Rhodes College

Computer Science III: Data Structures and Algorithms
COMP 241

Syllabus
Fall 2012
CRN: 13761

Basic Info:

- Tuesday/Thursday, 11:00am—12:15pm
- Buckman 222

Instructor:

- Betsy (Williams) Sanders
  - Office: Olendorf 419
  - Office Phone: 901.843.3791
  - Email*: sandersb@rhodes.edu

***To ensure a quick response, the subject line of your emails should read "cs241: [subject of question]" NOTE: I will respond to your emails within 24 hours on weekdays and 48 hours on weekends.**

Office Hours:

- **Monday, Wednesday 1:30—3:00,** by appointment

Help through the Math Support Center:

- Course Teaching Assistant:
  - Matt McCaleb and Jeremey Key
- Place: **Barret 033 or 035**
- Hours: **TBA**

Book:

- The course textbook is
  - *Data Structures and Algorithm Analysis in C++, 3e* by Weiss
- Supplemental material will be distributed in class

Other Course Materials:

- Some course material will be available in my public folder.

Prerequisites:

- You must have *taken COMP 142: Object-Oriented Programming* or an equivalent.

Course Description:

- So far, you have acquired proficiency in programming. This course will start the process of your transformation from a programmer to a computer scientist. One of the important tasks of a computer scientist is to make efficient use of computational resources. This course will teach you about different ways of organizing the data to facilitate such efficient use, and will also discuss efficient techniques to perform some fundamental operations in computer science.

- You will learn to use the techniques discussed in this course to solve commonly encountered computer science problems efficiently. This course will also teach you to analyze the efficiency your program in a mathematically rigorous manner. Material you learn in this course is critical to your becoming a good software developer later.

- This course will focus on algorithms, analysis, and the use of basic and advanced data structures. Among the specific data structures covered are strings, stacks, records, linked lists, trees and graphs. Recursion will also be covered. Sequential and random files, hashing and indexed

sequential access methods for files will be discussed. Finally, some standard computer science algorithms (sorting and searching) will be discussed.

- You are free to develop the code for the assignments on your own computer. However, keep in mind that the source code that you submit for the homework assignments must run successfully on the computers in the on-campus lab.

- This IS NOT a course in object-oriented programming!
    - This is a course about efficiency of programs and programming. In this course, we will pursue various meanings of "efficient". It is efficient to:
        - minimize the run time of code, especially code that is destined for re-use.
        - minimize the memory and storage needs of code, recognizing that there may be a trade-off between speed and memory requirements.
        - re-use code, instead of re-writing code.
        - select only the features you need without having to use costly extra features you do not need.

Course Objectives:

- At the end of this course, you should have a basic understanding of fundamental algorithms and data structures so that you are able to:
    - Describe the basic data structures, algorithms and programming techniques, including lists, stack, queues, search trees, hash tables, different sorting algorithms, search algorithms for graphs, and dynamic programming.
    - Apply the techniques from the course when solving programming/algorithm problems. These techniques include methods for sorting and searching, basic graph algorithms, recursion, dynamic programming, and time and space analysis of programs.
    - Select the best algorithm and/or data structure when solving a given programming problem.
    - Analyze the time and space required for the execution of a program, as well as the correctness of a program.
    - Formulate a given programming task as an algorithmic problem, in order to select the best method for solving it.
    - Combine and modify algorithms and data structures, in order to design an efficient program.

Course Outline:

- The course will cover the following topics (not necessarily in this order):
    - Iterative and recursive strategies
    - Basic algorithm analysis
    - Linked lists
    - Stacks and queues
    - Ordered list sorts
    - Hashing/ Hash tables
    - Trees
    - Search trees
    - Heaps and priority queues
    - Sorting Algorithms
    - Graphs and algorithms: Dikstra, minimum spanning trees

Workload:

- It is important to stay current with the material. You should be prepared to devote regular weekly hours to this course. More specifically, you should devote at least 3-4 hours for each in class lecture. In particular, you should expect to spend a significant amount of preparation for this course working on a computer trying example programs and developing programming assignments.

