Rhodes College

Computer Science II: Object – Oriented Programming
COMP 142

Syllabus
Spring 2012
CRN # 22269

Basic Info:

- Monday, Wednesday, Friday 12:00 pm – 12:50 pm
- Classroom: Barret 035

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 email should read "cs142: [subject of question]"*
NOTE: I will respond to your emails within 24 hours on weekdays and 48 hours on weekends

Office Hours:

- **Tues/Thurs 1:30 pm – 3:00 pm,** or by appointment
- Please come to office hours! I am here to help.

Peer Tutoring Hours:

- Teaching Assistant: Jeremey Key (keyjb@rhodes.edu) and TBA
- Hours: **TBA**
- Location: Barret 033 or 035 (TBA)
- You should plan to come and work on your assignments when the TA is around so that you can get help as needed.

Textbook:

- The course textbook is C++ *How to Program,* 8th ed., by Deitel and Deitel
- A useful online reference is **`http://cplusplus.com`**

My Public Folder:

- I will make programs that we discuss in class available to you in my public folder. Also, if I distribute any materials in class, I will put a digital copy in my public folder. You can find it on the network: \\Acad_Dept_Pgm\Math_CompSci\Sanders_Betsy\Public\CS142

Prerequisites:

- The course assumes successful completion of CS141 (a grade of C- or better) or significant programming experience. Please come see me if you have not had CS141.

Course Description:

- CS142 is the second course in the sequence for computer science majors or minors and ideally should be taken immediately after CS141. CS142 offers a new perspective on software design through an introduction of the object-oriented paradigm. Special emphasis is placed on the process of building hierarchies of abstractions to hide implementation details through a careful and systematic analysis of problems of moderate complexity. Various design approaches will be explored with the goal of identifying the situations for which each approach is applicable. In addition the course will cover classic data structures, generic programming, and basic memory management techniques.
- This course will use the C++ programming language as the vehicle for exploration of fundamental computer science concepts. However, this is not a course about C++; it is about the structure and interpretation of computer programs.
- The particular C++ environment that will be used in this course is available in the computer labs on Rhodes College campus. Check the postings at Paul Barret Jr. Library for the hours of operation and locations of the on-campus computer labs.

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

Course Objectives:

- At the end of this course, you should be able to:
  - Explain and use the basic principles of object-oriented design
  - Make design decisions that promote reusable code
  - Analyze problems of moderate complexity and propose a design of hierarchies of independent entities that communicate through a clear and well-defined interface
  - Use and implement rudimentary data structures such as linked lists, stacks, queues
  - Use generic programming and be familiar with the Standard Template Library

Course Outline:

- The course will cover the following topics (not necessarily in this order):
  - C++ basics
  - Objects and classes
  - Object-oriented design
  - Inheritance and polymorphism
  - Operator overloading
  - Pointers and memory management
  - Parameterized classes (templates)
  - The Standard Template Library
  - Simple data structures (linked lists, stacks)

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 2-3 hours for each in class lecture. You should expect to spend significant amount of preparation for this course working on a computer to try out example programs and to develop the programming assignments. **Do not wait to the last minute to start your programming assignments.**
- 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 a richer and more engaging learning experience. However, you are expected to turn in your own code 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 be dropped in my Inbox on the day they are due (before class, 11:59 am). Projects received after 12:00 pm on the date due are considered late.
- **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: You are expected to work individually on assigned programs. Please see the *Rules for Completing 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 possibly 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 Assignments Independently (compliments of Mike Sheard**)**

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

Exams:

- There will be two preliminary exams and one final exam:
  - **Prelim 1**: Friday, February 24th, in class.
  - **Prelim 2:** Friday, April 13th, in class.
  - **Final Exam** (pick one of the following):
    - Friday, May 4th at 1:00 pm in Barret 035

    - Friday, May 4th at 11:00 am in our classroom, Barret 020 (this is COMP172's official exam time)
    - Tuesday, May 1st at 1:00 pm in Olendorf 225
- 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.

Quizzes:

- We will have frequent 3 – 7 minute quizzes composed of 1 – 3 questions.
- The purpose of these quizzes is to make sure you keep current with the material and to help me keep track of your understanding.

Grade Breakdown:

- 45 % Programs
- 10 % Quizzes
- 25 % Preliminary Exams
- 20 % Final Exam

- *Note: You are responsible for keeping track of your grades. I do not use an online gradebook. Here is how your final average will be calculated:*

  Final Average= **45** * (sum of all your program grades)/(total # of points on all programs) + **10*** (sum of your quiz grades)/(total # points on all quizzes)+ **25** * (sum of Prelim1 and Prelim2)/(200) + **20** * (your final exam grade)/(100)

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 Ends: 1/18/2012
- Extended Drop Period and Pass Fail Period End: 2/1/2012
- Withdrawal Period Ends: 3/23/2012