Clemson University, Computer Science Division

CpSc 8700, Object Oriented Software Development Policies and Information

Brian Malloy, PhD

May 17, 2018

Office: 313 McAdams Hall	Internet: www.brianmalloy.com
Office Hours: by Appointment	e-mail: malloy@clemson.edu

1. There is no official textbook for the course but you may wish to purchase a Python book. I will post materials on my webpage and in the repository for the course. I will post directions and a video describing access to the repository.

2. Tentative schedule and grading policy ($\pm$ 10%):

Worksheets	40%	TBA
Projects	50%	TBA
Final Project	10%	TBA

3. *Grading*: Any questions about grading must be submitted through email within two (2) weekdays of the day that the graded material is returned.

4. *Attendance*: There are no face-to-face lectures but there will be videos posted and you should plan to watch these actively each day of the week, but not on weekends.

5. *Course objective*: To learn how to apply software engineering techniques to software development. This will entail exercise of the entire software life cycle, including: (1) requirements specification, (2) software design, (3) incremental software development, (4) software testing, and (5) maintenance.

6. *Policy and Syllabus*: Policy & syllabus can be found at: `www.brianmalloy.com`

7. *Venue*: The course meets every day of the week, not weekends.

8. *Programming Projects*: There will be six to eight (6–10) programming projects and these will be written in Python, using Tkinter, and perhaps other technology, such as XML.

9. *Grades*: The grading scale is: 90 or better is an A, 80 or better is a B.

10. *Academic Integrity*: "As members of the Clemson University community, we have inherited Thomas Green Clemson's vision of this institution as a 'high seminary of learning.' Fundamental to this vision is a mutual commitment to truthfulness, honor, and responsibility, without which we cannot earn the trust and respect of others. Furthermore, we recognize that academic dishonesty detracts from the value of a Clemson degree. Therefore, we shall not tolerate lying, cheating, or stealing in any form. In instances where academic standards may have been compromised, Clemson University has a responsibility to respond appropriately to charges of violations of academic integrity."

11. *Accomodation for Student Disability*: Student Disability Services coordinates the provision of reasonable accommodations for students with physical, emotional, or learning disabilities. Accommodations are individualized, flexible, and confidential and are based on the nature of the disability and the academic environment, in compliance with Section 504 of the Rehabilitation Act of 1973 and the Americans with Disabilities Act of 1990. Students are encouraged to consult with the Disability Services staff early in the semester, preferably prior to the first day of class. Current documentation of a specific disability from a licensed professional is needed.

12. *Posting of grades*: The United States Family Educational Rights and Privacy Act prohibit the public distribution of grades or graded work. Public distribution is commonly understood to include posting grades by student names, initials, or student social security number. It also is understood to include placing of graded material in a public place where students access the material to find their own graded work.

13. *Sexual Harrassment*: Clemson University is committed to a policy of equal opportunity for all persons and does not discriminate on the basis of race, color, religion, sex, sexual orientation, gender, pregnancy, national origin, age, disability, veterans status, genetic information or protected activity in employment, educational programs and activities, admissions and financial aid. This includes a prohibition against sexual harassment and sexual violence as mandated by Title IX of the Education Amendments of 1972.

The goal is for each of you to acquire an understanding and a working knowledge of object-oriented software development using a *rapid prototyping language*, design patterns, XML, GUI development, and testing techniques. To achieve this goal, each of you will be a client in this course and, as such, you will specify the requirements for a major application that, hopefully, will be useful to you now and in the future and that you will want to maintain. The problem with most academic programming assignments is that the student develops a dixie-cup solution to a problem assigned by the instructor: you write it once, get the grade, archive it for your friends who may take the course next year, and never look at it again.

The scenario for actual software applications is antithetical to the academic scenario, where working applications are maintained, and patched for many years. This course offers you an opportunity to write an application that you can actually use, and that you will likely maintain, extend, and modify. The important features that you will incorporate into your application are that it satisfies your requirements, it is robust, easy to maintain, and easy to extend. To achieve robustness we will use the **unittest** framework to test classes. To achieve ease of maintenance, we'll use good design, including liberal use of design patterns.

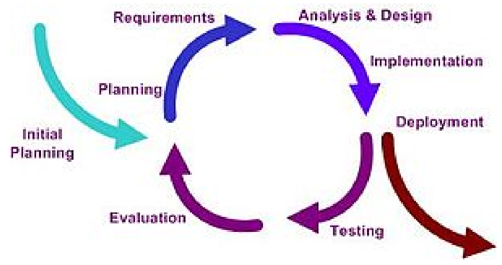

Figure 1: Iterative Development

We will use an *iterative and incremental* approach to development. To achieve this, we will begin by implementing some core functionality that "works," and then extend this core through the 5 1/2 weeks of the course. Each of the programming projects will focus on some aspect of the iterative development of your application, beginning with specification of the requirements, then some design, followed by an implementation of some core functionality, then extending the core, until the application is suitable for submission as a final project. Your application must provide a graphical interface, use XML to maintain state across executions, include test cases that use the **unittest** framework, and provides most of the functionality that you specified in your original requirements document.