COMP 231 — Intro to Computer Organization — Fall 2024

Instructor:	Nate Phillips
Times:	11:00 AM-12:15 PM Tuesday/Thursday
Classroom:	Briggs 119
Course website:	http://www.cs.rhodes.edu/231 or https://ncp38.github.io/cs231-f24
Email:	phillipsn@rhodes.edu
Office:	Briggs 210
Office hours:	Tues/Thurs 3:15-4:15 PM & Wed 1-3 PM Also available by appointment and via Zoom

Official Course Description: This course is a bottom-up exploration of the interaction between computer hardware and software. Topics begin with an introduction to digital logic, and continue through elementary processor design, to assembly language, machine data representation, and computer arithmetic. Other topics in contemporary computer architecture such as multicore processors and GPGPU's may be discusses as time allows. Projects include hardware and simulated digital circuits as well as programs in assembly language.

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

- convert between decimal, octal, and hexadecimal number systems and perform computations involving signed binary number systems.
- perform combinational and sequential logic circuit analysis and design.
- demonstrate the ability to design, construct, test, and debug logic circuits in a laboratory environment.
- demonstrate an understanding of the purpose, behavior, and relationships between the basic components of a Von Neumann machine architecture.
- demonstrate a basic understanding of low-level programming by completing programs that require multi-level control flow, addressing modes, and both special and general-purpose registers.
- define and explain the process of translating, linking, and loading a program.
- demonstrate an understanding of the relationship between a high-level language and a low-level assembly language.
- demonstrate an understanding of the trends of technology migration from the higher to lower ends of computing.

Required Textbook: Null & Lobur, *The Essentials of Computer Organization and Architecture*, 5th edition (4th edition is okay)

Prerequisites: The course assumes successful completion of CS141 or significant programming experience. Please come see me if you have not had CS141.

Classroom Technology:

- **Canvas:** Course content will be hosted on Canvas, available at https://canvas.rhodes.edu.
- **Hardware Construction/Assembly Language:** This course uses educational hardware boards that use Field- Programmable Gate Arrays to act as programmable devices. These devices are available in the hardware lab in the basement of Briggs.

Coursework:

	Tentative weight	Tentative date
Labs	40%	
Homework	10%	
Exam 1	15%	Tuesday, October 29, in class
Exam 2	15%	Tuesday, November 26, in class
Comprehensive final exam	20%	

Grades of A–, B–, C–, and D– are correlated with final course grades of 90%, 80%, 70%, and 60%, respectively. If your final course grade falls near a letter grade boundary, I may take into account participation, attendance, and/or improvement during the semester.

Assignments are due at 11:59pm unless otherwise specified. **Late work is unprofessional** and will generally be **penalized at a rate of 8% per 24 hours late** or 1 point per day for homework assignments. If the assignment is submitted three days after it is due, it will not be accepted except in special cases, which might include sickness, injury, or other unforeseen events. In these situations, you should contact the professor as soon as is feasible to come up with a plan for submission.

Office Hours: In addition to regular office hours, I am often available immediately before or after class for short questions. You don't need an appointment to see me during regular office hours; you can just come by.

Don't be shy about coming by my office or sending me emails!! My goal for this course is to help you understand course concepts and reach your full potential. I think proactive communication is important and necessary in achieving this goal.

Attendance: Prompt attendance is expected for each class and is necessary for your success. If you know ahead of time that you will be absent from class for any reason, please discuss this with me at the beginning of the semester or as early as possible. Otherwise, if your attendance deteriorates, you may be referred to the dean and asked to drop the course. Note that in-class assignments or bonuses, if missed, will not be available to retake.

Attendance and participation may be considered when assigning a final grade. Attendance will be checked each class lecture period, with consequences for **excessive absences (more than five)**. For each additional absence, points will be deducted from your grade, up to and including **dropping a letter grade**.

Workload: It is important to stay current with the material. You should be prepared to devote at least 2–3 hours outside of class for each in-class lecture. In particular, you should expect to spend a significant amount of time for this course working in the lab trying example programs and developing programming assignments. For the labs, it is expected to take **an average of 15 hours** to complete. The number of available machines in the lab is limited, and availability is not guaranteed on a particular day or time. It is recommended to start and complete these assignments early; do not wait until the last minute to start your programming assignments.

Even with intentional focus and preparation, it is possible for issues to come up that delay assignment submission. As such, a **once-a-semester extension** is available upon request, allowing for up to 48 hours' grace for submissions.

You are encouraged to form study groups with colleagues from the class. The goal of these groups is to clarify and solidify your understanding of the concepts presented in class, and to provide for a richer and more engaging learning experience. However, you are expected to turn in your own code that represents the results of your own effort.

Getting Help: Students can get help with their coursework several different ways. The first place to reach out for help is to come to office hours. Scheduled office hours are listed on Canvas, the class website, and this syllabus. Additional meeting times are available by request. You may also email the instructor.

There will be tutoring available five nights a week specifically for this course. More information will be provided once the tutoring schedule is determined.

Rules for Completing Assignments Independently

- Unless otherwise specified, assignments handed in for this course are to be done *independently.*
- Talking to people (faculty, other students in the course, others with programming experience) is one of the best ways to learn. I am willing to answer your questions or provide hints if you are stuck. But when you ask other people for help, sometimes it is difficult to know what constitutes legitimate assistance and what does not. In general, follow these rules:
  - **Rule 1: Do not look at anyone else's solution for the same project/lab, or a different project that solves a similar or identical problem.**
    Details: "Anyone else" here refers to other members of the class, people who have taken the class before, people at other schools enrolled in similar classes, or any code you find online or in print. "Similar or identical problem" here should allow you to look at code that uses techniques applied in different situations that you can then adapt to your project. However, if you find yourself copying-and-pasting code or directly transforming code line by line to fit into your program, then that is considered plagiarism.
    Exception: You may help someone else debug their assignment, or seek assistance in debugging yours. However, this requires the person writing the assignment being debugged to have made a good-faith attempt to write the program in the first place, and the goal of the debugging must be to fix one specific problem, not re-write something from scratch.
  - **Rule 2: Do not develop or write code with anyone else.**
    Details: You must make a good faith effort to develop and implement your ideas independently before seeking assistance. Feel free to discuss the assignment *in general* with anyone else before you begin and as you're developing your program, but when you get to the level of writing code or pseudocode, you should be working independently.

  The underlying idea is that you can seek assistance in ways which will genuinely help you to learn the material (as opposed to just getting the assignment done). Programming assignments are graded as a benefit to you; they are your chance to show what you have learned under circumstances less stressful than an exam. In return, I ask that your work fairly reflect your understanding and your effort in the course.

Class Conduct:

- I encourage everyone to participate in class. Raise your hand if you have a question or comment. Please don't be shy about this; if you are confused about something, it is likely that someone else is confused as well. Teaching and learning is a partnership between the instructor and the students, and asking questions not only helps you understand the material, it also helps me know what I'm doing right or wrong.
- Do not use your cell phone while in class, and keep the ringer on silent.
- If you cannot make it to class for whatever reason, make sure that you know what happened during the lecture that you missed. It is your responsibility, and nobody else's, to do so. The best way to do this is to ask a classmate.
- If you have to leave a class early, inform the instructor in advance. It is rude to walk out in the middle of a lecture.

Additional policies: On the class webpage, there are additional class and college policies covering accommodations, academic integrity, diversity, sexual misconduct disclosure, and recording lectures.