CS444: Introduction to Operating Systems

FALL 2026 Draft 27 July

Dr. J. Holly DeBlois

Office: McCormack, 3rd floor, room M-3-201-32, phone 287-6490

Office hours: 12:00-1:30pm Tues and Thurs

Lectures and Class	Tuesday & Thursday, 2:00-3:15, W01-0005 section 01 (course 1409) Tuesday & Thursday, 5:30-6:45, W01-0004 section 02 (course 3737)
Instructor Email	jane.deblois@umb.edu
Instructor Website	Lecture and assignments are at: https://www.cs.umb.edu/~hdeblois/cs444/f26 Grades are on canvas https://www.umb.edu/canvas/
Portal:	**Register for cs444 course directory**: https://portal.cs.umb.edu/
Piazza:	Join and post questions: https://piazza.com/umb/fall2026/cs444

Course Description: In CS444, we present the basic features of operating systems (OS), a layer of software between hardware and user software. We code in C. We use the Linux servers of the Computer Science (CS) department. We use Emacs. We compare and contrast types of OS. We develop abstractions to understand how various OS functions, written mostly in C, inter-operate.

Textbooks: (1) *Modern Operating Systems, 5th ed.* (Pearson, 2023) by Andrew S. Tanenbaum and Herbert Bos (T&B), with ch1-12. ***(2)*** C code textbook: *The C Programming Language, 2nd edition* (Prentice Hall, 1978) by Brian W. Kernighan and Dennis M. Ritchie (K&R), online: https://archive.org/details/the-ansi-c-programming-language-by-brian-w.-kernighan-dennis-m.-ritchie.org . (**3**) *Operating Systems Design and Implementation, 3rd edition* (Pearson, 2006) by Tanenbaum and Albert S. Woodhull (T&W), with ch1M-5M and MINIX3 OS code. (**4**) Emacs editor, Richard Stalllman, 1984, see: https://www.gnu.org/software/emacs/refcards/pdf/refcard.pdf .

Attendance: Mandatory. **If you miss >5 classes, your final grade will be lower.** If you miss 6 or 7 classes, you lose 1 letter grade step (for example, B to B-). If you miss 8 or more classes, you lose 2 letter grade steps (for example, B to C+). Being late to class or leaving class for more than a few minutes will lower your participation points. See Participation below.

Topics: We shall cover the following topics:

- **Operating System** definitions, speeds, sizes and features by type
- **Processes** and scheduling algorithms, threads, the critical region problem and semaphores
- **Memory management**, binary, ASCII, hex, cache memory, virtual memory, paging and segmentation
- **File systems**: implementation, sizes, permissions, management and optimization
- **Input/Output control**: buffering, spooling, disk management and disk access scheduling
- **Deadlocks**: detection, recovery, avoidance and prevention
- **Virtualization** and cloud computing, VMware case study
- **Multiple processor systems:** multiprocessor, multicomputer and distributed systems
- **Security** threats and defenses
- **Linux/Android/Windows**: case studies, differences and similarities
- **C Code:** grammar, sample code, MINIX3 code, code for algorithms to implement
- **CS Server Skills**: access linux servers, use secure copy to and from, passwordless login, command line, see who's working and what's running
- **Abstractions of OS features:** sketch diagrams of: hardware, layers of software (down to drivers, up to containers), timelines, architectures and mechanisms

Note: No courses required by the CS major, minor or certificate may be taken pass/fail.

Prerequisites: CS310 Algorithms and CS341 Computer Architecture. Use skills from CS240 Programming in C and from probability distributions in CS220 Applied Discrete Math.

Evaluation: Your score is a weighted average of eight groups of assignments:

- four **C** code programming projects (**36%**) each including four parts:
  - subdirectory in course dir named proj<n> ("proj1" etc.),
  - <starter>.c named starter<n> (in-class exercise handwritten, collected, scanned),
  - readMe.txt (handwritten page, added to as you work, in-class collected, scanned),
  - project C files (including typed <starter>.c) and executables (in subdir proj<n>),
- three **H**omework assignments (**27%**) each including three parts:
  - subdirectory in course dir named hw<n> ("hw1" etc.),
  - hw<n>.txt (in-class exercise on-server),
  - hw<n>.pdf (including photos of handwritten drawings),
- two oral **M**idterm tests (**20%**),
- **S**erver access and server skills (in-class exercises on-server) (**3%**),
- **D**iagrams of OS abstractions (in-class exercises handwritten, collected, scanned) (**4%),**
- **G**rammar of C (in-class exercises on-server and handwritten) (**6%**),
- **A**ttendance: whether you miss >5 classes, explained above (**2%**), and
- **P**articipation: whether you participate during class, explained below (**2%**).

