CSCE 416 - Introduction to Computer Networks

Credit Hours

3 hours

Contact Hours

3 lecture hours, 14-week format, about 150-minutes per week from two 75 minutes lectures per week in regular face-to-face lecture.

Class Times

This class is listed as “Face-to-Face”, i.e. is a regular face-to-face lecture format in class.

Section	Class Time	Room
416-002	Monday/Wednesday 2:20-3:35 PM	Innova 1400

- Important dates this semester:
  - MLK Day: Jan. 16th, Monday
  - Midterm: March 1st
  - Midpoint of Semester: March 2nd
  - Spring Break March 5th-12th
  - Last day to drop a course or withdraw without a grade of "WF" being recorded: March 27th
  - Final Exam: April 26th, Wednesday, 12:30 p.m. Note that this is the first day of exams.

Final Exam Times (official times, according to Registrar)

Section	Final Exam Time
416-002	April 26, Wednesday, 12:30 p.m.

Instructor

James O’Reilly (oreillyj@cse.sc.edu)

Teaching Assistant

Moh Sadaat (“Sabbir”) (msaadat@email.sc.edu)

- Specific questions related to the grading or submission of a particular assignment go to the TA, though please copy me. Most others go to me. Questions regarding a lab/programming assignment can go to either of us.

Office Hours

All (my) office hours are posted here https://james-oreilly.youcanbook.me If you need to meet outside these times, please email me at oreillyj@cse.sc.edu . We can meet F2F or on Teams but posted office hours will generally be *preferred* for those physically at my office. ***If you schedule office hours with the above link but want to meet virtually, please send me an email telling me so.*** Regular business hours are preferred but not mandatory if these times do not work. **Please give me a window of times to pick from when requesting to talk outside of regular office hours to avoid another round of emails.**

Bulletin Description

Concepts and components of computer networks and the Internet; network applications; network protocol stack.

Prerequisites

CSCE 146

Learning Outcomes

Students will be able to:

- Demonstrate an understanding of the elements of a protocol and the concept of layering.
- Describe how to control access to a shared channel by multiple stations
- Explain the concepts of error control, flow control and congestion control.
- Illustrate how a packet is routed over the Internet.
- Design, build and describe a client-server application.

Required Materials and Software

Required Textbooks:

James F. Kurose and Keith R. Ross, Computer Networking: A Top-Down Approach (8th Edition). Addison-Wesley 2021. ISBN 9780136681557. We will also use GitHub for programming assignment submission.

All readings/materials comply with copyright/fair use policies.

Course Overview

Topics covered (Time spent is Approximate)

- Layered network architectures
- Network programming interfaces (e.g., sockets)
- Transport and data link protocols
- Physical media
- Local area networks
- Network routing protocols

Schedule of Topics/Assignments (Approximate)

Week	Week's 1st Day	Topics	Assignment Released (Due Date)
Week 1	M 1/9	Intro, Chapter 1	
Week 2	**W 1/18**	Chapter 1, Begin Chapter 2 (Application Layer) **(MLK Day)**	Quiz 1 (1/29)
Week 3	W 1/25	Chapter 2 (Application Layer)	Program 1 (2/19)
Week 4	W 2/1	Chapter 3 (Transport Layer)	Quiz 2 (2/12)
Week 5	W 2/8	Chapter 3 (Transport Layer)	
Week 6	W 2/15	Chapter 3 (Transport Layer)	Quiz 3 (2/26)
Week 7	W 2/22	Chapter 4 (Network Layer)	Program 2(3/26)
Week 8	**W 3/1**	Chapter 4 (Network Layer), Midterm, **Spring Break**	MT1 (3/1)
Week 9	**W 3/15**	Chapter 4 (Network Layer)	Quiz 4 (3/19)
Week 10	W 3/22	Chapter 5 (Link Layer)	

Week 11	W 3/29	Chapter 5 (Link Layer)	Quiz 5 (4/4), Program 3 (4/9)
Week 12	W 4/5	Chapter 5 (Link Layer)	Quiz 6 (4/18)
Week 13	W 4/12	Chapter 6 (Physical Layer)	Program 4 (4/23)
Week 14	W 4/19	Special Topics, Review, "Flex" Time	

Notes: A Week corresponds to (2) lectures and may not always line up with Monday and Wednesday of the same week; Most of the weeks start on Wednesday due to MLK day.

