CS3 Introduction to Software Design

COMP 3081, Fall 2025

Section 001
Monday, Wednesday 12:40-2:05 p.m.
Fogelman Classroom Bldg (FCB) 131

https://memphis.instructure.com/courses/179277

Section 002
Tuesday, Thursday 9:40-11:05 p.m.
Psychology Bldg 206

https://memphis.instructure.com/courses/179789

Instructor: Katie Bridson <kbridson@memphis.edu>
Office Hours: In person: Mon 10-11am, Th 1-2pm, Fri 10-11am
Use course Teams to receive help outside of in person hours
Office: Dunn Hall 303; meetings held in Teams by appointment

Teaching Assistant: Miguel Perez <meperez@memphis.edu>
Md Muminul Hossain <Md.Muminul.Hossain@memphis.edu>
Cong Guo <cguo@memphis.edu>
Lakshmanasiva Nunna <lnnunna@memphis.edu>

Please begin all email subject lines with the course number in brackets, then your topic (e.g., [COMP 3081] Missing class 9/7).

1 Catalog Description

COMP 3081 - CS3 Introduction to Software (3)
Full-stack development; current technologies for software development; event-driven programming in the model-view-controller design; regular expressions; declarative programming and object-relational mapping; syntax, lexical analysis. PREREQUISITE: COMP 2150, or permission of instructor

2 Topics and Learning Objectives

The course will cover the following technical topics. Across these areas, students will build overarching skills in problem solving, software design, testing, and professional collaboration.

- **Unix-like Environments and the Command-Line**
  - Setting up the environment
  - Basic Unix commands and file system
  - Scripting and creating custom commands (Zsh, Python)
  - ***Objective****: Use the command line to navigate, automate, and solve practical computing problems.*
- **Version Control Systems with Git and GitHub**
  - Git data structures and basic commands
  - Creating and managing individual repositories

    - Branching and merging
    - Team repositories and collaboration
    - ***Objective**: Apply version control to manage code history and coordinate effectively on software projects.*
- **Programming Paradigms**
    - Functional programming (Python)
    - Procedural programming in Ruby
    - Object-oriented programming in Ruby
    - Object-oriented design principles
    - UML class diagrams
    - ***Objective**: Write programs in multiple paradigms and select appropriate approaches to implement maintainable solutions.*
- **Automated Testing with Ruby and RSpec**
    - Testing frameworks
    - Unit testing fundamentals and strategies
    - ***Objective**: Design and apply tests to verify correctness, prevent regressions, and improve code quality.*
- **Regular Expressions**
    - Writing basic regular expressions (Rubular)
    - Parsing documents with regex (Ruby)
    - ***Objective**: Apply pattern-matching techniques to analyze, extract, and transform text data.*
- **Web Communication**
    - HTTP requests and Web APIs (JSON)
    - HTML and CSS
    - JavaScript: parsing and manipulating the DOM
    - Hosting static websites on GitHub Pages
    - ***Objective**: Build interactive client-side applications and integrate them with external web services.*
- **Full-Stack Development with Frameworks**
    - Web servers and framework basics
    - Model-View-Controller (MVC) design pattern
    - Object-relational mapping (ORM)
    - Hosting a web server (Render)
    - ***Objective**: Construct and deploy full-stack applications using modern frameworks and hosting platforms.*

3 Course Technology

3.1 Required Equipment

Students will be required to bring a laptop computer to lecture that can reliably run programming tools and a development environment.

Minimum System Requirements:

- **CPU:** Dual-core (Intel i5 / AMD Ryzen 3 / Apple M1)
- **RAM:** 8 GB
- **Storage:** 50 GB+ free

- **OS:** macOS 14+, Windows 11 (64-bit)+, or Linux (Ubuntu 22.04+)
- **Battery:** At least 2 hours of use on a full charge
- **Connectivity:** Stable Wi-Fi and ability to connect to campus networks

Other Considerations:

- A functional webcam and microphone are required for remote activities.
- The device must be capable of running the development environment and screen sharing software simultaneously without significant performance degradation.
- Tablets, Chromebooks, and iPads do not meet the requirements for this course, as they cannot run all necessary development tools.

It is the student's responsibility to have and maintain a working laptop development environment to use for this class.

If you do not have a suitable machine in your possession by midnight on the first day of class, I strongly recommend that you postpone taking the course until you do.

3.2 Required Software Tools

University Email: Check daily. Used for official or record-keeping communication (e.g., attendance issues, grade inquiries, or other administrative matters).

Microsoft Teams: Used primarily for course announcements and getting help. You can configure email notifications for missed messages and activity when you are away from the app. This is your best option for posting technical questions or getting help with assignments in the evenings.