The eight assignment groups are letters in bold. Part weights are set by points in assignments.

Dropped Scores: Because 5 missed classes are allowed, 5 of your lowest in-class exercsie scores will be dropped, one from **S**, **D**, **G** on-server, **G** handwritten and **P** handwritten.

Avoid Lost Points on Submitting. Edit files in your course directory or upload files ahead of when due. You lose all points if permissions are incorrect on your course dir or subdir and we cannot run your code or copy your files. You lose all points on an assignment if scripts cannot find your subdirectories and files (linux is case-sensitive so "HW1" is NOT the same file as "hw1").

Bring Computer and Pencil to Class for In-Class exercises On-server and Handwritten. You must bring a computer to class each day, capable of accessing your CS course directory. You will get zero points if you do not create the files in your course directory during class. Bring a pencil or pen for handwritten work. Use the given assignment page and fill in your name as it is in wiser and your student id. You will get zero points if you do not hand in the page during class.

Compile C Code on the server. The C code libraries on the CS server are NOT the same as the ones on your laptop or home computer, so you MUST compile, run gcc and test your executables on the server. Using VSCode is not enough. You need to be proficient "on" the servers. VSCode can cause you to exceed your given disk space and be unable to submit. **Type 'du -h' to check your usage. Delete unneeded files to fix or email operator@cs.umb.edu for help.**

Academic integrity will be strongly enforced. Your code and homework must be your own product, may include some code copied or modified from outside sources or generated by Large Language Models (LLMs), but must pass the MOSS criteria given above and you must understand the code you submit. See https://www.umb.edu/academics/provost/academic-integrity. We use the MIT guidance on putting sources into code. See https://integrity.mit.edu/handbook/writing-code/ In addition, for citing LLM-generated code, **document any code you generated and used by marking the start as START, citing the prompt you entered and further down marking the end as END.**

AI Policy. Per UMB syllabus requirement, we state that this course DOES allow the use of AI on C Code projects, so include start and end markers stated above, but DOES NOT allow use on homework. We recommend not allowing any LLM to store results for retraining. Know the name(s) of the LLM(s) you use. Put them in your handwritten readMe. Google search bar is "staffed" by Gemini. UMB provides student access to Microsoft 365 Copilot. Access to Claude Code is being explored.

Student Conduct: You must be honest in all your conduct. The University presupposes that work for academic credit is the student's own and complies with University policies above and here**:** **https://www.umb.edu/camp-life/dean-of-students/student-conduct-process/.** You may not post the work that you submit for this class publicly either during or after the semester is concluded.

Diagrams of OS Features on Paper. You must master abstracting OS features, different from coding. OS run in nanosecs with lines of code: >20 million (Linux), >50 million (Windows).

C Code AI Use Cases. Be deliberate about the amount of non-original C code you copy, copy and modify or generate. Your readMe must indicate your use (or non-use) of AI.

C Code Programming Projects: Incrementals. Your submitted project C files must include code and executables with at least two prior versions of each. To save an earlier version of your <file>.c, copy it to a name with a date indicator ('cp -p <file>.c <file_23jul>.c'). Then edit <file>.c.

Grading C Projects. We test your code using Measure of Software Similarity (MOSS) provided by Stanford University. **Software similarity to other students' code or sample generated code greater than 45% earns 0 points.** We do a practice MOSS run on <starter>.c. We may call you in to explain your code. If we cannot identify your own work in C vs. your generated code, your grade will be 50.

Grading All Work. A zero is posted for late work. Ross Center students must request the extension before the work is due; when graded, their score will be posted. Late penalty is 1% per hour down to half the score. The instructor may raise or lower any grade that is posted in error.

Homework: Upload your homework to the server. Drawings in the homework are handwritten, photographed, and included in your typeset document converted to PDF by an editor. Some homeworks request partial submission as .txt files. Use "ls -la" to check filenames.

Midterm Oral Tests: The two tests are oral exams, answering the instructor's questions in a 30- or 20-minute session. For test1, students take the test in pairs. The sign-up is a doodle signup sheet posted a week ahead. The two students in test1 are given the same score, so it does not matter who gives the correct answer. For test2, each student is individually tested. Tests are randomized.

Participation: The instructor may deduct participation points if you are late, leave early or leave the classroom for too long or too often. We use 1 pt per 10 mins, 8 points per session and 25 sessions total. Asking a good question or helping another student are ways to earn participation points. There is not enough time to be sure to give points for every such helpful or generous act. Be sure the instructor and your classmates know your name, so your questions and/or your helpful acts can be noticed. In-class on-server uploading a photo of your early readMe <n> counts 1 point.

