CMSI 387-01

OPERATING SYSTEMS

http://myweb.lmu.edu/dondi/spring2013/cmsi387

Spring 2013 — Pereira 207
TR 1:35–2:50pm, 3 semester hours
Office Hours: TR 10am–12pm, or by appointment

John David N. Dionisio, PhD
email: *dondi@lmu.edu*
Doolan 106; (310) 338-5782

Objectives and Outcomes

This course explores the computer science subfield of *operating systems*. A computer's operating system is its "program that runs all other programs"—it is a very special type of program, however, and so it is grounded in common concepts, theories, and algorithms. Long after the course concludes, my hope is that you will be able to:

1. **Perform operating system tasks that are typically viewed as "power user" activities.**
2. **Implement common operating system functionalities and algorithms.**
3. **Demonstrate genre literacy within the operating system field.**

In addition to the course-specific content, you are also expected to:

4. **Follow academic and technical best practices throughout the course.**

Prerequisites/Prior Background

Programming proficiency in a systems-level language, particularly C; a prior course in computer system organization (LMU CMSI 284 or equivalent). Familiarities with Java, shell scripting, and system administration are also beneficial.

Materials and Texts

- Abraham Silberschatz, Peter Baer Galvin, and Greg Gagne, *Operating System Concepts*, 8th edition, John Wiley & Sons, 2008 — *a.k.a. "SGG"*
- Daniel Pierre Bovet and Marco Cesati, *Understanding the Linux Kernel*, Third Edition, O'Reilly Media, 2005
- Jonathan Corbet, Alessandro Rubini, and Greg Kroah-Hartman, *Linux Device Drivers*, Third Edition, O'Reilly Media, 2005
- Assorted handouts, articles, and sample code to be distributed throughout the semester

In addition, do not hesitate to look for further information regarding the concepts, techniques, tools, and paradigms that we will discuss.

Course Work and Grading

This course uses standards-based grading: your proficiency in each course objective is directly evaluated according to the outcomes shown on page 4 of this syllabus. Proficiency is measured according to the following key:

+	Advanced proficiency
|	Appropriate proficiency
/	Approaching appropriate proficiency
–	Needs practice and support
O	No basis for evaluation

Your submitted work is used to evaluate these outcomes (see below). Letter grades are then assigned as follows:

	+	|	/	–
A	**many**		*none*	*none*
B		**many**	**few**	*none*
C			**some**	**few**
D				**some**
F				**many**

A–, B+, B–, C+, and C– grades may be assigned based on "close calls" along the proficiency measure thresholds and qualitative considerations such as degree of difficulty, effort, class participation, time constraints, and overall attitude throughout the course. You may inquire at any time about the proficiency measures that I currently have on record for you.

Homework

Homework consists of questions, exercises, and programming assignments to be given throughout the semester. Homework is one mechanism for demonstrating the proficiencies expected of the course. You will be given feedback on these proficiencies, and *may resubmit your homework throughout the semester* in order to improve upon them.

With great flexibility comes great accountability. First off, *you must submit your homework on time*. The assignment due date is encoded in the homework number. Late homework detracts from outcome *4f* (*Meet all designated deadlines*).

Quizzes and Tests

Some outcomes are best demonstrated by answering questions or doing exercises in class. These resemble traditional quizzes and tests, but, like homework, they are evaluated according to standards and do not produce a numerical score. *They are typically spontaneous and unannounced.*

Questions may include content-oriented elements as well as forward-looking, applicative portions (i.e., "use this knowledge to resolve this situation"). Tests are open-paper-everything; no sharing. "Open computer" might be allowed depending on the circumstances. You may neither solicit nor give help during a quiz or test. Late or missed tests are handled case-to-case; in all instances, talk to me.

Term Portfolio

Your accumulated homework and tests for the semester comprise the *term portfolio*—the final, definitive artifact that demonstrates the proficiencies you have reached for each course outcome. The term portfolio provides you with an opportunity to polish the work done throughout the semester; it is how you show that you learned from your mistakes or improved on already established knowledge.

Throughout the semester, you may improve your work based on received feedback and show it to me for re-evaluation. Improvements in proficiency are recorded and give you a good idea of how your term portfolio will fare long before its final version is submitted.

The final version of your term portfolio is due on May 10. Late portfolios detract from outcome *4f*.

Version Control

Version control is an indispensable part of today's computer science landscape in industry, the academe, and the open source community. We use version control heavily in this course: make sure that you get the hang of it.

Attendance

Attendance at all sessions is expected, but not absolutely required. If you must miss class, it is your responsibility to keep up with the course. The last day to add or drop a class without a grade of W is January 18. The withdrawal or credit/no-credit deadline is March 22.

University Policy on Academic Honesty

Loyola Marymount University expects high standards of honesty and integrity from all members of its community. All students are expected to follow the LMU Honor Code and Process, as stated in the *LMU Undergraduate Bulletin*.

Americans with Disabilities Act

Students with special needs as addressed by the Americans with Disabilities Act who need reasonable modifications, special assistance, or accommodations in this course should promptly direct their request to the Disability Support Services (DSS) Office. Any student who currently has a documented disability (physical, learning, or psychological) needing academic accommodations should contact DSS (Daum 224, x84535) as early in the semester as possible. All discussions will remain confidential. Please visit *http://www.lmu.edu/dss* for additional information.