Canvas: Your main source for assignment instructions, due dates, submissions, grades, and feedback. Canvas announcements will not be used in this course; announcements will be posted in Teams. Do not use Canvas comments to ask questions about a grade you received; email the instructor *and* TA directly.

GitHub: Private GitHub repositories under the course GitHub organization will used as part of the deliverables for some coding assignment submissions.

4 Required Textbooks

There are no required textbooks for this course. The instructor will provide all reading materials.

5 Evaluation

Grading weights are as follows:

- 15% Participation Activities
- 35% Skills Assignments
- 40% Exams (midterm & final @ 20% each)
- 10% Projects (midterm & final @ 5% each)

To convert from percentages to letter grades, see the following table:

≥ 97%	91–96%	89–90%	87–88%	81–86%	79–80%	77–78%	71–76%	69–70%	67–68%	60–66%	≤ 59%
A+	A	A-	B+	B	B-	C+	C	C-	D+	D	F

I reserve the right to *lower* the percentage threshold for letter grades as I see fit (i.e., I may make the grading scale better for you but never worse).

5.1 Attendance and In-class Participation Policy

Attendance is expected for all scheduled class sessions. Active participation in class during lectures and activities is crucial for your success and skill development in this class.

If you miss or know you are going to miss class, contact the instructor as soon as possible—preferably before the absence. If you believe the absence may be excused, provide documentation as required (see Section 5.7)

5.2 Time Expectations and Study Habits

- **General Guideline:** Plan to spend **at least 2-3 hours working outside of class for every hour in class**. Typically, a new topic and assignment is introduced each class session, and you will need at least 2 hours to complete the assignment before the next class.
- **15-Minute Rule:** If you cannot solve or make progress on a problem within 15 minutes, ask for help—whether from classmates or the instructor/TA. This mirrors professional developer practices and prevents wasted effort.
- **Avoid Restarting Entirely:** Instead of discarding all progress when you hit a roadblock or ask for help, debug and iterate on your existing work. This approach cultivates problem-solving skills and reflects industry standards more accurately than "starting over."
- **Time Management:** If you struggle to balance coursework with other responsibilities, consider seeking out time-management resources or discuss strategies with the instructor. Consistent practice, efficient problem-solving techniques, and asking for help when stuck will help you succeed.

5.3 Assignments

Assignments are central to this course and make up 60% of your final grade. You cannot pass unless you do them! Assignments are designed to help you practice concepts, apply problem-solving strategies, and reflect on your learning. The assignments fall into three categories:

5.3.1 Lecture Participation

Class time consists of interactive lectures with active learning activities. You will be expected to:

- Follow along with the lecture material.
- Respond to short in-class prompts or questions after each major topic to deepen your understanding.
- Engage in discussions and ask questions to clarify concepts.

These activities are due by the end of class and count toward your participation grade. They cannot be made up if you are absent. If you miss an activity, refer to Section 5.6.1 for late work and absence policies. All participation activities are weighted equally and are not graded for accuracy.

5.3.2 Skills Assignments

Skills Assignments are medium-length independent practice activities completed outside of class. They focus on applying the technical skills introduced in lecture to new situations. Typical components include:

- Hands-on coding and problem-solving tasks.

- Documentation of your work (e.g., screenshots, code submissions, explanations).
- Responses to self-assessment and reflection prompts.

These activities are due before the next class meeting (typically the day after they are assigned). All Skills Assignments are weighted equally, and your grade is based on:

- **Execution**: Show that your solution works and is verified.
- **Reflection**: Explain clearly how you know your solution is correct (or where it went wrong).
- **Confidence & Plan**: Rate your confidence in the learning objectives and give a concrete plan to improve.

Each Skills Assignment allows up to one second-chance submission, as defined in the *Skills Assignment General Instructions & Rubrics* document.

5.3.3 Projects

The course includes two larger projects designed to integrate and demonstrate the skills you have developed through Skills Assignments. Projects are designed to integrate multiple skills and demonstrate more advanced competence. Unlike Skills Assignments, which usually focus on a single topic (e.g., writing unit tests), projects typically require combining several skills at once (e.g., developing object-oriented code that follows good design principles, includes automated tests, and is documented with UML diagrams).

- Projects are completed individually, unless otherwise specified.
- Expectations and grading criteria will be detailed in project-specific instructions and rubrics.
- Projects may include checkpoints or progress submissions to help you pace your work.

5.4 Exams

The course will have two exams: a midterm and a final. The final will be given during the official final exam period for each section (https://www.memphis.edu/registrar/calendars/exams/25f-final-exams.php). The date of the midterm will be announced at least 2 weeks in advance.

