

Prof: Dr. Peter M. Maurer

Office: Rogers 220.08

Phone: 710-7305

Email: Peter_Maurer@Baylor.edu

Web: <http://cs.ecs.baylor.edu/~maurer>

Text: Silberschatz, Galvin, Gagne

Operating System Concepts

Seventh (or Sixth) Edition

Office Hours: 9-10, 11-12 MWF, 8-12:00 TR

1. Aug 22	Introductions	
2. Aug 24	What and why is an operating system?	Chapter 1
3. Aug 26		
4. Aug 29	Hardware Organization	Chapter 2
5. Aug 31		
6. Sep 2		
7. Sep 5	Operating System Structure	Chapter 3
8. Sep 7		
9. Sep 9	Processes and Threads	Chapters 4 and 5
10. Sep 12		
11. Sep 14		
12. Sep 16	CPU Scheduling	Chapter 6
13. Sep 19	Exam Review	
14. Sep 21	Exam #1	
15. Sep 23		
16. Sep 26		
17. Sep 28	Process Synchronization	Chapter 7
18. Sep 30		
19. Oct 3		
20. Oct 5	Deadlocks	Chapter 8
21. Oct 7		
22. Oct 10	Memory Management	Chapter 9
23. Oct 12		
24. Oct 14	Virtual Memory	
25. Oct 17		Chapter 10
26. Oct 19		
27. Oct 21	Fall Break	
28. Oct 24	File Systems (in memory)	Chapter 11
29. Oct 26	Review for Exam #2	
30. Oct 28	Exam #2	
31. Oct 31		
32. Nov 2	File Systems (on disk)	Chapter 12
33. Nov 4		
34. Nov 7	I/O Systems and Mass-Storage Structure	Chapters 13 and 14
35. Nov 9		
36. Nov 11		
37. Nov 14	Distributed System Structures	Chapter 15
38. Nov 16		
39. Nov 18		
40. Nov 21		
41. Nov 23	Thanksgiving	
42. Nov 25	Thanksgiving	
43. Nov 28	Distributed File Systems	
44. Nov 30	Distributed Coordination	
45. Dec 2		
46. Dec 5	Review for Final	
Final Exam:	Monday Dec 12, 2:00-4:00 PM (Rogers 104)	

Course Objectives

By the time you have finished with this course, you should be familiar enough with the principles of operating systems that you could (at least theoretically) construct your own operating system for a new computer. To accomplish this goal, there are several smaller objectives that we must meet. These are as follows.

1. Learn where the boundary lies between hardware and software. When you interact with any computer system, part of the interaction is handled by the hardware (moving mouse-ball, depression of keyboard keys) and part of the interaction is handled by the software (displaying characters on the screen). We need to know where the boundary lies so that we know what is available in the hardware, and what must be implemented in software.
2. Learn how operating system code gets executed. Despite the sophistication of today's computers, for the most part they still execute only one instruction at a time. That means when a user program is executing, the operating system must be idle. One of the most important things you will learn is the mechanisms that are used to give the operating system its "turn to execute."
3. Learn the types of tasks that are normally relegated to an operating system. Some things that seem to be part of the operating system are not. For example the LINUX shell is not part of the LINUX operating system.
4. Learn the most common and popular algorithms and data structures for performing standard operating system tasks. If an operating system must perform a particular task for virtually every program (memory allocation is an example) then you should know the most common methods for performing the task.

Grading

Final Exam: 35%

Projects and homework: 15%

Other Exams: 50% -- Equally divided among all exams.

Other Information

Exam grades will be curved, if necessary – but it probably won't be necessary.

University attendance policy will be enforced.

You are expected to attend every class. If you are unable to attend a particular class, you are still responsible for the material covered in the class. You must make arrangements to obtain this material from another student. Lectures will not be repeated.

Do not leave early!

Do not come late!

I have an open door policy with respect to students. I'm in my office most of the time. I am willing to meet with you any time I am in my office. Feel free to come to me with any matter that is troubling you, even if it has nothing to do with the class.