

4143 Programming Language Concepts

General Course Info

- **Days:** TTh 9:30 a.m. - 10:50 a.m.
- **Location:** BO 320
- **Semester:** Monday August 28nd - Friday December 8th
- **Holidays:**
 - **Labor Day** Monday September 4th
 - **Thanksgiving** Wednesday November 22nd - Sunday November 26th
- **Last Day for "W":** Monday October 30th
- **Last Day of Class:** Friday December 8th
- **Final Exam:** Tuesday December 12th from 8:00 pm - 10:30 pm

Office Hours

Dr. Terry Griffin
Fall 2023
ph: (949) 917-4029 email: tergriffin@uci.edu

	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3:00-4:00					
4:00-5:00					
5:00-6:00					
6:00-7:00					
7:00-8:00					
8:00-9:00					
9:00-10:00					
10:00-11:00					
11:00-12:00					
12:00-1:00					
1:00-2:00					
2:00-3:00					
3					

- How can our programs know about themselves? How can we answer questions about the code itself?

Semester Schedule

Weeks	Description
1-2	Introduction and Names, Binding, and Scope
<input type="checkbox"/>	Introduction to programming languages and their importance
<input type="checkbox"/>	Syntax vs semantics
<input type="checkbox"/>	Compilation vs interpretation
<input type="checkbox"/>	Names, identifiers, and keywords
<input type="checkbox"/>	Binding time: static, dynamic, early, and late binding
<input type="checkbox"/>	Scope: lexical vs dynamic scoping
<input type="checkbox"/>	Nested scopes and scope rules
<input type="checkbox"/>	Static and dynamic scoping examples
3-4	Evaluation (Expressions) and Execution (Control Flow)
<input type="checkbox"/>	Expressions and their evaluation
<input type="checkbox"/>	Precedence and associativity
<input type="checkbox"/>	Order of evaluation
<input type="checkbox"/>	Short-circuiting and its effects
<input type="checkbox"/>	Control structures: sequencing, selection, iteration
<input type="checkbox"/>	Conditionals: if, if-else, switch
<input type="checkbox"/>	Loops: while, for, foreach
<input type="checkbox"/>	Control flow pitfalls and examples
5-6	Types
<input type="checkbox"/>	Data types and their significance
<input type="checkbox"/>	Static typing vs dynamic typing
<input type="checkbox"/>	Strong typing vs weak typing
<input type="checkbox"/>	Type checking and type inference
<input type="checkbox"/>	Primitive types: integers, floating-point, booleans, characters
<input type="checkbox"/>	Composite types: arrays, records, tuples
<input type="checkbox"/>	Type compatibility and type coercion
7-8	Functional Abstraction (Subroutines and Coroutines)

Weeks	Description
<input type="checkbox"/>	Introduction to subroutines and functions
<input type="checkbox"/>	Function declaration, parameters, and return values
<input type="checkbox"/>	Call stack and activation records
<input type="checkbox"/>	Recursion and tail recursion
<input type="checkbox"/>	Higher-order functions and function composition
<input type="checkbox"/>	Introduction to coroutines and cooperative multitasking
<input type="checkbox"/>	Coroutines vs threads
<input type="checkbox"/>	Coroutine synchronization and communication
9-10	Data Abstraction (Objects and Modules)
<input type="checkbox"/>	Introduction to data abstraction
<input type="checkbox"/>	Object-oriented programming principles
<input type="checkbox"/>	Classes, objects, methods, and attributes
<input type="checkbox"/>	Inheritance and polymorphism
<input type="checkbox"/>	Encapsulation and information hiding
<input type="checkbox"/>	Introduction to modules and modularity
<input type="checkbox"/>	Module interfaces and implementations
<input type="checkbox"/>	Packaging, namespaces, and access control
11-12	Concurrency
<input type="checkbox"/>	Introduction to concurrency and parallelism
<input type="checkbox"/>	Threads vs processes
<input type="checkbox"/>	Thread synchronization and coordination
<input type="checkbox"/>	Race conditions and critical sections
<input type="checkbox"/>	Mutual exclusion and semaphores
<input type="checkbox"/>	Deadlocks and livelocks
<input type="checkbox"/>	Parallel programming models
<input type="checkbox"/>	Concurrent programming pitfalls
13-14	Metaprogramming
<input type="checkbox"/>	Introduction to metaprogramming
<input type="checkbox"/>	Macros and code generation
<input type="checkbox"/>	Reflection and introspection

