Software Development Methods

CS 2110 – Example Syllabus

Course Description

If you take a moment to think about all the ways computerization has penetrated our daily lives, from having the weather forecast at your fingertips to life and death decision making, the importance of writing precise and correct computer instruction (code) is self-evident.

Who Should Take This Course?

This is a second course in computing and software development, with an emphasis both on software development concepts and on principles central to computer science. To achieve this, the course has a number of objectives:

(a) understand and implement object-oriented programming principles;
(b) be introduced to the way software systems are designed, implemented, tested, and maintained;
(c) apply best practices in terms of system testing to save time, minimize cost, and avoid hardships;
(d) expand on existing experience with the art of computer science;
(e) provide practical experience in software engineering and to increase your skill as a programmer.

Prerequisites:

To be successful, students should:

- Have the equivalent of one semester of programming knowledge (specific language does not matter), as demonstrated by any of the following:
- Have taken a CS 1 course with a C- or better.
- Have credit on your transcript for an equivalent course from high school or another university.

If you feel you have *not* met these prerequisites, please contact the instructor immediately.

Evaluating Your Progress in the Course

Participation (15%): Studies show the more you engage the more you will learn. In order to achieve this, everyone should be participating in activities, peer collaborations, and discussions both in and out of class. Further, speaking with peers will result in different points of views and ideas to come to the surface. Participation will chiefly be measured in the form of peer collaborations through various in-class activities. For most in-class activities you will be expected to turn your work in individually. Three (3) in-class activities may be missed without penalty. This is intended to cover health issues, school conflicts, school-related travel, and other circumstances.

For most classes there is an assigned reading (see **Course Schedule** on Collab Home page). We expect you to have completed this reading **before** class. We will conduct lecture under the assumption that you have completed this reading. This is intended to get you familiar with the topics, but we don't expect you to have mastered the material at this time. Some lecture topics will not be part of any of the course texts.

Labs (10%): Activities in lab provide supplementary hands-on experience with concepts, skills, and problem solving relating to topics from the current material, including material not covered in lecture. Given the group setting, labs are a perfect way to learn from your peers and discuss the material. You also have the opportunity to discuss your lab work with the TAs present who can provide additional feedback and support. Unlike homework assignments, lab activities allow you to practice skills in a group setting and, in some cases, explore certain topics deeper. Attendance in lab is mandatory and will be taken directly. Arriving late, leaving early, or not participating while present may result in a reduction of lab grade.

One (1) lab may be missed without penalty. This is intended to cover health issues, school conflicts, school-related travel, and other circumstances. If you know that you will miss a lab but could meet at another lab time, you may contact the head TAs of both labs to see if attending the other lab is acceptable. Abuse of this policy will result in your request being denied and possibly a lowering of your lab grade.

Homework Assignments (25%): Homework assignments help you practice coding, apply concepts and skills, and develop your computer science and software development skills. There will be around eight homework assignments. The assignments will be a combination of programming problems and paper-and-pencil activities that address the topics studied. These assignments provide you hands-on experience with course material. All coding assignments are individual assignments. Coding questions do show up on exams and possibly in other contexts, so the more practice you have the stronger your problem solving and programming skills will be!

Homework assignments will not be handed out in class; they will be made available online, and will be submitted online. Each assignment will describe its grading criteria. Resources available to you for completing homework assignments are: Piazza, instructors (asking in class or during office hours), TAs, textbooks, and official Java APIs (Oracle website). Homework assignments are due by **11:30 PM** on the day they are due, and may be submitted up to 48 hours late, with 10% off if submitted by the submission time the day after it was due and 20% off if submitted two days after it was due. Homework will not be accepted 48 hours after the deadline. Homework submission details will be provided within each assignment. No homework assignments will be dropped.

Quizzes (10%): There will be a number of quizzes given throughout the course, typically one each week, to ensure you are understanding the main ideas of the topic at hand. All quizzes are accessed through the Quizzes tab on Collab. Quizzes will typically be short and will mainly be multiple choice questions. Quizzes will be posted on Friday at 5:00 PM and be due the following Sunday by **11:30 PM** unless otherwise noted (it's the student's responsibility to keep up with this each week). The equivalent of one quiz will be excused.

Exams (Exam 1, 2, Final Exam) **(10% for each midterm, 20% for the final)**: There will be three exams. These exams are designed to assess the content knowledge, skills and concepts you learn during the semesters. Exam 1 will cover the first third of the course, Exam 2 will cover the second third of the course, and the Final Exam will mainly cover the remainder of the course, with some material from the first two-thirds. Each exam will consist of a mix of questions, and may include: multiple-choice, coding, short-answer, compare-contrast, applied concept, and true-false questions.