Exams will be proctored and given in-person during class time. They will be closed-everything (note, neighbor, course materials, IDE, AI, etc.). The use of any prohibited aids during exams will constitute academic misconduct. You must submit your exam before leaving the testing room to receive credit.

If you are unable to take an exam as scheduled for an excused circumstance (see Section 5.7), please let me know as soon as possible. Make-up exams must be scheduled and taken within 3 days following the originally scheduled exam. Be prepared to show some kind of documented proof of your situation.

The exams will be designed to evaluate student's mastery of the conceptual knowledge covered in the course.

5.5 Assignment Submission Policies

All assignment instructions and official due dates will appear in Canvas.

Note that some assignments have multiple parts with their own due dates. These will be listed within the Canvas assignment description and/or announced in Teams. Therefore, be aware that not all tasks will show up in Canvas's "Upcoming Deadlines"—you must track them yourself.

You may be required to submit work via Canvas dropboxes and/or by pushing (uploading) code to a GitHub repository, depending on the assignment.

5.5.1 Canvas Dropbox Submissions

- **Due Date:** Typically, **11:59 pm** on the stated day.
- **Grace Period:** Usually open until **5:00 am** the following morning, with **no penalty** for submissions in that window.
- **Closure:** After 5:00 am (or the stated grace-period end), your work is **late**.
- **No "Work-Arounds"**: Submissions sent by email or Teams or attached in Canvas comments *after closure* are invalid. If you are having temporary Canvas issues, you may send your submission to the instructor via email *before closure*.
- **Submitting Links**: For most assignments, you will be expected to upload files to the Canvas dropbox. However, since this is sometimes infeasible for code projects or large files, you may instead be asked to submit a link to an externally hosted file or GitHub repository.
    - **Grant Access to Graders**: You must provide access to the instructor and TA before the due date.
    - **Verify Your Link**: Always confirm your link is working before submitting. A link to a resource that does not exist will be treated as not submitted.
    - **No Modification After Submission**: Externally hosted files can sometimes be changed after submission. We will only consider work that was completed *before closure*. Timestamps and version histories may be checked to ensure compliance with deadlines.

5.5.2 GitHub Repository

- **Push Deadline:** Only commits pushed before the due date (or grace period end) count as on time.
- **Push Timestamp:** This is the official submission time. If a push-time exploit is discovered, I reserve the right to introduce new constraints.
- **In some cases, the version history may be important.** Following best practices, you should make small frequent commits to show how your submission evolved over time.

5.6 Late Work Policy

Any work not submitted before the due date or grace-period end (when applicable) is considered late. Late work is generally not accepted, except in cases of excused circumstances (see Section 5.7).

5.6.1 In-class Participation Activities

No late work accepted. If you miss such an activity due to an excused absence, that activity's grade will be dropped.
If you miss such an activity without an excused circumstance, you will receive a zero.

5.6.2 Skills Assignments

For both attempts, no late work will be accepted unless you have an excused circumstance.
If you miss the deadline without an excused circumstance, you will receive a zero.

If you miss the primary submission due to an excused circumstance, that assignment's grade will be dropped.

If you received a second attempt but miss that submission due to an excused circumstance, you can choose to have the assignment's grade dropped or take the first attempt grade.

5.6.3 Projects and Other One-Chance Assignments

No late work will be accepted unless you have an excused circumstance.
If you miss the deadline without an excused circumstance, you will receive a zero.

If you miss a primary submission due to an excused circumstance, I will consider the situation on a case-by-case basis and work with you to reach an appropriate resolution.

5.7 Acceptable vs. Unacceptable Excuses for Absences or Late Work

Late work will be accepted only for documented circumstances that prevent on-time submission. Below are some examples (not exhaustive).

Other Circumstances: If you face a situation not listed here, notify me as soon as possible (preferably before class) with any relevant documentation. I will review it on a case-by-case basis according to university policies.

5.7.1 Acceptable Excuses (with Documentation)

- Medical Emergencies: Serious illness or hospitalization, with a note from a licensed healthcare provider.
- Family Emergencies: Documented bereavement or critical family situations.
- University-Sanctioned Events: Official travel to conferences, athletic events, or other approved academic commitments (with documentation from an official university sponsor).
- Natural Disasters: Incidents like flooding or storms that cause campus closures or widespread power outages, with verifiable proof.

5.7.2 Unacceptable Excuses

- Forgot About the Deadline: "I didn't realize it was due today."
- Travel Plans: Personal vacations or early departures for holidays not recognized by the university.
- Procrastination: "I started too late" or "I had other assignments."
- Minor Technical Issues: Losing internet access briefly (unless documented by an IT service disruption that was severe and widespread) or forgetting a charger.

