Rhodes College

Computer Science III: Data Structures and Algorithms **Syllabus**
COMP 241 Fall 2009
CRN: 10354

Basic Info:

- Monday, Wednesday, Friday 11:00 pm – 11:50 pm
- Barret Library 033

Instructor:

- Betsy Williams Sanders
  - Office: Olendorf 420
  - Office Phone: 901.843.3791
  - Email*: williamsb@rhodes.edu

***To ensure a quick response, the subject line of your emails should read "cs241: [subject of question]"**

Office Hours:

- **Monday, Wednesday, Thursday 1:30 pm – 3:00 pm**
- By appointment

Book:

- The course textbook are
  - *Data Abstraction and Problem Solving with C++*, 5 ed. by Carrano

- Supplemental material will be distributed in class

Other Course Materials:

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

Prerequisites:

- Students must have *taken COMP 142: Object-Oriented Programming* or an equivalent.
- They must also be enrolled or previously taken *COMP 172: Discrete Structures*.

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.
        - spend less time writing a program of equal quality. In fact, it is even more efficient to spend the same time writing a program of higher quality.
        - re-use code, instead of re-writing code.
        - select only the linguistic 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++.
- Each student is responsible for keeping a back-up copy on disk of all source code turned in for an assignment. Failure to do so could result in loss of credit for an assignment.
- Assignments should be dropped in my inbox on the day they are due (before 11:59 pm). Projects received after 11:59 pm on the date due are considered late.
- You will be given a pool of 5 late days. Meaning you can use a late day so that you may turn in a program a day late with no penalty.
- If you use a late day you must indicate this in the assignment when it is turned in. It is your responsibility to keep track of late days that you have used.
- When the late days have been used, LATE programs will be accepted, with a penalty of one letter grade per day. (If a genuine emergency situation arises**,** please see me and we will work something out.**)**
- You are allowed to use the course textbook and the course notes for these programs. The use of any other material is forbidden.
- Collaboration: You are expected to work individually on assigned programs. However, you are allowed encouraged to discuss high-level details of the programs. If group work is allowed, it will be mentioned explicitly in the assignment.
- 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.

Homework Assignments:

- You will be given a number of homework assignments throughout the semester.
- Homework assignments are due at the beginning of class. You may not use a programming late day for homework assignments. I WILL NOT ACCEPT LATE HOMEWORK ASSIGNMENTS. However, if an emergency arises, please let me know.
- 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 midterms and one final exam:
  - **Midterm 1:** Friday, August 2nd, in class.
  - **Midterm 2:** Friday, November 6th, in class.
  - **Final Exam:** Saturday, December 12, 8:30 am **OR** Wednesday, December 16, 8:30 am
- Make-up exams will only be given in extreme circumstances.

Grade Breakdown:

- 40 % Programming Assignments
- 25% Homework Assignments
- 20 % 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 trend may be considered in assigning a final grade.
- Attendance will be checked each class lecture period. . After 5 unexcused absences, each additional absence reduces the final grade for the course by one letter grade.

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.
- Unlike the homework, all major programs and tests must be the student's *own* work, unless otherwise instructed by your instructor. Copying all or part of a major program assignment, or downloading code from the Internet and submitting it as your own, or having someone else write code for your assignment, or having someone else debug your assignment, or *allowing* someone else to copy from you, or coding or debugging someone else's assignment --- these are all included in the definition of reportable Honor Code violations for this course. If you have any

doubts about whether or not a program development practice on a major program assignment is acceptable, please clear it with the instructor before proceeding.
- When you come to class, you are expected to pay attention! Cell phones are prohibited. You should work through the class exercises and NOT surf the web, etc.

Important Dates
- Drop Add Ends: 9/1/2009
- Extended Drop Period ends: 9/16/2009
- Pass Fail Period Ends: 9/16/2009
- Withdrawal Period Ends: 10/30/2009

I reserve the right to alter this syllabus as necessary.