CSCI 303/503, Algorithms, Spring 2015

Robert Michael Lewis	Office hours:	
127 McGlothlin-Street Hall	Mon/Tue/Wed/Thu	2pm–3pm
`rmlewi@wm.edu`		
757-221-2032	or by appointment	

Gadgets and gizmos The use of laptop computers, cellular telephones, and other personal electronic devices in class is strictly forbidden except as directed by the professor.

Web sites The main course web site, where homework assignments, programming projects, &c. will be posted, is

`http://rmlewi.people.wm.edu/teaching/csci303`.

I will also use the College's Blackboard site to post grades.

Course description This course starts with an introduction to the idea of algorithm analysis and some of the mathematical techniques we will use. We then undertake a systematic study of some classic algorithms and their complexity, including sorting, searching, and graph algorithms. As we study these topics, we will encounter a variety of algorithm design methods, including divide-and-conquer, greed algorithms, dynamic programming, and randomization.

There is a significant programming component to this course. Programming will be done using the 2011 revisions of C++ and C.[1] The use of C is intended to help explain the rationale for parts of C++.

Attendance, preparation, and participation Each lecture will have an associated reading assignment and possibly other work such as computer experiments, examinations of code, &c., that should be done before the lecture. You are expected to complete these assignments and come to class prepared to discuss them.

Grading The grade for this course will be based on the following components.

Homework	30%
Programming projects	40%
Midterm exam	10%
Final exam	20%.

Both exams will be closed-book, closed-notes.

The final exam is cumulative. As of January 21, 2015, the final exam is scheduled for 9:00 am to 12:00 noon (in MCGL 020) on Monday, May 11. Rescheduling the final exam requires a petition on your part to either the Dean of Undergraduate Studies or the Dean of Students, depending on the situation:

`https://www.wm.edu/offices/deanofstudents/alpha_forms/deferredexams/index.php`

Petitions to the Dean of Undergraduate Studies are due by 5:00 pm on the last day of the classes. **I do not have the authority reschedule a final exam unilaterally.**

Your final grade for the semester will be determined using the following scale. If S is your total score for the semester, then

$90\% \le S \le 100\%$	A- to A
$80\% \le S < 90\%$	B- to B+
$70\% \le S < 80\%$	C- to C+
$60\% \le S < 70\%$	D- to D
$S < 60\%$	F

I reserve the right to relax these standards, but I will not make them more stringent.

[1] Also known as C++ 11 and C11, respectively.

Textbooks The text for this course is

Algorithms, Fourth Edition, Robert Sedgewick and Kevin Wayne, Addison–Wesley, 2011.

The authors have a website associated with the text (and their version of the course) at

`http://algs4.cs.princeton.edu`.

The text uses Java, but we will use C/C++. This is a deliberate choice on my part. On the one hand, you have concrete code to guide you in understanding and implementing the algorithms we study. On the other hand, it also means you cannot mindlessly copy the text's code (or download the code at the Princeton website) and have it work—you will need to understand the algorithms and C/C++ to translate.[2]

The text we will use for C is

The C Programming Language, 2nd edition, by Brian Kernighan and Dennis Ritchie, 1989.

Known simply as K&R, this book is a classic that no computer scientist should be without. While dated, it remains an excellent introduction to C—Ritchie was the original developer of C—and you can learn a lot about programming style from this book.

The textwe will use for C++ is *C++ Primer Plus*, by Stephen Prata, 2012. I only came across this book this summer, but it is a good, systematic introduction to C++. Be warned that some of the comments on style are odd, and his discussion of the harmonic mean is amusingly wrong-headed. These peculiarities are probably explained by the fact that the author has a background in physics.

Suggested supplemental references I have found that it is useful to have multiple C and C++ references. For C, my favorite reference to supplement K&R is

- *C: A Reference Manual*, Fifth Edition, by Samuel P. Harbison and Guy L. Steele, Jr., 2002. As the title says, this is a language reference manual, and not a "teach yourself C" book. The book reflects the C99 standard and has particularly detailed description of the expanded floating point features of C99.

C did not change as much in the 2011 revision as it did in the 1999 revision (C99), so a C99 reference is generally adequate.

