CSCI 201L Syllabus
Principles of Software Development
Fall 2017

Instructor: Jeffrey Miller, Ph.D.
Email: jeffrey.miller@usc.edu
Web Page: http://www-scf.usc.edu/~csci201
Office: SAL 342
Phone: 213-740-7129
Lectures: 30303R, Tuesday/Thursday 8:00a.m.-9:20a.m., SSL 150
29909R, Tuesday/Thursday 9:30a.m.-10:50a.m., GFS 116
30389R, Tuesday/Thursday 11:00a.m.-12:20p.m., LVL 17
Quiz: 30028R, Thursday, 7:00p.m.-8:50p.m.
Labs: 30237R, Tuesday, 2:00p.m.-3:50p.m., SAL 126
30134R, Tuesday, 4:00p.m.-5:50p.m., SAL 126
30241R, Tuesday, 6:00p.m.-7:50p.m., SAL 109
30396R, Tuesday, 6:00p.m.-7:50p.m., SAL 126
30239R, Wednesday, 10:00a.m.-11:50a.m., SAL 126
30238R, Wednesday, 12:00p.m.-1:50p.m., SAL 127
30385R, Wednesday, 4:00p.m.-5:50p.m., SAL 109
29904R, Wednesday, 6:00p.m.-7:50p.m., SAL 109

Office Hours: Tuesday 12:30p.m.-1:30p.m.
Thursday 1:30p.m.-3:45p.m.
Any day by appointment

Textbooks: Liang, Y. Daniel. Introduction to Java Programming, Comprehensive Version, 10th Edition, Prentice Hall, Inc., 2014. ISBN 978-0133761313

Description: Object-oriented paradigm for programming-in-the-large in Java; writing sophisticated concurrent applications with animation and graphic user interfaces; using professional tools on team project.

Grades:

Labs	10%	Assignments	20%
Group Project	30%	Exams	40%
Written Exam #1	15%	Written Exam #2	15%
Programming Exam	10%		

Grades will be based on a curve that operates in favor of the students, with at least the following grades for a given percentage x. If the average in the class is lower than 80%, the average will become the cut-off between a B- and a C+.

x >= 93	A	73 <= x < 77	C
90 <= x < 93	A-	70 <= x < 73	C-
87 <= x < 90	B+	67 <= x < 70	D+
83 <= x < 87	B	63 <= x < 67	D
80 <= x < 83	B-	60 <= x < 63	D-
77 <= x < 80	C+	x < 60	F

Lecture Schedule: Chapter references are from Y. Daniel Liang, Introduction to Java Programming, Comprehensive Version, 10th Edition, Prentice Hall, 2014. ISBN 978-0133761313

Week	Lecture	Date	Lecture Topic	Lab Topic	Chapter
1	1	August 22, 2017	Introduction, Environment, Methods	Github Tutorial, Environment Setup	1-8
	2	August 24, 2017	Classes, Packages, File I/O, Abstract Classes and Interfaces		9-10, 12-13
2		August 29, 2017	Inheritance, Polymorphism	File I/O	
	3	August 31, 2017	Garbage Collection, Exception Handling, Serialization, Generics, Inner Classes		11
3	4	September 5, 2017	HTML, CSS	Inheritance	12, 19
	5	September 7, 2017	Java Servlets, JSP		
4	6	September 12, 2017	JavaScript	HTML, CSS	37-38
	7	September 14, 2017	AJAX		
5	8	September 19, 2017	Concurrent Computing	JSP and Servlets	42
	9	September 21, 2017	Thread Methods, Thread Pools, Thread Priorities		30
6	10	September 26, 2017	Software Engineering, Methodologies, Testing	JavaScript and AJAX	30
	11	September 28, 2017	Project Description, Review for Exam		
7	12	October 3, 2017	Networking Theory	Software Engineering	
	13	October 5, 2017	Network Programming		31
8	14	October 10, 2017	Multi-Threaded Network Programming	Threads	31
	15	October 12, 2017	Network Serialization		31
		October 12, 2017 7:00p.m.-8:50p.m.	**Written Exam #1**		
9	16	October 17, 2017	Web Sockets	MySQL Installation	31
	17	October 19, 2017	Databases		32, 35
10	18	October 24, 2017	SQL	JDBC	32, 35
	19	October 26, 2017	JDBC		32, 35
11	20	October 31, 2017	Concurrency, Monitors	Networking Worksheet	30
	21	November 2, 2017	Locks, Conditions, Producer/Consumer		30
12	22	November 7, 2017	Sleeping Barber, Locks Programming	Web Server	30
	23	November 9, 2017	Multi-Threaded Programming Design, Semaphores		30
		November 9, 2017 7:00p.m.-8:50p.m.	**Programming Exam**		
13	24	November 14, 2017	Parallel Computing	Locks and Monitors	30
	25	November 16, 2017	Distributed Computing, RMI		39
14	26	November 21, 2017	Project Demonstrations	Sleeping Barber	
		November 23, 2017	*No Class - Thanksgiving*		
15	27	November 28, 2017	Project Demonstrations	Parallel Computing	
	28	November 30, 2017	Project Demonstrations		
16		**December 11, 2017 4:30p.m.-6:30p.m.**	**Written Exam #2**		

Exams:	Written Exam #1	Thursday	October 12, 2017	7:00p.m.-8:50p.m.
	Programming Exam	Thursday	November 9, 2017	7:00p.m.-8:50p.m.
	Written Exam #2	Monday	December 11, 2017	4:30p.m.-6:30p.m.

The written exams are closed book and will consist of theoretical questions and may have code to be analyzed, though very little code will be required to be written. The programming exams are open book and open Internet, though no other people can be used (i.e. no posting on discussion boards, email, chatting, texting, etc.). You will need to write a program that compiles based on certain specifications, similar to assignments, though adjusted based on time constraints.