- **Do not wait to the last minute to start your programming assignments. The programming assignments are designed to be challenging. I will give you plenty of time to complete the assignment and will expect you to use your time wisely.**
- You are encouraged to form study groups with colleagues from the class. The goal of these groups is to clarify and solidify your understanding of the concepts presented in class, and to provide for richer and more engaging learning experience. However, you are expected to turn in your own code/solution that represents the results of your own effort.

Programming Assignments:

- All programs assigned in this course must be written in C++. Unless otherwise stated, submit only C++ source files (`.cpp, .h`) and not other files generated by Visual Studio (e.g, `.exe, .obj, .pdb`).
- The first line of each program source code file submitted for credit must be a comment that states your name and the assignment number.
- Each student is responsible for keeping a back-up copy on disk/USB of all source code turned in for an assignment. Failure to do so could result in loss of credit for an assignment.
- Assignments should dropped in my inbox on the day they are due (before class, 10:59 am). Projects received after 10:59 am on the date due are considered late. I WILL NOT ACCEPT emailed assignments (unless I tell you otherwise).
- **LATE** programs will be accepted, with a penalty of one letter grade per day. If you send a late assignment, you need to email me so that I know to check my Inbox for it and grade it. (If a genuine emergency situation arises**,** please see me and we will work something out.)
- **Collaboration:** Please see the *Rules for Completing Programming Assignments Independently* section on this syllabus for further details.
- Programming grades will be graded on **correctness, style AND efficiency**. You will receive a handout that further expands this idea. This document addresses things like good variable names, how to comment your code appropriately, code organization, debugging techniques, etc.
- Grades are assigned to programs as follows by this general guideline:
    - A (100 pts): Program is carefully designed, efficiently implemented, well documented, and produce clearly formatted, correct output.
    - A- (94 pts): This is an 'A' program with one or two of the minor (?) problems described for grade 'B'.
    - B (88 pts): A 'B' program typically could easily have been an 'A' program, but it may have minor/careless problems such as poor, inadequate, or incomplete documentation; several literal values where symbolic constants would have been appropriate; wrong file names (these will be specified per program assignment); sloppy source code format; minor efficiency problems; etc. (This is not an exhaustive list.) You would be wise to consider 'B' the default grade for a working program --- this might encourage you to review and polish your first working draft of an assignment to produce a more professional quality final version of your program.
    - C (75 pts): A 'C' program has more serious problems: incorrect output for important special cases (the "empty" case, the "maxed-out" case, etc.), failure to carefully follow design and implementation requirements spelled out in the assignment description, very poor or inefficient design or implementation, near complete absence of documentation, etc.
    - D: (60 pts): A 'D' program compiles, links, and runs, but it produces clearly incorrect output for typical cases. Or, it may deviate greatly from the design or implementation requirements stated in the assignment description.
    - F (35 pts): Typically, an 'F' program produces no correct output, or it may not even compile. It may "look like a program" when printed as a hard copy, but there remains much work to be done for it to be a correct, working program.

Rules for Completing PROGRAMMING Assignments Independently

- Unless otherwise specified, programming assignments handed in for this course are to be done <u>independently</u>.

- Talking to people (faculty, other students in the course, others with programming experience) is one of the best ways to learn. I am always willing to answer your questions or provide hints if you are stuck. But when you ask other people, what constitutes legitimate assistance, and when does it cross the border and become cheating? As a practical guide, use the following rule:
  - ***RULE**: In working on an assignment, you cannot look at any correct program or correct piece of code for the same assignment which someone else has written.*
- Here are some sample applications of this rule:
  - *If my program does not work, may I ask another student to see if she can spot what is wrong with it?*
    Answer: Yes, since you are showing her incorrect code.
  - *May she suggest the needed changes?* (E.g.: "You need to assign x = 0 up here.")
    Answer: Yes.
  - *May she write the needed changes for me?*
    Answer: No, since then she would be showing you correct code.
  - *May she show me one of her programs from earlier in the course which uses the same technique?*
    Answer: Yes, since that is not for the same assignment.
  - *May two of us compare the outputs from our programs?*
    Answer: Yes. Output is not program code.
  - *If our outputs differ, may we compare our programs to see what the difference is?*
    Answer: No. Assume one of them is correct; the other person should not be looking at it.
- The underlying idea is that you are entitled to seek assistance in ways which will genuinely help you to *learn* the material (as opposed to just getting the assignment done). Programming assignments are graded as a benefit to you; they are your chance to show what you have learned under circumstances less stressful than an exam. In return, I ask only that your work fairly reflect your understanding and your effort in the course.