Collaboration policy: You are encouraged to ask questions in class and may consult with your fellow students about assignments. When writing C code, do not stay "stuck" – ask for help via

piazza. Be careful not to help another student by giving code or the specifics of your design – keep your laptop closed while conversing and don't share videos that picture code you used.

Extensions and Late Penalties: In-class exercises, homework, C code projects and tests are scheduled for precise times. Late test arrival means you will be given proportionately less of the test and thus a lower grade. Late class arrival means you may lose participation points or miss hand-written work and be graded zero. UMB-mandated extensions will be given, but you must write the instructor **before the work is due**. If you are ill, write the instructor and we will work something out. If you are ill enough to miss a test, your make-up can only earn 50% of the full grade unless you provide a doctor's note. Traveling or delayed flights are not valid reasons for extensions. Joining the class late earns you 48 hours extension on hwk1 and an opportunity to make-up in-class assignments you missed. Late work loses 1% per hour down to half the points.

Accommodation: Section 504 of the Rehabilitation Act of 1973 offers gu delines and support for curriculum modifications and adaptations for students with documented disabilities. Contact the Ross Center at 617-287-7430 and please discuss your accommodations with the instructor.

Syllabus and Schedule Subject to Change: The instructor reserves the right to change the syllabus when necessary and will let you know. Here is the Schedule (25 classes + 2 tests over 14 weeks):

WEEK	Class#/Date/Post&Due	Reading	Lecture Topic	Class Activities **Note: In-Class is abbreviated IC, not all IC assignments are designated yet**
1	#1-Tue Sep 8 hw1 posted	website, syllabus, ch1	ch1: Introduction, client-server	**Use** browser, review the syllabus **Lecture 1**: ch1, read hw1 **IC On-server 1:** #S1 login (register at portal first), 'ls -la', hw1 dir, scp/cp instructor f.pdf **IC Handwrit 1**: #D1 draw client-server, hpath
	#2-Thu Sep 10	Huffman, emacs, ASCII chart	Huffman slides, C code	**Lecture 2:** Huffman slides, read hw1 **IC On-server 2:** #S2 shell, emacs f, hexdump **IC Handwrit 2**: #D2 draw simple Huff tree
--	----	----	----	----
2	#3-Tue Sep 15 hw1 due, proj1 posted Add/drop Ends	ch12, ch1M, K&R, ApA	ch1: Introduction ch12: Design OS	**Lecture 3:** ch12, grammar, ch1M, read proj1 **IC On-server 3**: #S3 ls -la hw1, shell, emacs <file>.c, comment grammar, gcc, run endian **IC Handwrit 3**: #C1-1 <reqd-start -proj1>.c
	#4-Thurs Sep 17	ch1, ch2 K&R pointers, ApB	ch1: Introduction ch2: Processes, threads	**Lecture 4:** ch1, ch2, read proj1 **IC On-server 4:** #G1-S emacs f.c print pointers **IC Handwrit 4**: #G1-H endian.c grammar
--	----	----	----	----
3	#5-Tue Sep 22	ch2, ch2M	ch2: Scheduling	**Lecture 5**: ch2, ch2M **IC On-server 5**: #P1 running processes, scp photo your readMe page to upload **IC Handwrit 5**: #D3 timeline diagram
	#6-Thu Sep 24	ch3	ch1: Introduction ch3: Memory	**Lecture 6:** ch3, ch3M **IC On-server 6:** #S4 passwordless login