Student-to-Instructor (S2I), Student-to-Student (S2S), and Student-to-Content (S2C) Interactions

This course will be delivered **fully face-to-face**. I am going to attempt to record lectures, as the room assigned is well laid out for this, but the quality will not be sufficient for more than occasional use, e.g., as a backup for illness.

• Student-to-Instructor (S2I) Interaction: Students will attend regular lectures and may ask questions there, by email, or in office hours.

- Discussions in virtual office hours may be recorded, *assuming the student agrees*, and uploaded to Blackboard if the discussions are like in-class lecture questions and not regular "office hour" questions, e.g., troubleshooting programming assignments. Recorded lecture questions may also be uploaded.

• Students-to-Student (S2S) Interaction: Students will engage in discussions through in class discussion or on homework.

• Student-to-Content (S2C) Interaction: Students will engage with course content by completing homework/quizzes to follow the lecture, and other assessments.

Communication and Feedback

The instructor will reply to all feedback in a reasonable amount of time; the same is expected of the students. Specifically:

• Communication: Responses to email communication and questions will be provided within one business day. Questions arriving on the weekend may be answered the following Monday.

• Assignment Grading: Grades for assignments (quizzes, programming assignments, tests) will be returned within one business week (5 business days) of the due date.

If I fail to respond within these timeframes, feel free to send another email. Sometimes there's a spam filter (or an accidental swipe on my cellphone…).

Technology

Specific Technologies, Programming Languages, Programs

The students will install and/or use Java.

Minimal Student Technical Requirements

Students should be familiar with Java. The prerequisites cover these sufficiently.

Assignments

- The total points for this semester will be 2000. I expect some students to skip the second programming assignment; it will be doable by everyone but programming experience may make

the difference here, though the post-145/146 information will be covered in class and/or provided by video.

- Quizzes 25%: Online quizzes (2 attempts) and homework uploaded to Blackboard (Bb). About half of the midterm and final will be multiple choice questions inspired by these questions, mostly, so it is worth doing this right, versus gaming it. You will have 60 minutes to complete the quiz.
- Midterm: 20%. Planned date is March 1st. It will be timed, closed book, closed notes.
- Final Exam: 30% (cumulative) This will be a cumulative final and about one third of the material will be pre-midterm material. These are likely to be "fusion" questions to make sure you can combine earlier with later material.
- Labs/Programming: 25% 4 lab assignments of varying weights. We will use Java. Rules:
    - Does not compile or goes into infinite loop/recursion/out of memory: 0% if we cannot get running quickly
    - Your first programming assignment will be 140 points and be mandatory.
    - The next three will consist of a programming assignment, followed by two labs. You are expected to do 2 of the 3 at 180 points each. If you do all three, not including the first, then the lowest will get scaled down to 100 points, i.e. scaled down by 5/9th but added to your score without effecting the denominator, **but only if the score reaches 60 after scaling or more**. This is the extra credit for the class. You will have to do this yourself when calculating the grade.
    - Please run your code one last time before submitting it.
    - Programming Assignment Questions (by email to me oreillyj@cse.sc.edu): We will talk about programming early in the semester, when the first assignment is assigned. If you have a technical issue, please email me or the TA as much information as you think might be necessary to troubleshoot, i.e. please screenshot or at least copy the error with a small explanation and give your code.