An exam can only be taken on the scheduled date and at the scheduled starting time. Accommodations for students with letters from DSP will be provided, though the exam will still need to be taken on the scheduled date. There are no makeup exams. If you miss an exam due to an emergency, official written documentation, whatever that may be based on the situation, will need to be submitted to me as soon as you are physically able (before the exam if able). Approval will be solely based on my discretion though it should be based on a documented illness or emergency. Based on the exam, here are the rules that will be followed:

- If an excuse is not approved, you will be given a 0 on the exam.
- If there is an approved excuse for written exam #1, the percentage for that exam will be added to the percentage for written exam #2.
- If there is an approved excuse for the programming exam, the percentage for that exam will be added to the percentage for written exam #2.
- If there is an approved excuse for written exam #2, you will receive an Incomplete grade in the course and have to make up the exam based on the conditions of an Incomplete.

Assignments: Assignments will be discussed in class and worked on individually. Discussion among students is fine, but no copying of other student's code is allowed. The program needs to compile, and grading will only occur if the program is able to be run. Grading criteria will be provided at the time the assignment is assigned. Graders will grade the assignments. If any questions arise based on the grade on the assignment, students should first contact the grader who graded the assignment. If a satisfactory resolution is not reached, you should next contact the TA. If a satisfactory resolution is still not reached, then come to the professor. Assignments will be submitted via Blackboard and are due by 11:59p.m. on the due date (see Late Policy below).

Project: The project in the class will be assigned approximately half-way through the semester. The project will consist of between 4-6 students. Formal documentation following the software engineering process will be required. The project will be discussed in class with the corresponding due dates. The project deliverables will be submitted via Blackboard and is due by 11:59p.m. on the due date (see Late Policy below).

Late Policy: There is no late policy. In extenuating circumstances, students may be allowed to submit an assignment late, but only if approved by the professor. This typically should be done before the due date, though I understand some situations may not allow this. For any assignment or project that is submitted after 11:59p.m. on the due date, the student will receive a 0.

Labs: The TA/CPs will lead the lab section each week. There will be an assigned lab program each week that reinforces the topics covered in the lectures for the week. The lab assignments will be graded based on effort and attendance. You should NOT complete the lab before your assigned lab section. The labs are intended to be completed during the lab period, and you are expected to work on the lab during the section. The lab assistants have been instructed NOT to grade any labs until after half of the lab has elapsed. You also must show up to the lab within the first 10 minutes or you will not be allowed to join the lab that week. Solutions will be posted to the labs, but you should make sure you are using the lab periods to learn the material that has been covered. The lab assistants are there to answer any questions and help you, so use your time in lab wisely. Each lab is worth 0.75% of the final grade, and the total lab score is out of 10%. With 15 labs, that gives a total of 11.25%. That means that you are able to miss one lab without penalty to your final grade, or you can earn 1.25% extra credit if you get full credit on all of the labs.

Prerequisites: CSCI 104L – Data Structures and Object Oriented Design

Students with Disabilities: Any student requesting academic accommodations based on a disability is required to register with Disability Services and Programs (DSP) each semester. A letter of verification for approved accommodations can be obtained from DSP. Please be sure the letter is delivered to the professor as early in the semester as possible.

Academic Integrity: The Viterbi School of Engineering's policy on Academic Integrity can be found at http://viterbi.usc.edu/academics/integrity/. All students are expected to understand and abide by these principles. SCampus (http://scampus.usc.edu), the Student Conduct Code, contains the information about violating University standards in Section 11.00. Any potential violations will be taken seriously and the proper academic process will be followed, including reporting to the USC Student Judicial Affairs and Community Standards (SJACS).

Viterbi Honor Code: Engineering enables and empowers our ambitions and is integral to our identities. In the Viterbi community, accountability is reflected in all our endeavors.

Engineering+ Integrity.
Engineering+ Responsibility.
Engineering+ Community.
Think good. Do better. Be great.

These are the pillars we stand upon as we address the challenges of society and enrich lives.

Course Outcomes (expected after you've finished the course)

i	The ability to understand the software engineering in terms of requirements, design, and implementation;
ii	An understanding of how to use interaction diagrams to help define requirements;
iii	The ability to produce a software design based on requirements;
iv	The ability to produce software, including graphical user interfaces, from a design;
v	The ability to unit test a module;
vi	An understanding of concurrency and how it works in computer operating systems;
vii	The ability to write multi-threaded programs and correctly solve a mutual exclusion problems using semaphores or monitors;
viii	The ability to use Java in writing programs;
ix	The ability to use HTML and CSS in designing graphical user interfaces;
x	The ability to use messaging as a communication method;
xii	The ability to apply a software engineering process to a large software project;
xiii	The ability to work effectively on a team;
xiv	An understanding of the ethical issues in working within a group;

ABET Program Outcomes (after you have finished your degree)

a	An ability to apply knowledge of computing and mathematics appropriate to the discipline;
b	An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution;
c	An ability to design, implement and evaluate a computer-based system, process, component, or program to meet desired needs;
d	An ability to function effectively on teams to accomplish a common goal;
e	An understanding of professional, ethical, legal, security, and social issues and responsibilities;
f	An ability to communicate effectively with a range of audiences;
g	An ability to analyze the local and global impact of computing on individuals, organizations and society;
h	Recognition of the need for, and an ability to engage in, continuing professional development;
i	An ability to use current techniques, skills, and tools necessary for computing practices.
j	An ability to apply mathematical foundations, algorithmic principles, and computer science theory in the modeling and design of computer-based systems in a way that demonstrates comprehension of the tradeoffs involved in design choices;
k	An ability to apply design and development principles in the construction of software systems of varying complexity.