CS 241: System Programming Spring 2010
Course Syllabus

Staff:

Robin Kravets (Instructor)
Office: 3114 SC
Office Hours: Tuesdays 11-12, SC 3114
e-mail: rhk@illinois.edu

Brighten Godfrey (Instructor)
Office: 3128 SC
Office Hours: Wednesday 2-3, SC 3128
e-mail: pbg@illinois.edu

TAs: Wade Fagen, Liping Chen, Farhana Ashraf, Riccardo Crepaldi
See class web page for office hours times and locations.

Contact: All questions about or problems with the class contents, web page, procedures, HWs, MPs or other material should be posted to the class newsgroup.
news: class.cs241

- All class questions
- This is your one-stop help-line!
- Will get answer < 24 hours

news: class.cs241.announce

- All class announcements (staff only)

Email should only be used for personal questions not postable on the news group.
e-mail: cs241help-sp10@cs.uiuc.edu

Textbook:

Introduction to Systems Concepts and Systems Programming
University of Illinois Custom Edition

Taken from:
Operating Systems: Internals and Design Principles, Fifth Edition
by William Stallings
UNIX™ Systems Programming: Communication, Concurrency, and Threads
by Kay A. Robbins and Steven Robbins
Computer Systems: A Programmer's Perspective
by Randal E. Bryant and David R. O'Hallaron

ISBN 0-536-48928-9

Prerequisites:

CS 225, CS 231 and credit or concurrent registration in CS232 are the prerequisites for CS 241.

Description:

A computer needs an operating system to manage its resources and provide support for common functions such as accessing peripherals. There are two categories of "customers" that an operating system must support. The first category is the community of users. We have all used computers and you may recognize operating systems functions such as creating folders (directories) and moving files around. These are examples of operating system support for users. User support is not the objective of this course. This

course addresses operating system support for the second category of customers; namely, the programmers. Those are people who write code to execute on the computer. When you write a program, it may have to interact with physical hardware (keyboard, screen, mouse, printers, hard disk, or network. For example, you may want to get input from a keyboard or mouse, you may want to read some configuration file stored on disk, you may want to output data to a screen or printer, or you may want to access a remote server across a network. The operating system presents common interfaces for programmers to perform these functions. The operating system also provides useful abstractions such as "tasks" (also called processes) "threads", and "semaphores". You can make the computer multitask by calling the operating system interface for creating new tasks or new threads. You can make these tasks coordinate and synchronize by using operating system semaphores. You can tell the computer the order in which you want tasks to be executed, which is called a scheduling policy. Finally, you can manage computer memory by calling the operating system function for memory management. System programming refers to writing code that tasks advantage of operating system support for programmers. This course is designed to introduce you to system programming.

By the end of this course, you should be proficient at writing programs that take full advantage of operating system support. To be concrete, we need to fix an operating system and we need to choose a programming language for writing programs. We chose the C language running on a Linux/UNIX operating system (which implements the POSIX standard). The C over UNIX/Linux is a very common combination used heavily by software that must provide high performance. It is much faster, for example, tha Java or C++ over Windows. Hence, this course introduces you to systems programming via the specific case of C over UNIX. By the end of the course you should be proficient with this programming environment and should be able to write non-trivial pieces of software from web server code to your own multiplayer Internet games. More specifically, after taking this course you should be able to accomplish the following:

1. Identify the basic components of an operating system, describe their purpose, and explain how they function.
2. Write, compile, debug, and execute C programs that correctly use system interfaces provided by UNIX (or a UNIX-like operating system).
3. List UNIX system calls, and invoke them correctly from within C programs.
4. Describe the difference between programs, processes, and threads.
5. Explain the meaning and purpose of process control blocks and other mechanisms that the operating system uses to implement the process and thread abstractions.
6. Write, compile, debug, and execute C programs that create, manage and terminate processes and threads on UNIX.
7. Define concurrency and explain the problems that may arise because of concurrent execution of multiple processes or threads. Explain how these problems can be avoided. Write code that avoids these problems.
8. Define semaphores, mutexes, and other synchronization primitives, explain their purpose, and describe their internal implementation.
9. Describe possible problems that arise from improper use of synchronization primitives (such as deadlocks) and present their solutions.
10. Write, compile, debug, and execute C programs that use UNIX synchronization primitives.
11. Describe operating system scheduling and use UNIX interfaces to set and modify scheduling policy parameters.
12. Define UNIX signals and signal handlers, and describe their use.
13. Write, compile, debug, and execute C programs with processes and threads that interact by invoking and catching signals.

14. Describe, configure, and use operating system timers and clocks.
15. Identify and apply principles of queueing theory to evaluate system performance.
16. Describe the concepts of I/O devices, files, directories.
17. Explain the internal implementation of files systems and operating system I/O.
18. Write, compile, debug, and execute C programs that use files and I/O on UNIX.
19. Describe the machine memory hierarchy, describe its components such as caches and virtual memory, and explain memory management mechanisms pertaining to these components such as paging and segmentation.
20. Write, compile, debug, and execute C programs that make use of memory management functions.
21. Explain the concept of DMA.
22. Describe the protocols (such as TCP and IP) and interfaces (such as sockets) used for communication among different computers.
23. Write distributed applications that communicate across a network.

Lectures:

11-11:50 am Monday, Wednesday and Friday in Room 1404 SC. Lectures cover important operating system concepts, and their implementation. It is the students' responsibility to read the textbooks and related materials. You are expected to attend lectures, and will be responsible for announcements made during lecture, on the cs241 web page, and on the newsgroup, class.cs241 and class.announce.cs241.

Discussion Sections:

Discussion sections are all held in 0220 Siebel Center. They will be a mix of short presentations, discussions and time for help with coding.

Grading:

Final Exam:	30%
Mid-term Exam:	20%
Homework (3):	15%
Team Machine Problems (7):	30%
Participation (Class involvement and pop quizzes):	5%

Academic Honesty:

Cheating is taken very seriously in CS 241. Be sure to understand the departmental policy on cheating. Your work in this class must be your own. If students are found to have collaborated excessively or to have blatantly cheated (e.g., by copying or sharing answers during an examination or sharing code for the project), all involved will at a minimum receive grades of 0 for the first infraction and reported to the academic office. Further infractions will result in failure in the course and/or recommendation for dismissal from the university.

Note on Machine Problems and Homework Assignments:

The programming assignments in this class will be based on Posix Programming Interface. We will have eight machine problems with different difficulty degrees. The MP assignments will be done in groups. There will be three homework assignments. The homework assignments will be done individually by each student. Please use the TSG Linux machines. To access these machines from outside the university, please use CITES VPN client.

Late Policies:

- Homework
  - Deadlines are strict
  - Late submissions will not be considered
- MPs
  - Please respect posted deadlines to ensure quick grading
  - Late MPs will be penalized 2% for each late hour (rounded off to the higher hour)
  - No submissions past 48 hours

Laboratory Facilities from CSIL:

We will be using linux machines, running POSIX system programming interface, in labs in 1245 DCL, 1265 and 1275 DCL and in 0216 SC (basement). You should make sure that you have accounts on the CSIL machines to do your machine problem assignments. We recommend that you use the machines `csil-linux-ts1.cs.uiuc.edu` or `csil-linux-ts2.cs.uiuc.edu` since our auto-grader program will check your MP submissions on these machines.

TSG runs the CSIL labs, you can find out more details here:

TSG Machine Labs: https://agora.cs.uiuc.edu/display/CSIL/Facilities
TSG Web Site: https://cs.uiuc.edu/tsg

The DCL lab hours are 10 am to midnight, Sunday through Thursday, and 10 am to 6 pm Friday and Saturday. 0216 SC is open 24/7. More details of the available facilities are available.