    - Technical Difficulties (email me at oreillyj@cse.sc.edu):
        1. Please screenshot or at least copy an error with a small explanation.
        2. If it is an issue preventing submitting on time, I expect you to zip your submission and send to Sabbir (mailto:msaadat@email.sc.edu). Copy me at oreillyj@cse.sc.edu. Again, we will talk more about programming early in the semester, when the first assignment is released.
- Late Penalty for non-exam assignments: -15% of the total points per day late, up to two days late. Not accepted after that. Do not wait until the last minute. Please do not send me frantic emails if you have to submit a few minutes late, we will ignore the late penalty for assignments up to a little late, where "a little late" means about one hour late.
- There is an opportunity for a *significant (altogether about half a letter grade)* of extra credit throughout the semester. Test Security and Honor Code

All students will be expected to follow the Honor Code for all assignments and explicitly state that they have done so for the Midterms and Final.

Grading Schema

Item	Weight
Quizzes	25%
Midterm	20%
Final Exam	30%
Lab Assignments	25%
Extra Credit	<=5% (see lab/programming section)

Grading Policy (rounding 0.5 up); Scale: A (90-100%), B+ (86-89%), B (80-85%), C+ (76-79%), C (70-75%), D+ (66-69%), D (60-65%), F (0-59%)

Important: <u>You are responsible for checking your grades in Blackboard regularly</u>. No grading issues for an assignment will be addressed more than two weeks after a grade has been entered, or less as the end of the semester comes, where I will likely send reminders for you to check. I or the grader will never leave something they have graded blank – there will be either a zero or some other numerical grade and some feedback explaining the score, though zero certainly means we don't see any work and doesn't require a comment, obviously.

Side note regarding Blackboard grades: The grades on Blackboard are "raw". I will do the *minimum* manipulation there to preserve the data's integrity. I won't go back and do explicit drops, scale grades for extra credit, or anything like that. I want you to see exactly what I will use to calculate your grade, "unfiltered", as errors are almost always per assignment.

***Roughly* calculating your grade is fairly easy with total points; just take your points earned and divide by points assigned (2000 at the end of semester). If you want to get a more accurate idea of where you stand, remember a few things:**

1. **If you are "dropping" a programming/lab assignment (not the first one, however), remove any points granted from the denominator (or 0 points if not attempted) and the 180 points from the denominator.**
2. **If you do all three of the last three 180 lab/programming assignments, just subtract 4/9*<the lowest score> from the numerator if the score is 60/100 (after scaling down) and the full 180 points from the denominator; don't remove the same assignment from the denominator (case 1) twice, obviously. If the extra assignment doesn't meet the 60% threshold follow 1, above, instead.**
3. **Remember, your lowest quiz score will be dropped and not added to the final score in any way, so if you are "planning" on dropping a quiz, reduce the denominator by 100 points and remove whatever points granted from the numerator. Again, all "dropping" will be done automatically in a spreadsheet… you won't see it in Blackboard.**

Attendance Policy

"Attendance" means reviewing the material and keeping up with the course. Students are expected to be responsible and keep up with the material. If there is a sudden issue, e.g. from illness or injury, notifying me as soon as possible is best for all involved. I do not track attendance but presence in the

classroom and logging into Blackboard and viewing material (which can be tracked) are things that can factor into an assessment of whether a student has done their part to keep up with the material.

Accommodating Disabilities

Reasonable accommodations are available for students with a documented disability. If you have a disability and may need accommodations to fully participate in this class, contact the Student Disability Resource Center: 803-777-6142, TDD 803-777-6744, email sasds@mailbox.sc.edu, or stop by LeConte College Room 112A. All accommodations must be approved through the Office of Student Disability Services. See https://www.sa.sc.edu/sds/. If you need extra time for quizzes, please register with SDRC and I'll configure your quizzes appropriately. Tests should probably be taken with SDRC.

Honor Code

The Honor Code is a set of principles established by the University to promote honesty and integrity in all aspects of the campus culture. It is the responsibility of every student at the University of South Carolina to adhere steadfastly to truthfulness and to avoid dishonesty in connection with any academic program. A student who violates, or assists another in violating the Honor Code, will be subject to University sanctions. (See http://www.sc.edu/policies/ppm/staf625.pdf)

All Cheating will result in a grade of zero for the assignment and a referral to the Office of Academic Integrity. Cheating on a test/exam or any combination of assignments totaling 10% or more of the final weighted grade is grounds for failure in the entire course, pending a final ruling by Academic Integrity.

