Course Syllabus

Edit

San José State University

Science/Computer Science
SE/CS 149, Operating Systems, Section 2, Fall 2020
Course and Contact Information

Instructor:	Ben Reed
Office Location:	https://sjsu.zoom.us/j/4077267356 (https://sjsu.zoom.us/j/4077267356) (actually the spare bedroom...)
Telephone:	(408) 924-5174
Email:	ben.reed@sjsu.edu
Office Hours:	MW 4:30-5:45PM TR 11AM-12PM F 9-10AM
	Monday & Wednesday/ 1:30-2:45
Classroom:	https://sjsu.zoom.us/j/94774145765?pwd=bzZDSk12QkhYN28xVWhoSjFxajJOdz09cyberspace! (https://sjsu.zoom.us/j/94774145765?pwd=bzZDSk12QkhYN28xVWhoSjFxajJOdz09)
Prerequisites:	CS 146 or SE 146 (with a grade of "C-" or better)

Course Description

Fundamentals: Contiguous and non-contiguous memory management; processor scheduling and interrupts; concurrent, mutually exclusive, synchronized and deadlocked processes; files. Substantial programming project required.

Course Learning Outcomes (CLO)

Upon successful completion of this course, students will be able to:

1. Know the difference between kernel and user space.
2. Know how to manage processes.
3. Understand virtual memory management.
4. Understand CPU scheduling policies.
5. Know how to manage threads and the issues associated with them.
6. Understand and implement concurrent data structures.
7. Know how to observe kernel operation and configuration.
8. Understand the issues associated with different storage technologies.
9. Understand how virtual file systems work.
10. Identify and reason about ethical issues surrounding various operating system concepts.

Required Texts/Readings

Textbook

Operating Systems: Three Easy Pieces: http://pages.cs.wisc.edu/~remzi/OSTEP/

Other technology requirements / equipment / material

Programming assignments will be a significant part of this course, so access to a computer is required. Your computer must be able to run Oracle VM VirtualBox in 64-bit mode https://www.virtualbox.org/.

Course Requirements and Assignments

Homework will be given, but will not be graded. It is intended for self-evaluation and will be the basis for future exams. I encourage students to work on homework in groups and discuss possible solutions together. We will take time at the beginning of each class to discuss any difficulties students have completing the homework.

Along with technical questions in the homework, we will also discuss ethical issues related to operating systems. We want you to understand that along with technical choices come moral implications, and we want to be able to identify and reason about them. There will be 2 written (1 page) assignments to discuss contemporary ethical issues in operating systems today.

We will be using iClicker to make sure everyone is up to speed. To encourage participation 1% of your final grade will come from your participation. At the end of the semester the points you will 100% if you get at least 70% of participation. Anything under 70% will be prorated.

I do not grade on a curve. The exams and projects measure what you are expected to have learned. There aren't many opportunities for extra credit, but there are bonus questions on exams.

We will be doing individual programming assignments. You will have one week after a programming assignment is assigned to complete it. I will allow assignments to be submitted up to one week late. You may not submit assignments that are over a week late. **Individual programming assignments are not group projects.** If students get help on assignments, even to resolve a stupid problem, it must be documented in the code with the name of the person rendering the help and a brief description of the help provided. Extensive help on a project will result in a reduced grade. Failure to document help, or any other forms of cheating will result in a failing grade on the assignment at a minimum and may result in failure of the course. All incidents will be reported to the Office of Student Conduct & Ethical Development. See **http://info.sjsu.edu/static/schedules/integrity.html (http://info.sjsu.edu/static/schedules/integrity.html)** for more information. Even in open source, you cannot copy code from one open source project to another without attribution. Sharing solutions with other students, even if it is indirectly through public source repositories, falls under "aiding and abetting".

The University Policy S16-9 **(http://www.sjsu.edu/senate/docs/S16-9.pdf)**, Course Syllabi (http://www.sjsu.edu/senate/docs/S16-9.pdf) requires the following language to be included in the syllabus:

“Success in this course is based on the expectation that students will spend, for each unit of credit, a minimum of 45 hours over the length of the course (normally three hours per unit per week) for instruction, preparation/studying, or course related activities, including but not limited to internships, labs, and clinical practica. Other course structures will have equivalent workload expectations as described in the syllabus.”

Final Examination or Evaluation

This course will have a cumulative final exam given during exam week.

There will be three in-class exams given in the semester (the last being the final exam :)). The second exam will have two questions derived from the previous exam, and the final exam will have two questions derived from the first exam and two questions derived from the second exam.

Grading Information

Determination of Grades

Grades will be calculated by averaging the percentages of the average of group project grades, the individual project grades, the two mid-semester exams, and the final. Thus, the grade distribution is 23% individual projects, 21% exam 1, 21% exam 2, 24% final exam, 10% ethic projects, and 1% participation via (iClicker).

I do not curve grades. I do round the reported grade up. I don't feel that it is fair that only students who beg for their grade to be rounded up to get it, so I automatically round-up for everyone. So if you get an 89.1, that will be rounded to an A-. However, do not expect a 68.9 to be rounded to a C-, even if you can

scrounge up some points to get it to a 69.1. The whole point of rounding is to get people that got close enough the grade they deserve. Getting a 68.9 is not close to a C-.

Percentage	**Grade**
97 and above	A+
92-96	A
90-91	A-
88-89	B+
82-87	B
80-81	B-
78-79	C+
72-77	C
70-71	C-
68-69	D+
62-67	D
60-61	D-
59 and below	F

Classroom Protocol

This is your class. Please ask questions. Please come prepared. Do not engage in activity that may distract other students.

I do not take attendance except for the first two classes. Students not attending either of the first two classes will be dropped to make room for students on the waiting list. Attempting to get marked as present (by have someone else attend in your place or using technological deceptions) will be considered academic dishonesty and at a minimum will result in you getting dropped from the course.