Weeks	Description
<input type="checkbox"/>	Compile-time vs runtime metaprogramming
<input type="checkbox"/>	Template-based metaprogramming
<input type="checkbox"/>	Aspect-oriented programming
<input type="checkbox"/>	Language-integrated query
<input type="checkbox"/>	Examples of metaprogramming in various languages
15	Review and Future Trend
<input type="checkbox"/>	Recap of key concepts from the course
<input type="checkbox"/>	Discuss emerging programming language trends
<input type="checkbox"/>	Domain-specific languages (DSLs)
<input type="checkbox"/>	Metaprogramming and code generation advancements
<input type="checkbox"/>	Language support for parallelism and distributed systems
<input type="checkbox"/>	Language design challenges and opportunities

Adventures in GoLang

- ☐ Introduction to Go
 - ☐ Introduction to Go programming language
 - ☐ Setting up the Go development environment
 - ☐ Go syntax and basic program structure
 - ☐ Variables, data types, and basic operations
 - ☐ Control flow statements (if/else, loops)
 - ☐ Functions and packages in Go
- ☐ Data Types and Structures
 - ☐ Complex data types (arrays, slices, maps, structs)
 - ☐ Working with strings and characters
 - ☐ Pointers and memory management
 - ☐ Error handling in Go
 - ☐ File I/O operations
- ☐ Concurrency and Goroutines
 - ☐ Understanding concurrency in Go
 - ☐ Goroutines and channels
 - ☐ Synchronization and data sharing
 - ☐ Patterns for concurrent programming
 - ☐ Error handling in concurrent programs
- ☐ Object-Oriented Programming in Go

- ☐ Structs and methods in Go
- ☐ Encapsulation and data hiding
- ☐ Inheritance and composition
- ☐ Polymorphism and interfaces
- ☐ Object-oriented design principles in Go
- ☐ Error Handling and Testing
 - ☐ Error handling best practices in Go
 - ☐ Panic and recover mechanisms
 - ☐ Unit testing in Go (using the testing package)
 - ☐ Writing testable code in Go
 - ☐ Code coverage and test automation
- ☐ Advanced Topics in Go
 - ☐ Reflection and runtime type information
 - ☐ Concurrency patterns (e.g., worker pools, fan-out/fan-in)
 - ☐ Memory optimization and profiling
 - ☐ Benchmarking and performance tuning
 - ☐ Working with external libraries and APIs
- ☐ Web Development with Go
 - ☐ Overview of web development in Go
 - ☐ HTTP server programming in Go
 - ☐ Routing and handling requests
 - ☐ Middleware and authentication
 - ☐ Introduction to web frameworks (e.g., Gin, Echo)
- ☐ Final Projects and Wrap-up
 - ☐ Student final projects: Implement a significant Go application
 - ☐ Project presentations and feedback
 - ☐ Recap and review of course concepts
 - ☐ Q&A session and open discussion

Grading

Will be based on number of problems solved. Full credit for on time solutions. Half credit for late submissions. Some credit for accepted solutions with issues (e.g. presentation errors),

Categories	Portion of Course	:::	Letter Grade	Grade Range
Exams	45%	:::	A	90-100
Github	10%	:::	B	80-89
Participation	5%	:::	C	70-79
Presentation	10%	:::	D	60-69

Categories	Portion of Course	:::	Letter Grade	Grade Range
Project	10%	:::	F	below 60
Final	20%	:::		
		:::		

Teaching Philosophy

I approach each course with the mindset that every class has a different set of students and not every lecture or list of topics should be given in a "lock step" manner tied to a calendar. I try and incorporate current industry topics, student questions, and student interests into my everyday lectures. I will always cover the core content of the course, but it just may not be in the same order, or using the same canned examples for each topic every time. A single question may send our lectures off on a path not previously planned.

I call this my "problem based approach" to teaching. Usually a question is dealing with a specific problem, so I tailor my lectures to incorporate topics necessary to solve the problem. It also means topics do not get delivered in the same order every semester. Of course, I guide the solution we use to keep in line with content commensurate with overall course objectives.

This method of content delivery is not for everyone. But based on the vast majority of course evaluations most students do enjoy my lecture style and content delivery. Having said this, I know my methods are not perfect and not every student responds positively. To alleviate distress for those students I create study guides for each exam. This way no matter what order content is delivered, they have a concrete list of topics whereby the exam is a subset of those topics. In fact, I give the students more than just a list of topics to study, I also provide example exam questions to that can be answered as part of the study process.

How to Succeed

I encourage every student to firstly just **go to class!** Beyond that, attempt to participate in class discussions and also ask questions in class. Believe me, classes that have many student questions with a subsequent discussion, seem do much better in understand a topic as a class.

More importantly you should interact with your fellow students outside of class as well. Start a study group, post links on Slack, ask questions on Slack! Stay involved with your classmates. Also, if you see posts on slack, respond to these posts either with a text response or an emoticon reaction to the post. No one likes to post something, and then feel as no one has read it.

