Rhodes College

Computer Science I: Programming Fundamentals — Syllabus

COMP 141 — Fall 2008

CRN: 19090

Basic Info:

- Monday, Wednesday, Friday 9:00 am - 9:50 am
- Barret Library 035

Instructor:

- Betsy Williams
  - 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 "cs141: [subject of question]"**

Office Hours:

- **Monday, Wednesday 3:00 pm - 4:30 pm**
- By appointment
- From time to time, I will hold office hours at night in one of the computer labs in Barret Library so that I may address questions about our programming assignments. The details of these office hours will be emailed to you. TAs will also be available (see below).

Help through the Math Support Center:

- Teaching Assistants will be available to you throughout the semester.
- They are a great resource (and are paid to help you!)
- Place: **Barret 033**
- Hours: (subject to change)
  - **Sunday 7:00-10:00 Nick Volgas**
  - **Tuesday 7:00-10:00 Ryan Heinrich**
  - **Thursday 7:00-10:00 Brian Yuan**

Book:

- The course textbook is *Python Programming: An Introduction to Computer Science*, by John Zelle
- Supplemental material will be distributed in class

Other Course Materials:

- All other course material will be available on Moodle. This includes lecture slides, programming assignments, practice tests, and example programs.
- You are required to take an exam at the specified time regardless of Moodle accessibility. In other words, keep a copy of the practice tests, notes, etc. on YOUR computer or network folder in case you have issues logging onto Moodle.

Prerequisites:

- None. This course does not assume any previous programming or computer science experience. You are expected to have a reasonable high-school mathematics background to appreciate the use of the mathematical notation.

Course Description:

- CS 141 is a required course for computer science majors and should be taken during the first year. It is the first course in the sequence for majors and offers an introduction to the fundamental principles of programming, abstraction, and design.

- This course is aimed at helping students acquire the reasoning and abstraction skills needed for designing algorithms and programs. This course teaches you how to think as a computer scientist, by teaching the process of building abstractions to hide implementation details, and of controlling the intellectual complexity of designing large software systems by decomposing problems into simpler sub-problems.

- This course will use the Python programming language as the vehicle for exploration of fundamental computer science concepts. However, this is not a course about Python; it is about the structure and interpretation of computer programs.

- The particular Python environment that will be used in this course is available in the computer labs on Rhodes College campus. Check the postings at the Barret 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. 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.

Course Objectives:

- At the end of this course, you should be able to:
    - Use good programming style in the programs you write
    - Apply your problem-solving skills and knowledge of data structures to a wide variety of computational problems
    - Explain and use basic principles of program design, such as abstraction to hide implementation details, and decomposition to control the intellectual complexity of the problem
    - Understand and modify code written by somebody else
    - Design, implement, and test a program that solves a given problem, and have some basic understanding of the computational complexity of your program

Course Outline:

- The course will cover the following topics (not necessarily in this order):
    - Program design
    - Testing and debugging
    - Variables, expressions and statements
    - Functions and problem decomposition
    - Fundamental data structures
    - Loops and Iteration
    - Sequential structures
    - File reading and writing
    - Bubble Sort

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. 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.**
- 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 that represents the results of your own effort.

Programming Assignments:

- All programs assigned in this course must be written in Python.
- 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 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.
- **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.

Programming Style:

- Programming is not a dry mechanical process but an art form. Well-written code has an aesthetic appeal while poor form can make other programmers and instructors cringe. Programming assignments will be graded based on correctness and style. Good programming practices usually include many of the following principles:

    - A comment at the top of the program that includes
        - Program authors
        - Date or Dates
        - A brief description of what the program does
    - Concise comments that summarize major sections of your code
    - Meaningful variable and function names
    - Well organized code
    - White space or comments to improve legibility
    - Avoidance of large blocks of copy-pasted code

Exams:

- There will be two midterms and one final exam:
    - **Midterm 1:** Wednesday, October 18, in class.
    - **Midterm 2:** Friday, November 21, in class.
    - **Final Exam:** Monday, December 12, 8:30 am **OR** Tuesday, December 13, 5:30 pm
- Make-up exams will only be given in extreme circumstances.

Grade Breakdown:

- 42.5 % Programming Assignments*
- 35.0 % Midterms
- 22.5 % 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.
- Grades will be posted on Moodle.

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. You are responsible for signing the attendance sheet.
- 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: 9/3/2008
- Extended Drop Period ends: 9/17/2008
- Pass Fail Period Ends: 9/17/2008
- Withdrawal Period Ends: 10/31/2008

I reserve the right to alter this syllabus as necessary.