On the other hand, C++ changed significantly in the C++ 11 revision, so when choosing a C++ reference, be sure to choose one that is up-do-date and reflects the C++ 11 standard.[3] Here are two books I would recommend.

- *The C++ Programming Language*, Fourth Edition, Bjarne Stroustrup, 2013. This is a somewhat idiosyncratic discussion of C++ by the original developer of C++. The first six chapters or so are a whirlwind introduction to the language, while subsequent chapters dig into the language in depth. This book bears close study, if for no other reason than to discover that if Stroustrup had followed his own advice, he'd have named the language ++C.

- *C++ Primer*, Fifth Edition, by Stanley B. Lippman, Josée Lajoie, and Barbara E. Moo, 2012. I have used this as a general reference for C++ since its first edition in the misty depths of time.

Academic accommodations of disabilities If you have a disability that may affect your participation in this course and wish to discuss academic accommodations, please contact me as soon as possible.

[2] On the third hand, Java is evil, and you are better off working in C/C++.

[3] But stay tuned for C++ 14 and C++ 17!

Policies on homework and programming projects

Please keep in mind that homework and programming projects are subject to the College's Honor Code.

In the interest of keeping the grading both fair to you and manageable for me and the graders, the following regulations are in effect for the semester with regards to homework and programming projects.

Homework policies

- No late homework assignments will be accepted.
- Under no circumstances will I accept homework that is emailed to me.
- Assignments must be delivered to me no later than the start of class on the day that they are due. If you have another class that meets before this one and this class delays your arrival, please discuss alternative arrangements with me. Assignments that are turned in early are welcome.[4]
- I will not answer any questions regarding a homework assignment on the day that assignment is due. This means that you should give yourself plenty of time to work on the problems before the due date.

Working with others on homework You are allowed to work with others *currently enrolled* in CSCI 303 on the homework assignments. However, certain rules are in force, and if you have any questions about whether a consultation with another student might constitute an Honor Code violation, please ask me before you engage in such a consultation.

You may consult with each other, with me, and with texts, papers, or any published material, BUT the final written solution you turn in for the assignment must be composed by you without the help of others. This means that

- You may not look at the written solution of another student, nor show another student your written solution. Doing so constitutes a violation of the Honor System.
- You may not consult students who have already taken CS 303, nor may you consult any work they may have done for CS 303.
- An "empty hands" policy must be observed when you meet with your classmates to discuss the problems on a homework assignment. You are free to discuss any aspect of the homework assignment, but you must leave the meeting without any written or electronic record of these discussions.
- The actual writing of the details of a homework solution must be an individual activity, so that each student earns an individual grade for each homework assignment.
- If you work on a homework assignment with other students in the course, you are required to list their names when you turn in the assignment.

Consulting outside materials If you have any questions about whether consulting an outside source is allowed, please ask be sure to ask for permission before you do so.

Searching for a solution on the web, and then submitting it as your answer for a homework assignment, will be considered a violation of the Honor System. Your consultations are restricted to written material that has been published in books and journals.

If you find a solution in a published source, you may use it to derive the answer to a homework problem, but you should give this source proper credit in your write-up of the solution. In no case should you copy verbatim from a reference without proper attribution.[5]

[4] E.g., you can bring assignments to my office on the morning that they are due. So long as they arrive before I leave for class, I count them as being on time.

[5] That would be plagiarism.

If you have any questions about whether a particular source is acceptable, you are obliged to clear the source with me before consulting it. In general, web pages, except those sponsored by journals or professional organizations, are not acceptable.[6]

Expectations The homework you submit should be examples of effective technical communication.

- Your work must be neat. The graders will not grade papers that have anything crossed-out, are illegible, are untidily torn from notebooks, &c.
- If there are multiple pages, they must be stapled or paper-clipped together. Do not submit papers that are held together by folding, sticky bits of candy, static electriciy, saliva, &c.—they will not be accepted.
- Graders are more favorably disposed toward assignments that are typeset rather than handwritten. For some assignments, this may not be feasible, but when it is reasonable to do so, typesetting your answers to an assignment works to your advantage.[7]
- Your work should be written in correct technical English. If your answers do not contain complete sentences, expect to lose points.
- Proofs by induction must make explicit the inductive basis, the inductive hypothesis, and what is to be proved in the inductive step. The latter should be stated at the start of the inductive step.
- Do not oversimplify your proofs when a proof is the required answer for a homework problem. Showing a reasonable level of detail in a proof shows that you actually understand the proof.
- If critical steps in the solution to the problem are missing, expect to lose points. Since this decision is ours to make, you should be sure to show your work.
- Points will be deducted even if your answer is correct if your answer is more complicated than it needs to be, or if your work is otherwise difficult to read. The written solutions to your homework assignments must be clear and concise as well as correct. Expect to lose points, even if your solution is correct, if it is difficult to read and understand. This includes solutions that are confused, incomprehensible, unnecessarily complicated, verbose, or incomplete.

