# CS 470G Introduction to Operating Systems

**University of Kentucky**
**Department of Computer Science**

Spring 2011

## Course Information

### Meeting Times

Tues/Thur 2:00 pm - 3:15 pm, Room RMB 323

### Instructor

Jim Griffioen, Hardymon Bldg 229, 257-6746, griff@netlab.uky.edu

### Office Hours

Tues/Thur 1:00 pm - 1:45 pm (Hardymon 229) or by appointment. You can also send questions via email. (griff@netlab.uky.edu)

### Course Objectives

This course is intended as a general introduction to operating systems. Topics that will be covered include kernel/microkernel designs, multiprogramming/multitasking, process/thread management, memory management, virtual machines, time management, synchronization, mutual exclusion, interprocess communication, device control, and file structures. The course will also briefly touch on operating system administration, networking, and distributed operating systems. The course will emphasize the various design possibilities and discuss the tradeoffs/consequences of these designs. Hands-on projects/labs will make the concepts real and expose you to the challenges of developing operating systems. The course assumes an understanding of machine organization, programming languages, and data structures.

### Course Prerequisites

Successful completion of the CS 315 and CS 380 courses and Engineering Standing are prerequisites for this course. **Advanced programming skills are required** (see http://www.cs.uky.edu/%7eraphael/programming.html). Ability to think abstractly about systems (e.g., concurrency, parallelism, synchronization) is necessary. Basic understanding of computer hardware is also assumed (e.g., CS/EE 380).

### Textbook (Required)

We will be using the Operating Systems Concepts textbook which comes in two versions. You must get one of the following versions:

- Operating Systems Concepts, Eighth Edition, Abraham Silberschatz, Peter Baer Galvin, and Greg Gange, John Wiley and Sons, Inc., ISBN 978-0-470-12872-5, **OR**
- Operating Systems Concepts: Essentials, Abraham Silberschatz, Peter Baer Galvin, and Greg Gange, John Wiley and Sons, Inc., ISBN 978-0-470-88920-6

## Exams

- **Exams:** There will be three exams:
  - Exam 1: Tuesday, February 22, 2pm
  - Exam 2: Tuesday, March 29, 2pm
  - Final Exam: Friday May 6, 8:00am, Scheduled Exam Slot
- **Labs/Projects/Homeworks:** Various hands-on lab and programming projects as well as homework assignments will be given throughout the course to reinforce the concepts and material discussed in class. Labs will be done during specified lab hours or via appointment. Programing projects will be in the C/C++ programming language on Unix systems. Students are expected to be fluent with C/C++. Programming in C/C++ will not be taught in this course. See http://www.cs.uky.edu/%7eraphael/programming.html for a list of programming skills students are expected to have. Unless specified otherwise, projects will be done individually by each student. Students must also read and adhere to good programming standards (see http://www.cs.uky.edu/%7eraphael/checklist.html for a checklist of good programming characteristics). Programs will be penalized for poor programming style.

## Grading

Grades will be determined based on performance on labs/homeworks/programming assignments (50%) and tests (50% = 20% exam 1 + 20% exam 2 + 10% final exam). Final grades will be assigned according to the following scale:

A=90-100%, B=80-89%, C=70-79%, D=60-69%, E=0-59%.

For graduate students, a D grade will automatically be replaced by E. No incomplete grades will be assigned unless there exist exceptional, extenuating circumstances.

## LATE PENALTY

Late programs or homeworks will be penalized 10% (of the total possible points) per day for each day late not including weekend or holiday days. Labs must be completed at the scheduled time.

## ACADEMIC CONDUCT

Expected academic conduct is defined in the university regulations which can be found at http://www.uky.edu/StudentAffairs/Code/part2.html. You are expected to have read these regulations and the procedures for dealing with violations. As stated in section 6.3 “All academic work, written or otherwise, submitted by students to their instructors or other academic supervisors, is expected to be the result of their own thought, research, or self-expression. In cases where students feel unsure about a question of plagiarism involving their work, they are obliged to consult their instructors on the matter before submission. When students submit work purporting to be their own, but which in any way borrows ideas, organization, wording or anything else from another source without appropriate acknowledgment of the fact, the students are guilty of plagiarism."