Topics and Important Dates

Correlated outcomes are shown for each topic. Specifics may change as the course progresses. University dates (italicized) are less likely to change.

January	Operating systems overview; operating system "power use;" networking and I/O *(1a–1e; 3a)*
January 18	*Last day to add or drop a class without a grade of W*
February	Booting, installation, and the kernel *(2a, 2b)*; process management *(1b, 2c, 2d)*
March	Scheduling, synchronization, and deadlocks *(2d, 3b)*; memory management *(1b, 2d, 3b)*
March 4–8	*Spring break; no class*
March 22	*Withdraw/credit/no-credit deadline*
March 27–29	*Easter break; no class*
April	Mass storage and file systems *(1c, 2e)*; portfolio workshops *(1a–3b)*
May 10	*Term portfolios due*

You can view my class calendar and office hour schedule in any iCalendar-savvy client. Its subscription link can be found on the course web site (it's too long to provide in writing).

If necessary, this syllabus and its contents are subject to revision. Students are responsible for any changes or modifications announced in class.

Course Outcomes

1 Perform operating system tasks that are typically viewed as "power user" activities.

1a	*Effectively and efficiently use a command-based operating system shell.*	At a minimum, operating system "power users" must know how to "get around," configure things according to their needs, and have a clear, unabridged handle on the state of a running computer. These outcomes capture these basic abilities.
1b	*Monitor and manage processes and memory use.*	
1c	*Navigate, monitor, and manage a logical file system.*	The ability to do this implies knowledge of file system concepts such as directories, permissions, and attributes, to name a few.
1d	*Redirect input and output streams to and from files, processes, and networked computers.*	This includes not only the traditional activities of reads, writes, and pipes, but also rerouting network connections and services.
1e	*Interact with operating systems across the network.*	e.g., remote login (command or GUI), file copying, etc.

2 Implement common operating system functionalities and algorithms.

2a	*Build and deploy an operating system kernel.*	These outcomes map almost directly to specific assignments throughout the semester, most likely to be given in this order. Examples of the operating system scenarios and algorithms mentioned in outcome *2d* include: process creation, scheduling, synchronization, deadlock detection and avoidance, main and virtual memory management, etc.
2b	*Define, implement, and invoke a new system call.*	
2c	*Write a simple operating system shell.*	
2d	*Simulate or implement standalone demonstrations of operating system scenarios and algorithms.*	
2e	*Create a virtual disk and navigate it at the byte level.*	

3 Demonstrate genre literacy within the operating system field.

3a	*Identify and perform equivalent tasks across distinct operating system platforms.*	a.k.a. knowing your way around, or at least asking the right questions, on any modern operating system
3b	*State and describe seminal personalities and milestones from the field's history.*	Operating systems is one of the oldest subfields in computer science, and you should some knowledge of this legacy.

4 Follow academic and technical best practices throughout the course.

4a	*Write syntactically correct, functional code.*	Code has to compile. Code has to work. No errors, no bugs. Use unit tests as much as possible.
4b	*Demonstrate proper separation of concerns.*	This is the basis of good software design. It makes software easier to maintain, improve, and extend. Proper separation of concerns includes but is not limited to correct scoping of variables & functions and zero duplication of code.
4c	*Write code that is easily understood by programmers other than yourself.*	This outcome involves all aspects of code readability and clarity for human beings, including but not limited to documentation & comments, spacing & indentation, proper naming, and adherence to conventions or standards.
4d	*Use available resources and documentation to find required information.*	The need to look things up never goes away. Remember also that the course instructor counts as an "available resource," so this outcome includes asking questions and using office hours.
4e	*Use version control effectively.*	In addition to simply using version control correctly, effective use also involves appropriate commit frequency and descriptive commit messages.
4f	*Meet all designated deadlines.*	

Sample Standards Achievement Report

Based on these proficiencies, the student will get an A–.

1	**Perform operating system tasks that are typically viewed as "power user" activities.**	
1a	*Effectively and efficiently use a command-based operating system shell.*	+
1b	*Monitor and manage processes and memory use.*	+
1c	*Navigate, monitor, and manage a logical file system.*	|
1d	*Redirect input and output streams to and from files, processes, and networked computers.*	+
1e	*Interact with operating systems across the network.*	+
2	**Implement common operating system functionalities and algorithms.**	
2a	*Build and deploy an operating system kernel.*	|
2b	*Define, implement, and invoke a new system call.*	|
2c	*Write a simple operating system shell.*	|
2d	*Simulate or implement standalone demonstrations of operating system scenarios and algorithms.*	+
2e	*Create a virtual disk and navigate it at the byte level.*	|
3	**Demonstrate genre literacy within the operating system field.**	
3a	*Identify and perform equivalent tasks across distinct operating system platforms.*	|
3b	*State and describe seminal personalities and milestones from the field's history.*	|
4	**Follow academic and technical best practices throughout the course.**	
4a	*Write syntactically correct, functional code.*	+
4b	*Demonstrate proper separation of concerns.*	+
4c	*Write code that is easily understood by programmers other than yourself.*	+
4d	*Use available resources and documentation to find required information.*	+
4e	*Use version control effectively.*	+
4f	*Meet all designated deadlines.*	|