want an alternative exam date you must notify me 1 week in advance. Sleeping in is not an extenuating circumstance. Final exams are scheduled until May 10th, 2011. DO NOT make plans to leave campus before that. As of this writing we do not have the scheduled date for 15-123 final exam.

Office Hours and Getting Help: During my office hours you can stop by any time without an appointment. Any other time, if you need some help 5-10 minutes, just stop by. If I am in I will be happy to help out. If you can't stop in during office hours please call or email me to make an appointment at some other mutually convenient time. For **urgent matters** call me on my mobile 412-260-8647. But please don't use my minutes if the matter is not urgent ☺

Caveat: If you are attending class regularly I will make every effort to accommodate extra office hours to help you. But, if you are not attending, or attending very sporadically, I will not make time for you outside of normal office hours. Please attend class. If you don't need to attend class, you probably should not be in the course anyway. I cannot emphasis enough how important it is for you to ask for help as soon as you realize you do not understand a topic.

Course/Teaching Assistants are here to help you. Email them and make appointments to meet and discuss the assignment. Be aware that your TAs will NOT write code for you. They will explain the assignment and the theory behind it. They may suggest coding strategies and point you to help in your text but they will not write code for you. You must make the transition from thought to code.

Grading Policy:

Assessment	Weight
C Programming Midterm	7%
Written Midterm	10%
Quizzes/Salons	10%
Skills Labs	7%
Miscellaneous points	1%
Programming Assignments	40%
Final Exam	20%
Script Programming Exam	5%
Total	**100%**

Percentage	Grade
>= 90%	A
>= 80%	B
>= 70%	C
>= 60%	D
< 59%	**R**

Important: You must have at least 60% or more from supervised work (final exam – 20%, written midterm – 10%, C programming test – 7%, Script programming test – 5%) to get a grade of C or better. If you do not have at least 60% from supervised work, your highest grade will be a D (eg: you have 70% overall average, but supervised work average is less than 60%). Please ask course staff for clarification of this policy.

Cheating Policy

Cheating is a serious matter, and we will treat it as such. As such a large majority of the course grade in 15-123 is based on programming assignments, and we may be running MOSS (an automated tool for detecting program similarities) on all of your source code, as well as doing manual inspections as we see fit. If you choose to store your source code on AFS or other network accessible file systems, you are expected to use standard file system permissions to protect that source code.

1.1 **What Is Cheating**

To clarify what exactly we want you to never do, we're providing you with an explicit list.

1. Handing in work that is fully or partially identical to another student's, beyond any stubbed source code we might provide.
2. Handing in work that is not fully original (i.e. plagiarized source code from any source, including other students, old solutions, electronic sources, or textual sources)
3. Writing code with another student
4. Reading or altering the solution of another student in any way
5. Sharing test data with another student

1.2 What Isn't Cheating

The flip side of the cheating problem is the equally difficult problem of encouraging collaboration. To allow you to work together and teach each other as much as possible, we're also providing you with a list of things that might be considered cheating in other classes, but are allowed here.

1. Reading outside sources for high level algorithm information. This does not include reading full implementations, so much as reading documentation to understand how a particular algorithm works. (Of course, such sources should be cited).

2. Discussing high level implementation ideas with another student

1.3 Repercussions

If you're caught cheating, there are two tiers of reaction that you'll receive from the course staff. If you've not cheated before in this class, and it's a relatively minor offense, we will issue all involved parties a grade of no more than −200% on the assignment that was falsified. If you've been caught cheating before and we catch you again, or if you've cheated in a particularly heinous way (i.e. submitting character for character identical source code as another party), we will remove you from the course immediately and contact your Dean. The full process of cheating prosecution that will be carried out from there can be found online at **http://www.studentaffairs.cmu.edu/acad int**. It's worth noting that results from MOSS have never been overturned in any academic integrity hearing.