## ADDITIONAL REFERENCES

- *Operating Systems: Internals and Design Principles, 6e, William Stallings, Prentice Hall, 2009.*
- *Modern Operating Systems,3e, Andrew S. Tanenbaum, Prentice Hall, 2008.*
- *Operating Systems Design and Implementation, 3/e, Andrew S. Tanenbaum, Prentice Hall, 2006.*
- *Operating Systems Vade Mecum, Raphael Finkel, Prentice Hall, Second Edition, 1988.*
- *Operating System Design: The Xinu Approach, Douglas Comer, Prentice Hall, 1984*
- *The Linux Kernel Primer, Rodriguez, Fischer, and Smolski, Prentice Hall, 2006*
- *Structured Computer Organization,5/e, Andrew S. Tannenbaum, Prentice Hall, 2006*
- *The C Programming Language, Kernighan and Richie, Prentice Hall, Second Edition, 1988.*
- *The C++ Programming Language, B. Stroustrup, Addison Wesley, Third Edition, 1997.*
- *The Unix Programming Environment, Brian Kernighan and Rob Pike, Prentice-Hall, 1984.*

## Learning Outcomes

Students will acquire knowledge of the issues involved in the design and implementation of various components of an operating system and the interaction of the various components with hardware as well as their use in application development. More specifically, students will be able to

1. understand overall computer system structure and operating system design
2. apply the process management techniques and be able to write applications using the interfaces provided by the operating system.
3. apply the process synchronization mechanisms for writing concurrent applications
4. understand interfaces provided by the memory management component of the operating system
5. use the interfaces provided by the file system in developing applications.

6. understand methods used for protecting computer systems resources from unauthorized access.

Measures

The above outcomes will be evaluated based on the homeworks, labs, programming assignments, and exams which will focus on each of these outcomes. They will also be evaluated based on the students' self-assessment of their mastery of these outcomes performed at the end of the semester.

## MISCELLANEOUS

### Computer Accounts

You have been given accounts on the CS Multilab computers. If you do not know how to access the Multilab computers, please see me for instructions.

If you have accounts elsewhere on campus you can access the Multilab machines via ssh. **DO NOT telnet, rlogin, or ftp - they are insecure and may result in your password being stolen.** Be sure to use ssh, putty, or some program that uses encryption.

### Mailing List

Periodically I will post messages and announcements of general interest to the class (e.g., assignment changes, answers to specific questions). A mailing list for this class has been set up that will send email to your registered UK email account (usually yourlogin@uky.edu). You are responsible for reading the email sent to the list.

## Schedule

The following is a tentative outline of the material that will be covered each week. Readings will be assigned from the Silberschatz et. al. book and also articles placed on reserve in the Library.

(1) Course Overview, Introduction to Operating Systems
(2) Operating System Management: Install, Config, User/File/Software Mgmt
(3) Operating System Structure: Kernels, Modules, Virtualization, System Calls
(4) Process Management: Processes/Threads, Multiprogramming/Concurrency
(5) Process Management: Context Switching and Scheduling
(6) Process Management: Synchronization and Mutual Exclusion
(7) Memory Management: Physcial Memory, Allocation/Reclamation, Swapping
(8) Memory Management: Virtual Memory, Address Translation, Page Replacement
(9) Interprocess Communication: Shared Memory, Message Passing
(10) Interprocess Communication: Pipes, RPC, Sockets
(11) Input/Output: Device Hardware/Drivers, Interrupts/Polling
(12) Input/Output: Buffering/Caching, I/O API
(13) File Systems: File Structures, Naming

(14) File Systems: Access Control, Access Methods, Unix File Management
(15) Security: protection, access control, attacks, cryptography, authentication
(16) Advanced OS Topics: distributed/multimedia/realtime systems