University Policies

Per University Policy S16-9, university-wide policy information relevant to all courses, such as academic integrity, accommodations, etc. will be available on Office of Graduate and Undergraduate Programs' Syllabus Information web page **(http://www.sjsu.edu/gup/syllabusinfo/)** at http://www.sjsu.edu/gup/syllabusinfo/" Make sure to review these policies and resources.

CS 149, Operating Systems, Section 2, Spring 2019

Course Schedule

Week	Date	Topics, Readings, Assignments, Deadlines
1	8/19/2020	intro to OS architecture review: CPUs, memory, registers, ASCII crash course on C part 1 **https://learning.oreilly.com/library/view/c-programming-language/9780133086249/ch01.xhtml** **(https://learning.oreilly.com/library/view/c-programming-language/9780133086249/ch01.xhtml)**
2	8/24/2020	booting: **https://www.ibm.com/developerworks/library/l-linuxboot/index.html** **(https://www.ibm.com/developerworks/library/l-linuxboot/index.html)** crash course on C part 2 **https://learning.oreilly.com/library/view/c-programming-language/9780133086249/ch02.xhtml**

		(https://learning.oreilly.com/library/view/c-programming-language/9780133086249/ch02.xhtml)
2	8/26/2020	crash course on C part 3 https://learning.oreilly.com/library/view/c-programming-language/9780133086249/ch05.xhtml (https://learning.oreilly.com/library/view/c-programming-language/9780133086249/ch05.xhtml)
3	8/31/2020	processes: http://pages.cs.wisc.edu/~remzi/OSTEP/cpu-intro.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/cpu-intro.pdf)
3	9/2/2020	process API: http://pages.cs.wisc.edu/~remzi/OSTEP/cpu-api.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/cpu-api.pdf)
4	9/9/2020	direct execution: http://pages.cs.wisc.edu/~remzi/OSTEP/cpu-mechanisms.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/cpu-mechanisms.pdf)
5	9/14/2020	scheduling: http://pages.cs.wisc.edu/~remzi/OSTEP/cpu-sched.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/cpu-sched.pdf)
5	9/16/2020	multi-level scheduling: http://pages.cs.wisc.edu/~remzi/OSTEP/cpu-sched-mlfq.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/cpu-sched-mlfq.pdf)

6	9/21/2020	exam 1
6	9/23/2020	signals & memory http://pages.cs.wisc.edu/~remzi/OSTEP/vm-intro.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/vm-intro.pdf) (http://pages.cs.wisc.edu/~remzi/OSTEP/vm-intro.pdf)
7	9/28/2020	memory API: http://pages.cs.wisc.edu/~remzi/OSTEP/vm-api.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/vm-api.pdf)
7	9/30/2020	free space management: http://pages.cs.wisc.edu/~remzi/OSTEP/vm-freespace.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/vm-freespace.pdf)
8	10/5/2020	paging: http://pages.cs.wisc.edu/~remzi/OSTEP/vm-paging.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/vm-paging.pdf)
8	10/7/2020	page tables
9	10/12/2020	more page tables
9	10/14/2020	TLP & paging mechanisms: http://pages.cs.wisc.edu/~remzi/OSTEP/vm-tlbs.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/vm-tlbs.pdf) & http://pages.cs.wisc.edu/~remzi/OSTEP/vm-beyondphys.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/vm-beyondphys.pdf)

10	10/19/2020	COW http://pages.cs.wisc.edu/~remzi/OSTEP/vm-complete.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/vm-complete.pdf) section 23.1 in "Other Neat Tricks"
10	10/21/2020	replacement policies: http://pages.cs.wisc.edu/~remzi/OSTEP/vm-beyondphys-policy.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/vm-beyondphys-policy.pdf)
11	10/26/2020	exam 2
11	10/28/2020	threads & thread API: http://pages.cs.wisc.edu/~remzi/OSTEP/threads-intro.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/threads-intro.pdf) & http://pages.cs.wisc.edu/~remzi/OSTEP/threads-api.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/threads-api.pdf)
12	11/2/2020	locks: http://pages.cs.wisc.edu/~remzi/OSTEP/threads-locks.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/threads-locks.pdf)
12	11/4/2020	concurrent data structures: http://pages.cs.wisc.edu/~remzi/OSTEP/threads-locks-usage.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/threads-locks-usage.pdf)
13	11/9/2020	condition variables & semaphores: http://pages.cs.wisc.edu/~remzi/OSTEP/threads-cv.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/threads-

		cv.pdf) & http://pages.cs.wisc.edu/~remzi/OSTEP/threads-sema.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/threads-sema.pdf)
14	11/16/2020	concurrency bugs: http://pages.cs.wisc.edu/~remzi/OSTEP/threads-bugs.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/threads-bugs.pdf)
14	11/18/2020	files & directories: http://pages.cs.wisc.edu/~remzi/OSTEP/file-intro.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/file-intro.pdf)
15	11/23/2020	hard disks: http://pages.cs.wisc.edu/~remzi/OSTEP/file-disks.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/file-disks.pdf)
15	11/25/2020	RAID: http://pages.cs.wisc.edu/~remzi/OSTEP/file-raid.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/file-raid.pdf)
16	11/30/2020	file system implementation: http://pages.cs.wisc.edu/~remzi/OSTEP/file-implementation.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/file-implementation.pdf)
16	12/2/2020	file system implementation: http://pages.cs.wisc.edu/~remzi/OSTEP/file-implementation.pdf (http://pages.cs.wisc.edu/~remzi/OSTEP/file-implementation.pdf)

17	12/7/2019	review
Final Exam	12/15/2020	from 12:15-2:00