Course Resources

A textbook is a great resource that supplements the classroom experience. The **two** primary textbooks for the course are listed below, along with other suggested resources:

1. [Primary] *BIG JAVA Early Objects, Sixth Edition* by Cay Horstmann
2. [Primary] *Modern Software Development Using Java, Second Edition* by Paul Tymann and G. Michael Schneider. This text is freely available online at http://www.cs.rit.edu/~ptt/msd/. Affordable print copies are hard to find. **Don't use the First edition**.
3. You should own or have access to a good **Java reference book**. If you don't you might try an online source such as:
    - *Java Programming* (a wikibook)
    - *Thinking in Java* by Bruce Eckel;
    - *Introduction to Computer Science using Java* by Bradley Kjell; or
    - Fred Swartz's topic-based review.
4. **CodingBat** (`http://codingbat.com/java`) is a free online website to help you build coding skill in Java (Python, too!) Going through many practice problems is a great way to practice and solidify your understanding of coding concepts. We will assign specific problems from this site early on in the semester.

If you find other good online references***, let us know!***
Other readings may be assigned from the Web or provided by the instructors.

CS 2110 - Example Course Schedule

WK	Day	Date	Topic for class	Homework OUT	Homework DUE
1	Wed	28-Aug	Intro		
	Fri	30-Aug	Data Types		
2	Mon	2-Sep	Data Types Part 2		
	Wed	4-Sep	Control Structures, Arrays		
	Fri	6-Sep	Methods, Pass-by-referece, methods calling other methods, `main()` method testing	HW 0	
3	Mon	9-Sep	ArrayList, compare/contrast with arrays		
	Wed	11-Sep	Classes, State and behavior, UML, getters, setters, and constructors		
	Fri	13-Sep	Classes with other classes as fields, input validation, `.toString()` method	**Sprint 1**	HW 0
4	Mon	16-Sep	Intro to class hierarcies, Object class, `.equals()` method vs. ==, `contains()`		
	Wed	18-Sep	Inheritance 2, Lists and Sets		
	Fri	20-Sep	Sets and Maps	**Sprint 2**	**Sprint 1**
5	Mon	23-Sep	Unit Testing, Test Driven Development (TDD)		
	Wed	25-Sep	Unit Testing		
	Fri	**27-Sep**	**Exam Review**		**Sprint 2**
6	**Mon**	**30-Sep**	**Exam 1**		
	Wed	2-Oct	Software development life cycle, requirements and how they tie to testing		
	Fri	4-Oct	Software development life cycle 2		
7	***Mon***	***7-Oct***	***No Class ~ Fall Break***		
	Wed	9-Oct	Inheritance, abstraction, polymorphism, abstract classes		
	Fri	11-Oct	Inheritance, abstraction, polymorphism, abstract classes		
8	Mon	14-Oct	Intro to Interfaces	**Sprint 3**	
	Wed	16-Oct	Collections framework revisited, Comparables & Comparators		
	Fri	18-Oct	Stacks and Queues		
9	Mon	21-Oct	Event Driven		
	Wed	23-Oct	Event Driven	**Sprint 4**	**Sprint 3**
	Fri	25-Oct	Exception Handling		
10	Mon	28-Oct	Intro to Algorithms		
	Wed	30-Oct	Intro to Algorithms		
	Fri	1-Nov	Searching and Sorting		**Sprint 4**
11	**Mon**	**4-Nov**	**Exam Review**		
	Wed	**6-Nov**	**Exam 2**		
	Fri	8-Nov	Creating and Analyzing Algorithms		
12	Mon	11-Nov	Concurrency		
	Wed	13-Nov	Concurrency continued	HW 5	
	Fri	15-Nov	Intro to recursion		
13	Mon	18-Nov	Recursion continued		
	Wed	20-Nov	Divide and Conquer		
	Fri	22-Nov	Intro to trees	HW 6	HW 5
14	Mon	25-Nov	Tree traversals		
	Wed	27-Nov	***No Class ~ Thanksgiving Break***		
	Fri	29-Nov	***No Class ~ Thanksgiving Break***		
15	Mon	2-Dec	Heaps		
	Wed	4-Dec	Binary search Trees		
	Fri	**6-Dec**	**Final Exam Review**		HW 6