If you need help in understanding a topic, you need to message me immediately so I can either text you a response, or zoom with you to get things cleared up. And yes I will zoom late into the evening if that is necessary. I'm not saying I'm at your beckon call, but I try to make myself very available.

Lastly, this isn't totally about your own success, but it helps me succeed. Asking me questions with direct message or on the course channel is a huge help! I can turn that question into (what the military calls) an "overhead correction", meaning I can clarify something to the entire class. It does not mean I will use your name (unless you publicly post it, then that is on you 😊), but it does give me the opportunity to clarify

things to the entire class, since most questions tend to be what the majority of students in the class are thinking.

My View on Cheating / Plagiarism

- Most plagiarizing, when it comes to programming, happens for two reasons:
 - 1. You don't have a clue how to solve the problem, so you get a friend or the internet to help.

This can be ok, if you cite your resource, and only find small snippets of code to work into your own solution.

- 2. You didn't start early enough, and you're desperate to get something working the night before it's due, so you get a friend or the internet to help.

This is never ok.

- Both are easy to fix.
 - 1. Come ask me to explain. I promise you're not the only one who is confused.
 - 2. Start early. Then when you get stuck, you can ask for help the right way!

Presentations

- Presentations are a major component of your course work. The ability to discuss complex topics in front a group of your peers is an important skill to have.
- Depending on the course, and the size of the class, you may have many presentations, or just one presentation.
- The quality of a presentations that accompanies a programming project is highly coupled with the quality of your project. A poor project makes it hard to give a proper presentation on a project in which you did not complete.
- On the other hand, an excellent project doesn't ensure a great presentation either.
- **Preparation is key**, and I am **ALWAYS** available for help with presentations.
- I will give specific requirements for each presentation since each project may vary greatly, but in general project presentations in my course should follow a basic outline:
 - Project description (if necessary)
 - A logical progression of your steps in implementing the project. Make sure to include:
 - Pitfalls (any confusing components that gave problems)
 - Highlights (any good solutions or components you are proud of)
 - Summarize the results or final product whether it be the completed features of the project or the results of any data you processed.
- Be prepared! Sometimes showing your project seems easy since you spent many hours writing it and have a very deep understanding of it, but this does not translate to a good presentation.
- A good presentation is well thought out and practiced.
- Side Notes:
 - A well thought out presentation allows you to hide flaws or unfortunate "features" that you may not want anyone to know about.
 - I am also much less inclined to ask pointed questions if you have a well thought out and thorough presentation.

Miscellaneous

Some of these points are duplicated in other places in this document. It's ok.

- All students need a [Github](#) account
- All programs need to be turned in and running to pass the course
- General Assignment Rules:
- Due dates and times are as listed on assignment and can change with prior notice to class, and always in your favor (aka more time).
- Formatting of programs is important, and will be graded accordingly.
- Your name is required on ALL documents uploaded or turned in. A handwritten name is not acceptable.
- For any assignment, you will create a folder and include all documents created by you within this folder for submission. This includes programs, input / output files, readme's, documentation, etc. This folder will subsequently be uploaded to Github in the repo you created for the course.
- Attending class is one of the primary keys to doing well in this class. Students may be dropped for excessive absences. There is no distinction made between excused and unexcused.
- Make-up exams are not given. If I see fit, then I will replace a missed exam with your final exam test grade (but this is optional to instructor based on circumstances, attendance, participation, etc.).
- Late work will be accepted on a case by case basis. Late penalty is 15 points (out of 100) for initial lateness and 1 half a letter grade (5 points) for every class period until the total reduced is 50 (half credit). Extremely late work is totally at the instructor's discretion on whether it will be accepted or not.
- Programs containing syntax errors are unacceptable and will be returned without grading (your programs must work).
- Periodically homework assignments will be taken up and graded. It is the student's responsibility to keep up with assignments and to ask questions over the assigned work, even if absent. All homework assignments are due at the specified time that may or may not be in conjunction with a class day. All assignments / homeworks will be uploaded via Github.

Official Course and Department Policies

Attendance Policy

Although student attendance is not calculated into the grade, attendance will be taken each day to track students. If a student is absent more than 2 classes without an excuse and is not performing well in class, a report can be submitted to the Dean of Students and the student may be dropped from the class. Classes will not be streamed for absent students, whether it is excused or not.

Behavior in the classroom

Students are to assist in maintaining a classroom environment that is conducive to learning. This means that the presence of electronic devices other than your calculator are not to be seen, heard, or implied, ever. Questions are encouraged and discussion is acceptable, provided it is pertinent and does not distract from the lesson.

Make Up Work/Exams/Quizzes:

- For planned excused absences: Exam may be taken early by prior arrangement.