Programming projects

(Not) working with others You are free to talk to anyone at William & Mary about basic questions concerning C, C++, Linux, editors, and the like. For instance, you may ask about syntax, e.g., what does

```
auto z4 = [=,y]()->int {return (y ?  1 :  2);}
```

do?[8] You may also ask others about the standard library functions and classes, and compiler and linker errors and warnings. You may also search the web for answers about these matters.

You may discuss algorithm design with others in the class, but, as with homework, an ”empty hands” policy must be observed when you meet with your classmates to discuss a programming assignment. You are free to discuss any aspect of the assignment, but you must leave the meeting without any written or electronic record of these discussions.

Searching the web for solutions of the programming projects, and then submitting it as your answer for a project, will be considered a violation of the Honor Code.

Actively soliciting code from another student, whether or not you submit it as your answer for a project, will be considered a violation of the Honor Code. If, in their enthusiasm to help, someone sends you unsolicited code, you should immediately avert your gaze and delete it.

[6] This exclusion includes wikis, web pages for similar courses at other colleges and universities, blogs, &c. It is OK, however, to visit the Princeton website maintained by our text’s authors.

[7] This is a good time to learn LaTeX, if you have not already done so.

[8] This is valid C++.

Expectations The following rules concern your programs.

- All programs written in C must conform to the C11 standard. All programs written in C++ must conform to the C++ 11 standard.
- No credit will be given for programs that do not compile.
- Points may be deducted for programs that compile with warning messages, depending on the nature and severity of the warning. Some warnings actually point to bugs. Others are benign but are issued because of the sometimes cryptic nature of C++.
- Compile (using `g++ -std=c++11 -Wall -pedantic`) and test your program on a CS machine before submitting your project. You code will be graded using the `g++` compiler, which is part the GNU Compiler Collection (GCC). The `g++` compiler is available on any of the machines on the CS system. Since this is the standard by which your program will be judged, you should exercise your program on the CS system before submitting it to ensure that it satisfies the preceding strictures.
- Programs will be compiled and tested using the version of `g++` on the CS lab machines. As of January 21, 2015, this is version 4.8.1.
- Points will be deducted for programs that violate the specifications laid out in the programming project. Deviations from the program specification may prevent any points being assigned
- Except as otherwise noted, you may only include standard headers (e.g., `<iostream>` or `<string>`) in your program. The standard C++ header files are those stipulated in the ISO/IEC 14882:2011 C++ standard.[9]
- You are expected to exercise careful programming discipline. Points will be deducted if your program is poorly written, poorly designed, hideously inefficient, or otherwise Inutterably Gross,[10] even if the program appears to run correctly. In particular, code will be scrutinized closely for memory access and pointer errors.
- More generally, the programming projects will be graded based on the correctness of the programs, the efficiency of the algorithms, the quality of the design, the overall programming style, and the internal documentation.
- Mistakes identified by the graders in an earlier version of your program that remain uncorrected in a later version will lead to further point deductions.

Regrading policy

Requests for regrading must be submitted within one week of the return of the work question. Your submit, in writing, what you wish to have reviewed and an explanation of why you believe the grading to be in error.

If a regrading request is based on discrepancies between the grading received by two students, both students must submit their graded material for a regrade. Please note that the entire problem set or exam, not just the part in question, is subject to regrading. Thus, your total score may increase or decrease as a result.

[9] If you are not sure what is considered a standard header, consult the online documentation for the GNU compilers, specifically Header Files.

[10] For instance, you should not have epic procedures with hundreds and hundreds of lines of code, or a 50 case switch statement—you should modularize, instead.