6 Academic Integrity

The University of Memphis expects all students to behave honestly. The Student Code of Rights and Responsibilities explains what constitutes a violation of our Academic Integrity policy. For more information, please see the Office of Student Accountability's website: https://www.memphis.edu/osa/. Plagiarism, cheating, and other forms of academic dishonesty are prohibited. Students who violate the academic misconduct policy, either directly or indirectly, through participation or assistance, are immediately responsible to the instructor of the class in addition to other possible disciplinary sanctions which may be imposed through the regular institutional disciplinary procedures.

Examples of academic dishonesty include, but are not limited to:

- **Cheating – A student uses a smart phone to access the internet while taking a quiz.**
- Copyright infringement – A student uses a photograph found on the internet in a presentation without obtaining permission from the photographer.
- **Deception – A student gives a dishonest excuse when asking for a deadline extension.**
- Denying access to information or material – A student makes library or shared resource material unavailable to others by deliberately misplacing those resources.
- Fabrication – A student invents data in an academic work.
- Facilitating academic misconduct – A student knowingly allows a portion of their work to be used by another student.
- Plagiarism – A student represents the ideas of another in a paper without citing and referencing the work or a student turns in the same or nearly the same assignment for credit in more than one class.
- Sabotage – A student prevents others from completing their work by opening a window to affect a temperature-controlled experiment.
- **Unauthorized collaboration – A student works with other students on a paper without the specific permission of the instructor.**

6.1 Course-Specific Policies

As per department policy, any student caught engaging in academic misconduct in the course will receive at minimum a 0 grade on the assignment/test and be reported to the Office of Student Accountability.

By the end of this course, you are expected to have developed competencies in the technologies, concepts, and skills we cover. This is critical for success in future courses, and most importantly so that you can get a job later! To build these skills, you must practice them yourself. As such, *all grade items (unless specifically indicated otherwise) must reflect your own individual effort.*

You are encouraged to seek help — from me, your classmates, tutors, or online resources — but assistance should always focus on helping you arrive at the answer on your own.

Submitting material that entirely copied from the Internet, received from another person, or automatically generated by an AI tool such as ChatGPT outside the limits permitted in Section 6.2 is considered plagiarism. Plagiarism is academic misconduct.

6.2 AI Tools (e.g., ChatGPT, Github Copilot) Policies

AI tools may be used in this course, but only in ways that support your learning. They are never a substitute for the critical thinking, reasoning, and logic skills required to design and implement programs. This course is designed to help you strengthen those skills, which are essential both in college and in your future career.

6.2.1 Exams

AI tools are prohibited on all exams. Using AI on an exam will be treated as academic misconduct. Exams measure what you can reason through and solve on your own. You may be asked to read and understand code on an exam and answer questions about what it does, but you will not be asked to implement an end-to-end solution.

6.2.2 Coding Assignments

AI tools may be used to assist with coding assignments, provided you remain in control of the problem-solving process. They should help you practice and improve, not replace your own reasoning.

Keep in mind:

- AI-generated code is often mostly correct but subtly wrong, and fixing those errors requires strong debugging skills.
- Output quality depends heavily on the clarity of your design and instructions.
- You should only rely on AI for solutions you can test, verify, and explain yourself.

Important note: Submitting AI-generated code you have not tested and cannot explain the design of will be treated as academic misconduct. Any code you submit must be code you designed, can understand, can explain, and can adapt if asked.

Example of maximum permitted use:

You are asked to write a Python script that checks if a string is a palindrome. You might reason through the problem like this:

- Decide the script will consist of a function, is_palindrome, that takes one string parameter and returns a Boolean value; the script will call the function on command-line input.
- Recall that a palindrome is a string that is the same forwards and backwards, which requires a comparison.
- Consider possible comparison approaches: compare the string with a reversed copy, use a while loop with two pointers starting from the front and back that compares characters moving inward until they meet in the middle, etc.
- Choose to use the while loop because it has reduced space complexity.
- Consider possible edge cases and decide what preprocessing may be necessary (e.g., ignoring spaces, case-insensitive matching).
- Define test cases (e.g., "racecar" → True, "Race car" → True, "hello" → False).