- For unplanned documented absences: I may replace your exam grade with your final exam grade. I reserve the right to make that decision.
- For unplanned undocumented absences: Zero on the exam

Late Work

Late programs will be accepted with an initial **15 point** reduction and then a ****5 point **deduction** class day. ***No late programs for last programming assignment. No late homework will be allowed in the Connect System.***

Computer Requirements: Taking this class requires you to have access to a computer (with Internet access) to access online course material. ***Personal computer technical difficulties will not be considered a reason for extra time to submit assignments, tests, or online discussion postings.*** Computers are available on campus in various areas of the buildings, as well as the in the library. Contact your instructor immediately upon having computer trouble. If you have technical difficulties in the course, there is also a student helpdesk available to you. The university cannot work directly on student computers due to both liability and resource limitations, however they are able to help you get connected to our online services. For help, log into *****D2L.*****

Policy on Testing Process

The Department of Computer Science has adopted the following policy related to testing.

- All bags, purses, electronics (turned off), books, etc. will be placed in the front of the room during exams, or in an area designated by the instructor.
- Unless otherwise announced by the instructor, nothing is allowed on the desk but pen/pencil/eraser and test papers.
- A student who leaves the room during an exam must turn in the test and will not be allowed to return.

Policy on Programs

- Tests *will* have questions covering out-of-class assignments. Know the material!
- Students will be invited to orally answer questions regarding their assignments in my office and failure to answer those questions correctly will result in deductions from their grades. (Every student can expect to be invited 1-2 times during the semester to answer questions.)

Computer Science Tutoring

Tutoring is available in ***Bolin Room 119 & the Office of Tutoring and Academic Support Programs (TASP)*** in Moffett Library. A tutor may assist with programs and homework for computer science classes. The tutor will not do your work.

Academic Misconduct Policy & Procedures

Cheating, collusion, and plagiarism (the act of using source material of other persons, either published or unpublished, without following the accepted techniques of crediting, or the submission for credit of work not the individual's to whom credit is given). The Department of Computer Science has adopted the following policy related to cheating (academic misconduct). The policy will be applied to all instances of

cheating on assignments and exams as determined by the instructor of the course. (See below for link to MSU definitions.)**

- 1st instance of cheating in a course: The student will be assigned a non-replaceable grade of zero for the assignment, project or exam. *If the final grade in the course, does not result in a one letter grade reduction, the student will receive a one letter grade reduction in course.*
- 2nd instance of cheating in a course: The student will receive a grade of F in course & immediately be removed from course.
- All instances of cheating will be reported to the Department Chair and, in the case of graduate students, to the Department Graduate Coordinator.

Note: Letting a student look at your work is collusion and is academic misconduct!

See Also: [MSU Student Handbook](#): Appendix E: Academic Misconduct Policy & Procedures
https://msutexas.edu/student-life/_assets/files/handbook.pdf.

Recording of Class Lectures

Permission must be requested in writing and obtained from the instructor before recording of class lectures. If permission is granted, the recording may only be used by the student making the recording. Recordings (or any class materials) may NOT be posted on any internet source without written permission of the instructor. Failure to adhere to the policy may result in removal from the course with a grade of F or other appropriate punishment.

University Policies and Procedures

Student with Disabilities

Any student who, because of a disability, may require special arrangements in order to meet the course requirements should contact the instructor as soon as possible. Students should present appropriate verification from Disability Support Office during the instructor's office hours. Please note instructors are not allowed to provide classroom accommodations to a student until appropriate verification has been provided. For additional information, contact the Disability Support Office in Clark Student Center 168 - Phone: (940) 397-4140

Policy on Concealed Handguns on Campus

Senate Bill 11 passed by the 84th Texas Legislature allows licensed handgun holders to carry *concealed* handguns on campus, effective August 1, 2016. Please note, open carry of handguns, whether licensed or not, and the carrying of all other firearms, whether open or concealed, are prohibited on campus. Areas excluded from concealed carry are appropriately marked, in accordance with state law. For more information regarding campus carry, please refer to the University's webpage at [MSU Campus Carry Policy](#) *<https://msutexas.edu/campus-carry/rules-policies###> *. If you have questions or concerns, please contact MSU Chief of Police Steven Callarman at Steven.callarman@msutexas.edu.

Midterm Progress Report

In order to help students keep track of their progress toward course objectives, the instructor for this class will provide a Midterm Progress Report for all students in the course through each student's MSU Portal account. Midterm grades will not be reported on the students' transcript; nor will they be

calculated in the cumulative GPA. They simply give students an idea of where they stand at the midpoint of the semester. Students earning below a C at the midway point should a) schedule a meeting with the professor and b) Seek out tutoring.

Important Dates

Visit [MSU's Registrars](#) website [Important Dates](#)

https://msutexas.edu/registrar/_assets/files/pdfs/fall23front.pdf