				IC Handwrit 6: #D4 draw memory
--	----	----	----	----
4	#7-Tue Sep 29 proj1 due, hw2 posted with hw2.txt	ch3, ch4M	ch1: pointers ch3: Memory	**Lecture 7**: ch3, read hw2 **IC On-server 7**: #S5 output formats **IC Handwrit 7**: #C1-2 collect readMe proj1
	#8-Thu Oct 1	ch1, ch4	ch1: Introduction ch4: file systems	**Lecture 8**: ch1, ch4, read hw2 **IC On-server 8** : #G2-S emacs hw2.txt **IC Handwrit 8**: #G2-H grammar anal.
-	----	--	----	----
5	#9-Tue Oct 6 hw2 due, proj2 posted	ch4, ch5M	ch1: Introduction ch4: File systems	**Lecture 9**: ch4, ch5M, read proj2 **IC On-server 9**: #G3-S look MINIX3 **IC Handwrit 9**: #C2-1 <start-proj2>.c
	#10-Thu Oct 8 Fri Oct 9 - N/A grade due	ch5	ch5: I/O	**Lecture 10**: ch5, read proj2 **IC On-server 10**: #S6 look at shell, test **IC Handwrit 10**: #D6 lost print job
-	----	----	----	----
6	Mon Oct 12 - Holiday			
	#11-Tue Oct 13	ch5, ch3M	ch5, ch3M	**Lecture 11**: ch5 I/O, raid, nibbles **IC On-server 11**: #P2 scp photo readMe **IC Handwrit 11**: #D7 mini chart Ham.
	#12-Thu Oct 15 review posted	Popek	ch7.11 vmware	**Lecture 12**: ch7.11, read review **IC On-server 12**: # **IC Handwrit 12**: #
--	----	----	----	----
7	#13-Tue Oct 20 proj2 due, **test1 signup** proj3 posted	ch1, Hamming	ch1: Introduction Hamming	**Lecture 13**: Hamming, raid, nib., read proj3 **IC On-server 13**: # **IC Handwrit 13**: #C2-2 collect readMe page, C3-1 <start-proj3>.c
	#14 Thu Oct 22	ch1-5,12,C	Review session	**Lecture 14: Review**, sample problems
--	----	----	----	----
8	Mon-Wed Oct 26-28 no class Tues	ch1-5,12, C testing	test **1** (Ch1-5, 12, C code)	**Test1: Mon-Wed, 8am-8pm, students come two at a time, about every 30 minutes**
	#15-Thu Oct 29	ch6,ch3M	Deadlock	L**ecture 15**: chx, read proj3 **IC On-server 15**: #H2? **IC Handwrit 15**: #D8?
--	----	----	----	----
9	#16-Tue Nov 3	ch7	ch1,ch7, Virtualization and the Cloud	**Lecture 16**: ch7 **IC On-server 16**: # **IC Handwrit 16**: # in class C code **2-**from hw2
	#17-Thu Nov 5	ch8	ch8: multi-processor systems	**Lecture 17**: ch **IC On-server 17**: # **IC Handwrit 17**: #
--	----	----	----	----
10	#18-Tue Nov 10 proj3 due	queueing, threads	queueing, threads	**Lecture 18**: ch7.11 VMware **IC Handwrit 18**: #C3-2 collect readMe page

	hw3 posted, hw3.txt			**Field trip**, last 25 mins, CS server room
	Wed Nov 11 -Holiday #19-Thu Nov 12	ch9	ch9: Security	**Lecture 19**: ch9 **IC On-server 19**: # **IC Handwrit 19**: #
--	----	----	----	----
11	#20-Tue Nov 17 hw3 due, proj4 posted	ch1, ch10	ch10: Unix, Linux, Android	**Lecture 20**: ch10, read proj4 **IC On-server 20**: # **IC Handwrit 20a, 20b**: #C4-1 <start-proj4>.c
	#21-Thu Nov 19 Pass/Fail/Withdraw deadline	ch10.8	ch1: Introduction ch10.8: Androis	**Lecture 21**: ch1, ch10.8, read proj4 **IC On-server 21**: # **IC Handwrit 21**: # ----
--	----	----	----	**Lecture 22:** ch1, ch11 Windows, ch5.6
12	#22-Tue Nov 24 Holiday Thu Nov26-29 review posted	ch11	ch1,ch11: Windows	**IC On-server 22**: #P4 scp photo readMe **IC Handwrit 22**: # ----
--	----	----	----	**Lecture 23**: ch11 Windows
13	#23-Tue Dec 1 **test2 signup** proj4 due	ch11	ch11	**IC On-server 23**: # **IC Handwrit 23**: #C4-2 collect readMe page
	#24 Thu Dec 3	ch 6-11, C	review session	**Lecture 24**: **Review**, sample problems ----
--	----	----	----	**Test2: Mon-Wed, 8am – 8pm, individual, about every 20 minutes**
14	Mon-Wed Dec9 no class Tues	ch6-11, C testing	test **2** (Ch 6-11, C code)	
	#25-Thu Dec 10	to be posted	special topic	**Lecture 25**: special topic, last class
	14 weeks, 1 holiday, 27 class meetings	2 off for testing	25 lessons, 2 are review sessions	25 lectures

Revisions: None yet.

Table of score to grade conversions (default grading scheme “UMB letter” in canvas):

93 <= S = A
90 <= S < 93 = A-
87 <= S < 90 = B+
83 <= S < 87 = B
80 <= S < 83 = B-
77 <= S < 80 = C+
73 <= S < 77 = C
70 <= S < 73 = C-
67 <= S < 70 = D+
63 <= S < 67 = D
60 <= S < 63 = D-
S < 60 = F