If you register late or otherwise have problems relating to the **existence of your account**, send email to userhelp@cs.uiuc.edu.

Class Schedule:

Note: this is a tentative schedule and may change to accommodate changes in the class.

Week	Dates	Lecture	Topic and Slides	Readings	MPs and HWs	Comments
1	1/18	Martin Luther King Day				
	1/20	1	Introduction to OS's	How to study guide		Survey Sheet (to be filled by you; not a test! Please find it among the assessments in Compass and submit it before next lecture)
	1/22	2	Introduction to C	Chapter 1; Chapter 2. C Tutorial here.	HW1 out (Text version)	
2	1/25	3	C No Evil			
	1/27	4	Operating Systems Orientation			
	1/29	5	System Calls	I/O Syscall tutorial here. Use Chapter 3 for reference as needed.	HW1 due MP1 out	Note: Chapter 3 gives a really good overview of UNIX survival skills MP1: Experiment and refresh your knowledge of C
3	2/1	6	Processes	Chapter 4 (Sec 3.1- Sec. 3.4), Chapter 7 (Sec 3.1- Sec 3.4)		
	2/3	7	Processes (2)	Chapter 5 (Sec 4.1), Chapter 7 (all), Chapter 8 (Sec 12.1 - Sec 12.4)		
	2/5	8	Threads	Chapter 8 (all)		
4	2/8	9	pThreads Tutorial		MP1 due MP2 out	MP2: Basic Processes and I/O
	2/10	10	Threads Systems Concepts		HW2 out	
	2/12	11	Scheduling Principles	Chapter 14 (Sec 9.1, 9.2).		

5	2/15	12	Scheduling Principles (2)		MP2 due MP3 out	MP3: Scheduling	
	2/17	13	Synchronization	Chapter 9 (Sec 5.1, 5.2), Chapter 11.			
	2/19	14	Semaphores	Chapter 9 (Sec 5.3)			
6	2/22	15	Semaphores and Mutex	Chapter 12, Chapter 13 (Sec 13.1-13.3)	MP3 due MP4 out	MP4: Synchronization	
	2/24	16	Classic Synchronization Problems				
	2/26	17	More on Synchronization				
7	3/1	18	Deadlocks	Chapter 10 (Sec 6.1 - Sec 6.6)	HW 2 due.		
	3/3	19	Deadlocks				
	3/5		Midterm Review	Lectures 1-17			
8	3/8		In Class Midterm Exam				
	3/10	20	Interprocess Communication: Pipes and FIFOs				
	3/12	21	Interprocess Communication: Memory Mapping				
9	3/15	22	Introduction to Networking	Beej's Guide	MP4 due. MP5 out	MP5: IPC	
	3/17	23	Network Programming				
	3/19	24	More Network Programming				
10	3/22	**Spring Break**					
	3/24	**No Classes**					
	3/26						

11	3/29	25	Networked Applications	Code Example			
	3/31	26	Advanced Network Programming				
	4/2	27	Networking				
12	4/5	28	Introduction to Signals, Signals, Signal Mask	Chapter 15 (Sec 8.1 - Sec. 8.4)	MP5 due MP6 out	MP6: Networking and Sockets	
	4/7	29	Signals and Timers	Chapter 15 (all), Chapter 16			
	4/9	30	IO				
13	4/12	31	Files, IO, devices				
	4/14	32	Files, IO, devices		HW3 out		
	4/16	33	Files, IO, devices				
14	4/19	34	Allocation & VM		MP6 due MP7 out	MP7: Files and I/O	
	4/21	35	Paging Memory Hardware				
	4/23	36	Page Replacement and Page Allocation				
15	4/26	37	Advanced Topics				
	4/28	38	Advanced Topics		HW3 due.		
	4/30	39	Advanced Topics				
16	5/3	41	Advanced Topics		MP7 due.		
	5/5	42	Wrap up and Beyond CS241				
	5/10	**Final Exam: 7:00 - 10:00 p.m., Location TBA**					