Homework Assignments:

- You will be given a number of homework assignments throughout the semester.
- Homework assignments are due at the beginning of class.
- Late homeworks will be penalized one letter grade per day.
- If you decide to write (and not type) your solutions to your homework, your handwriting must be legible.
- **YOU MAY NOT DISCUSS THE SOLUTIONS TO THE HOMEWORK ASSIGNMENTS WITH YOUR CLASSMATES.** Working together is considered cheating and will be dealt with appropriately.
- **You are only allowed to use your textbook and your notes to answer your homework assignments.**

Exams:

- There will be two preliminary exams and one final exam:
  - **Prelim 1**: Thursday, October 4th, in class.
  - **Prelim 2:** Thursday, November 15th, in class.
  - **Final Exam** (pick one of the following):
    - Monday, December 10th at 5:30 pm in Buckman 222 (this is COMP241's official exam time)
    - Saturday, December 8th at 8:30 am in Barret 033 (this is COMP141-01's official exam time)
    - Wednesday, December 12th at 5:30 pm in Barret 035 (this is COMP142's official exam time)
- These exams will be closed-notes, closed-book, and closed-neighbor. Exam work will represent your own individual effort.
- Make-up exams will only be given in extreme circumstances.

Grade Breakdown:

- 40 % Programming Assignments
- 20% Homework Assignments
- 25 % Midterms
- 15 % Final

Grade Assignments:

- Grading is based on the below scale:
  - **A : [93%, 100%]**
  - **A- : [90%, 93%)**
  - **B+ : [87%, 90%)**
  - **B : [83%, 87%)**
  - **B- : [80%, 83%)**
  - **C+ : [77%, 80%)**
  - **C : [73%, 77%)**
  - **C - : [70%, 73%)**
  - **D : [65%, 70%)**
  - **D- : [60%, 65%)**
  - **F : [0%, 60%)**
- For borderline cases, I may take into account participation, and/or attendance, and improvement during the semester.

Attendance:

- Attendance is expected for each class as material that is not covered in the book may appear in class. If your attendance deteriorates you will be referred to the Dean and asked to drop the course. Attendance, participation, and apparent overall improvement may also be considered when assigning a final grade.
- Attendance will be checked each class lecture.
- After 5 unexcused absences, I reserve the right to reduce the final letter grade for the course by one letter grade for each additional absence.

Special Accommodation:

- If you are in need of special accommodations, please register with the Office of Student Disability Services (http://www.rhodes.edu/disability) as soon as possible so that all necessary arrangements can be made.

Scholastic Behavior

- Plagiarism, cheating, and similar anti-intellectual behavior are serious violations of academic ethics and will be correspondingly penalized. If you are concerned about a possible violation of this kind, please talk with me. I understand the pressure that students may experience while at Rhodes, and I will try to help as best as I can.
- All programs must be the student's own work and represent the student's individual effort as defined in the *Rules for Completing Assignments Independently* Section, unless otherwise instructed by me. These are all included in the definition of reportable Honor Code violations for this course:
  - Copying all or part of a problem, downloading solutions from the internet and submitting it as your own, having someone else provide the solution for you, or allowing someone else to copy from you.
- If you have any doubt about what type of behavior is acceptable, please clear it with me.
- When you come to class, you are expected to pay attention! Cell phones are strictly prohibited.

Important Dates:

- Drop/Add Period Ends: 8/28/2012
- Extended Drop Period Ends: 9/12/2012
- Withdrawal Period Ends: 10/26/2012

I reserve the right to alter this syllabus as necessary.