Regarding programming assignments:

- Code will be run through Moss whenever possible.
- There are two central, conflicting issues with coding assignments:
    - Answers or partial solutions to assignments are often available, even outside of "cheating" sites.
    - The Internet provides a deep bank of resources for many problems, libraries, etc. that you may find *legitimately* useful for learning some topic/technology or applying what you have learned to a new problem.
- Accordingly, a blanket ban on searching the Internet is unreasonable but then so is searching the Internet for an exact or even substantial partial solution to a given problem. Some situations are clearly cheating, others just using what resources you have at your disposal, and still more are somewhere in between. Here are the guidelines we will use for this course:
    - All code that is "paraphrased" from another source must be cited.
    - You should never need to cite more than a small amount of code, i.e., the next few following, obviously related lines.
    - A citation for a large region of code that trivializes a problem is no protection against an allegation of violating the Honor Code. **A resource that offers too much of a solution to a given problem is too specific and should never be used or read**. Additionally, if we must fairly "subtract" the large region's code from your work, then you are certainly better off just not turning it in, leaving that part blank, or developing a partial solution

that demonstrates what you *do* understand. It will be assumed that a student who submits a large block of code, even if cited, merely hoped we would not notice. Some assignments may be graded by script and it is expected that all students always follow the Honor Code, regardless of whether the professor or their teaching assistant is looking.

  - You must understand all code submitted. You may be called upon to explain it either your professor or their Teaching Assistant if we are concerned there may have been an Honor Code violation.
  - In general, smaller "snippets" are fine, assuming you understand how they work and so long as they do not trivialize a problem. A "snippet" should be a *small*, *cohesive* piece of code to solve a *single* programming task. Learning the ways other, likely more experienced, coders use a language or library is a good way to become fluent in a programming language and its libraries and idioms. You should not feel a need to obfuscate a single line of code if it matches what you need, given a citation, but if it is more than a very small number of lines (1-3), then that is much more suspect. You should never copy-paste code into your solutions.
  - Whether a region of code is evidence of cheating or not will generally be assessed based on:
    - Its size, i.e., whether it is a "snippet" or stolen *logic*, such as a whole algorithm.
    - Whether it trivializes a given problem, i.e., how much of the given problem is solved by your source. This is especially hazardous for *simpler* problems.
    - How similar it is to another programming solution, either a fellow student's or something online. Several faculty members have a Chegg account for <reasons>

- Collaboration: It is perfectly fine to talk in general terms with your peers, i.e. talk about language features, the basics of an algorithm to be implemented, etc., but in general, students working on individual assignments should almost never look at one another's code. **One easy rule: A student who is not done with an assignment should never view another's source code for that assignment.** If one student decides to help another then referring to general resources on an algorithm, demonstrating on pen and paper or whiteboard, or referring to programming tutorials/StackOverflow is certainly more illuminating than the final answer. This statement on collaboration is largely paraphrasing the following link and you can see there https://integrity.mit.edu/handbook/writing-code#:~:text=Search-,Writing%20Code,inline%20comment%20in%20the%20code for more examples. One exception is when someone is stuck and the peer viewing their code cannot use the information, e.g. the helper is *already done* and all the helper is doing is, e.g., finding a small syntax error or helping with a small region of code, especially if the helper is not viewing *their own* solution while helping. The two submissions will almost certainly be different if there was not too much help given and the person being helped should then be able to fully reconstruct and explain their solution.
- If I feel any assignment, programming or otherwise, is an Honor Code violation and have any doubt, I will consult a colleague and see if they agree. If so, then the case will be forwarded to the Office of Academic Integrity at https://www.sc.edu/about/offices_and_divisions/student_conduct_and_academic_integrity/index.php .

- When in doubt: Ask us whether what you are considering doing is acceptable. A “no, try looking at the problem like…” is much better than later being found to have plagiarized another’s work.
- On a somewhat lighter note, How to Cheat: https://github.com/genchang1234/How-to-cheat-in-computer-science-101