## 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. 17th, Monday
  + Midterm 1: March 2nd
  + Midpoint of Semester: March 3rd
  + Spring Break March 6th-13th
  + Last day to drop a course or withdraw without a grade of "WF" being recorded: March 28th
  + Final Exam: April 29, Friday, 12:30 p.m.

## Final Exam Times (official times, according to Registrar)

|  |  |
| --- | --- |
| **Section** | **Final Exam Time** |
| 416-002 | April 29, Friday, 12:30 p.m. |

Instructor

James O’Reilly ( [oreillyj@cse.sc.edu](mailto:oreillyj@cse.sc.edu) )

### Teaching Assistant

Moh Sadaat (“Sabbir”) ( [msaadat@email.sc.edu](mailto:msaadat@email.sc.edu) )

* Specific questions related to the grading or submission of a particular assignment go to the TA, though please copy me. All others go to me.

## Office Hours

All (my) office hours are posted here: <https://cse.sc.edu/~oreillyj> . If you need to meet outside these times, please email me at [oreillyj@cse.sc.edu](mailto:oreillyj@cse.sc.edu) . We can meet F2F or on Teams but posted office hours will generally be for those physically at my office. 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.

The college provides licenses for VMware, which will be optional, or lab machines for those without enough hard drive space on their machines. Students may remote in over VPN to the department lab machines. Instructions will be given early in the semester. This allows you to have access to the desktop of a lab machine, using Ubuntu.

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 (Approximate)

|  |  |
| --- | --- |
| Week | Topics |
| Week 1 | Intro, Chapter 1 |
| Week 2 | Chapter 1, Begin Chapter 2 (Application Layer) |
| Week 3 | Chapter 2 (Application Layer) |
| Week 4 | Chapter 3 (Transport Layer) |
| Week 5 | Chapter 3 (Transport Layer) |
| Week 6 | Chapter 3 (Transport Layer) |
| Week 7 | Chapter 4 (Network Layer) |
| Week 8 | Chapter 4 (Network Layer) |
| Week 9 | Chapter 4 (Network Layer), Midterm |
| Week 10 | Chapter 5 (Link Layer) |
| Week 11 | Chapter 5 (Link Layer) |
| Week 12 | Chapter 5 (Link Layer) |
| Week 13 | Chapter 6 (Physical Layer) |
| Week 14 | Special Topics, Review, “Flex” Time |

Note: A Week corresponds to (2) lectures and may not always line up with Monday and Wednesday of the same week

### 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.

• 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, and Ubuntu Linux.

### Minimal Student Technical Requirements

Students should be familiar with Java and have basic Linux Operating System, Ubuntu preferably, skills. The prerequisites cover these sufficiently.

## Assignments

* Homework & Quizzes 40%: Online quizzes and homework uploaded to Blackboard (Bb). I will average your highest and lowest quiz grade and replace the lowest with the average of the two.
* Midterm: 15%. Tentative date is March 2nd. It will be timed, closed book, closed notes.
* Final Exam: 25% (semi- cumulative) This will be a cumulative final and about one third of the material will be pre-midterm material.
* Labs/Programming: 20% 3 lab assignments of varying weights. We will use Java. Rules:
  + Does not compile or goes into infinite loop/recursion/out of memory: 0%
  + Please run your code one last time before submitting it.
  + **Important:** I *may* use scoring scripts, which output a score. These are thin wrappers for unit/integration tests which give you a *floor* for your grade if you satisfy the requirements. Notes on grading:
    1. Your grade may go *up* if there’s significant work. You should not expect more than 50% of the points back on anything incomplete with the exception of monolithic assignments (where you solve a single problem), but that’s contingent on your work, documentation, and possible unit tests you provide.
    2. Your grade may go *down* if you somehow pass a test and either fail to satisfy the requirements, such as “use threading to implement a program” or “solve Problem X in O(n2) time” and do not. It is possible but too cumbersome to test this so we may just look at your code (and the unit tests have times as well).
    3. Anything that can be construed as “cheating” the test will just get a zero for the *whole* assignment, e.g. remembering the expected outputs to a short list of inputs and just recording them.
  + We use GitHub and this is a *good* thing for all of us but I expect you to use it for backups:
    1. If you solve even *part* of a problem and take a break or whatever, then go ahead and push. These will be individual assignments, so you don’t have to worry about breaking other people’s code.
    2. You should, at a minimum, *look* at your final submission on GitHub. Cloning to a new directory and running all tests will probably be as quick and less error prone once you are accustomed to working with git on the command line.
    3. “I lost my code” or “My hard drive died” is not an excuse for losing a lot of work. Backing things up is part of our job. (Please let us know if you encounter major technical difficulties at some point but this is mostly for students who get a zero because of a git mistake *early* in the semester.)
    4. Don’t worry about GitHub if you’ve never used it. Our usage will be kept simple.
  + Programming Assignment Questions (by email to me [oreillyj@cse.sc.edu](mailto:oreillyj@cse.sc.edu)): We will talk about programming early in the semester, when the first assignment is assigned.
    1. Please screenshot or at least copy the error with a small explanation.
    2. Please push to GitHub before sending the question. Include in your email a hyperlink to your repository. Some questions can be answered by just looking at your code (easy on a smartphone) or perhaps running it. Both are made easier by just sending me a link. Your link, easily copied from GitHub will generally have the form of

**https://github.com/csce416-spring22/program-<NUM>-<GH Username>**

* <NUM> is some number 1,2,…
* <GH Username> is your GitHub username, which is often *not* the same as your UofSC username.
  + GitHub/Technical Difficulties (email me at [oreillyj@cse.sc.edu](mailto:oreillyj@cse.sc.edu)):
    1. Please screenshot or at least copy an error with a small explanation.
    2. Please give your GitHub Username and, if you are not using your UofSC email, your UofSC username.
    3. 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](mailto:oreillyj@cse.sc.edu). We will talk more about programming early in the semester, when the first assignment is released.
* Late Penalty for lab/homework assignments: -15% 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 may be *some* opportunities for a *significant (altogether about half a letter grade)* of extra credit throughout the semester.
* I am going to use total points (summing to 2000) this semester.
  + Quizzes (~4) will be 100 points each.
  + Homework (~4-5) will be about 80-100 points each. I will likely just hand you parts similar to old tests so you can do the non-multiple choice material before sitting for a closed book/note exam.
  + Lab Assignments (~3) will be about 400 total points in total, of varying weights.
  + Midterm and Final Exam will be 300 and 500 points each, respectively.
  + Looking at “Total” Points or “Weighted Total” in Bb may be helpful but not 100% accurate.

### 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 |
| Homework/Quizzes | 40% |
| Midterm | 15% |
| Final Exam | 25% |
| Lab Assignments | 20% |
| Extra Credit | ~5% |

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: You are responsible for checking your grades in Blackboard** **regularly. 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 send reminders for you to check. I 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 I don’t see any work.**

**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 to 2200, 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. It’s not hard to reduce the denominator by 50 points and then do the division to get a “current” grade.**

## 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, activity in the forums, 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/>.

## 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](https://theory.stanford.edu/~aiken/moss/) whenever possible.
* There are two central 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 large 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 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, 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.
* 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 very 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>