CS 4375: Operating Systems Concepts

Fall 2024

Instructor: Nigel Ward, nigel@utep.edu, Computer Science 3.0408, also 747-6827
Office Hours: Wednesdays 3-4 and Fridays 10:30-11:30, or by appointment.

IA: to be determined

Class Time: TuTh 10:30 – 11:50, CCSB G.0208


Course Description: Process and thread management, processor scheduling and concurrency, interprocess communication, memory management, input/output management, file systems, and networking basics

Goal: Introduce concepts that will be foundational for further study, whether academic or professional, of 1) Computer Security and Forensics, 2) Systems Administration and Network Administration, and 3) Systems Programming and Network Programming, including developing for embedded systems, cloud systems, and high-performance systems.

Course Policies

The prerequisite for this class is CS 3432 (Computer Organization) with a C or better. This may be enforced retroactively.

Assigned readings are to be done before class.

Bring to class a whiteboard marker or two and the relevant textbook, either hardcopy or on your device, or on your buddy’s device.

Assignments are to be submitted in hardcopy, unless otherwise specified. The maximum points attainable for late assignments will be reduced by 10% per day or partial day of lateness, for up to five days; or more if the lateness impedes participation in class or the assignment is received after the solution has been discussed. Email submissions of assignments are not accepted unless otherwise specified.

Assignments are to be done individually unless specifically designated as group assignments. While you may discuss assignments with others, and you may share testing data and strategies, your solutions should be designed, written/integrated, and tested by you alone. If you need help, consult the TA or the instructor.

The use of found code and shared code is acceptable, unless otherwise specified, provided that you acknowledge your sources, state specifically what you used, and understand every line of code.

Programming assignments will be graded primarily on functionality, design quality, thoroughness of testing, and readability, and for those with demos, ability to discuss. Some of these factors involve subjective judgments; if you have questions about the criteria, please see the TA or the instructor.

Tests will be closed-book, except that one single-sided page of hand-written notes may be used for the first test, two for the second test, and three for the final. If you leave the room for any reason, your test will be graded on only what you did up until that time. No make-up exams, assignments quizzes, etc. will be given except under the conditions set forth in the Catalog. The lowest quiz grade will be
Grades will be based on two components: tests and the final examination, and assignments and other factors. Balanced performance across both components is important, so the final grade will be based on the order-weighted average of the two, with the weakest score’s weight being twice that of the strongest score. Within the first component, each test will count for 28% and the final for 44%. Within the second component, the assignments will count for about 90%, with the remainder based on quizzes, in-class exercises, participation, and, very rarely, extra credit, notably for being the first to notify us of a problem with an assignment, or for producing top-quality code suitable for sharing with the class as an exemplar.

Assignments and tests will be challenging. Grading will be on a points-earned basis (points above zero), rather than a points-off basis (points below expectation). Letter grades will be assigned accordingly: the A/B break will be 80% or lower and the B/C break 70% or lower.

Students are free to attend class or not, bearing in mind that absence may annoy other students, interfere with learning, and result in a lower grade.

General Policies

No make-up exams or assignments will be given except under the conditions set forth in the Catalog. Students are expected to be punctual and to conduct themselves professionally and courteously.

If you have or suspect a disability and need accommodation, contact the Center for Accommodations and Support Services at 747-5148 or at cass@utep.edu or visit Room 106 Union East.

Topics, Readings and Major Assignments, tentative

<table>
<thead>
<tr>
<th>Topic</th>
<th>Duration</th>
<th>Readings/Assignments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>(3 days)</td>
<td>ADAD Ch. 1-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Python warm-up asst 3%</td>
</tr>
<tr>
<td>Processes</td>
<td>(3 days)</td>
<td>ADAD Ch 5-7, 11</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shell assignments 20%</td>
</tr>
<tr>
<td>Memory</td>
<td>(3 days)</td>
<td>ADAD Ch 12-18</td>
</tr>
<tr>
<td></td>
<td></td>
<td>sundry exercises 5%</td>
</tr>
<tr>
<td>Sockets</td>
<td>(3 days)</td>
<td>various readings</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Socket assignments 15%</td>
</tr>
<tr>
<td>Threads and Concurrency</td>
<td>(5 days)</td>
<td>ADAD 25 - 26; portions of 28 - 32</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Concurrency assts 30%</td>
</tr>
<tr>
<td>Networking</td>
<td>(5 days)</td>
<td>ADAD 48; PD Ch 1, §2.1, §3.1 thru 3.1.1,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>§3.3 thru 3.3.5, Ch 5 thru 5.2.3, §7.1, 8.5.6,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>§9.3.1; ADAD 56</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Networking assts 12%</td>
</tr>
<tr>
<td>Files and Devices</td>
<td>(4 days)</td>
<td>ADAD Ch. 35-37, 39-40; portions of 38-46</td>
</tr>
<tr>
<td></td>
<td></td>
<td>File System assts 8%</td>
</tr>
</tbody>
</table>

Important Dates

August 27: first day
September 26 (tentative): Test 1
Level 1 Outcomes: Knowledge and Comprehension
Level 1 outcomes are those in which the student has been exposed to the terms and concepts at a basic level and can supply basic definitions. The material has been presented only at a superficial level.

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

V1. Choose a scheduling approach suitable for a given simple problem.
V1j. Explain segmentation and its security implications.
V1l. Explain some ways in which virtualization creates vulnerabilities.
V1m. Describe and motivate the components of process and virtual machine context.
V1n. Explain the need for paging and the basics of demand loading.
V1o. Describe the motivation for and gross characteristics of a trusted computing base.
V1x. Explain how domain names, IP addresses, and file names work and why.
C1c. Given an application, identify the factors relevant to choosing a synchronous or asynchronous solution.
E1f. Choose when to use datagram versus virtual-circuit communication.
E1h. Differentiate transmission and propagation latencies and some factors affecting them.
E1i. Explain how data is serialized (byte order, representation, buffering).
E1j. Explain the difference between lossy and lossless compression.
E1l. Interpret the output of a packet capture tool.
E1n. Explain the role of cryptographic hashes and symmetric and asymmetric keys in security.
E1o. Explain the functionality handled at the physical, link, network, and transport layers.
E1p. Explain how data is handled at different network layers.
E1q. Explain some data structures used for storing files on disk.
C1r. Explain the memory hierarchy, locality, and redundancy.
E1s. Explain generic device APIs, including the bidirectional handling of interrupts and requests.

Level 2 Outcomes: Application and Analysis
Level 2 outcomes are those in which the student can apply the material in familiar situations, e.g., can work a problem of familiar structure with minor changes in the details.

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

V2. Use the concepts of process state and state transition to characterize system and process behavior.
V2r. Relate the distinction between supervisor and user permissions to the design and implementation of system calls.
V2t. Write programs that use interprocess communication, specifically pipes and/or sockets.
V2u. Use simple system calls for common needs.
C2g. Implement producer-consumer coordination.
C2h. Build a server-side program that uses multi-threading to handle multiple simultaneous clients.
C2i. Identify situations where deadlock may occur, and suggest ways to prevent it.
A2g. Perform simple arithmetic computations related to major families (e.g. determine page number or whether an address is within a power-of-2 segment)

Level 3 Outcomes: Synthesis and Evaluation
Level 3 outcomes are those in which the student can apply the material in new situations. This is the highest level of mastery.

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

V3. Choose among virtual machines, processes, containers and sandboxes as ways to support common programmer needs.
V3w. When a process or a computer is running too slowly, infer some probable causes.
C3j. Distinguish when blocking vs nonblocking calls are appropriate.