After making these choices, you might turn your design into a clear, specific request to AI that reflects your design:

> "Write a Python script that defines a function named is_palindrome which takes a single string argument and returns a Boolean. Before checking the string, the function should remove all spaces and convert the string to lowercase. Implement the logic using a while loop with two pointers: one starting at the beginning (left) and one at the end (right). The loop should continue while left < right. On each iteration, compare the characters at s[left] and s[right]. If they are not equal, immediately return False. After each comparison, increment left and decrement right. If the loop finishes without mismatches, return True. At the bottom of the script, include code that reads a string from the command line, calls is_palindrome with it, and prints the result."

This is acceptable because you supplied the full design: function name, parameter type, preprocessing rules, loop choice, stopping condition, return values, and test logic. The AI is only filling in syntax and boilerplate.

At this point, you must still confirm the program works: run it on your test cases, check the output, and debug or revise as needed. Debugging will require you to be able to read and understand the details and syntax of the implementation even if you did not write it yourself.

Important note: If you plan to pursue software development as a career, you will need to invest significant practice time in live implementation without AI before interviewing. If you are headed for another field, high-level design practice may be sufficient. In either case, make sure your approach in school reflects the skills you will need in your desired career.

6.2.3 Lecture Preparation

AI may be used to preview lecture material (e.g., generating overviews or clarifications from provided notes) to help you orient yourself before class.

However, you should not use AI to answer lecture comprehension questions that ask you to summarize, recall, or reflect on your current understanding. In these cases, the value lies not in the answer itself but in the act of:

- Attempting to recall information from memory,
- Noticing the gap between what you know and what you need to know, and
- Recognizing areas that require more practice.

Learning research shows that shallow practice (re-reading notes or asking AI to summarize) builds familiarity but not durable knowledge. In contrast, deep practice (retrieving, applying, and explaining knowledge) creates lasting understanding and transferable skills. The discomfort of this productive struggle is a signal of real learning taking place.

7 Classroom Behavior

Students should be aware of the Student Code of Rights and Responsibilities which describes examples of unacceptable classroom behavior. Disruptive classroom behavior will not be tolerated. Instructors are empowered to remove students from class and refer behaviors for sanctioning to the Office of Student Accountability.

8 Disability Resources for Students (DRS) Accommodations

Please see the instructor if you need accommodations for a disability, or to fulfill cultural or religious obligations. Students with requests for accommodations should contact Disability Resources for Students to register and learn about the services available to support their learning. Students with disabilities are encouraged to speak with us privately about academic and classroom accommodations. It is strongly encouraged that you register with Disability Resources for Students (DRS) to determine appropriate academic accommodations. Disability Resources for Students is located in 110 Wilder Tower, their phone number is (901) 678-2880 (V/TTY), their email is drs@memphis.edu, and their website is https://www.memphis.edu/drs/. Disability Resources for Students coordinates all accommodations for students with disabilities.

Qualified students with disabilities will be provided reasonable and necessary academic accommodations if determined eligible by the appropriate Disability Resources for Students staff at the University. Prior to granting disability accommodations in this course, the instructor must receive written verification of a student's eligibility for specific accommodations from the Disability Resources for Students staff at the University. It is the student's responsibility to

initiate contact with University's Disability Resources for Students staff and to follow the established procedures for having the accommodation notice sent to the instructor.

9 Mental Health

As a student you can sometimes feel overwhelmed, lost, experience anxiety or depression, and struggle with relationship difficulties or diminished self-esteem. Mental health challenges can interfere with optimal academic performance. However, many of these issues can be effectively addressed with some help. If you find yourself struggling with your mental or physical health this semester, please feel free to approach me. I will try to be flexible and accommodating. As your instructor, I am not qualified to serve as a counselor, but UofM offers confidential counseling services on-campus and via telehealth that are available to students taking six or more credits at no cost. UofM Counseling Center is staffed by experienced, professional psychologists, clinical social workers, and counselors, who are attuned to the needs of college students. I strongly encourage you to take advantage of this valuable resource. To connect with Counseling Center services, please visit 211 & 214 Wilder Tower, or call 901.678.2068. To know more about their services, you can visit their website at https://www.memphis.edu/counseling. In a crisis situation, please call 901.678.HELP (4357) to speak to the On-call counselor. Remember, getting help is an intelligent and courageous thing to do -- for yourself and for those who care about you.

10 Personal or Academic Challenges including Food & Housing Insecurity

If you are experiencing personal or academic challenges including, but not limited to food or housing issues, family needs, or other stressors, please visit the Dean of Students Office to learn about resources that can help. Any student who faces personal challenges including, but not limited to securing their food or housing and believes this may affect their performance in the course is urged to contact the Dean of Students Office at 901.678.2187 located in the University Center, Suite 359 for assistance. If you are comfortable doing so, please also let the instructor know you are experiencing challenges as they may be able to assist you in connecting with campus or community supports.