

In this ASU programming class, 'failing' is encouraged

Students build coding confidence through effort, persistence

By Kelly deVos, ASU News
October 14, 2025

Artificial intelligence is rewriting the rules of nearly every profession, and software development is no exception. In an era when large language models can spit out code in seconds, students studying computer science face a new challenge: How do you truly learn programming when AI can give you a shortcut?

In the [School of Computing and Augmented Intelligence](#), part of the [Ira A. Fulton Schools of Engineering](#) at Arizona State University, faculty members like [Erik Trickel](#) are rethinking what it means to prepare students for careers in a rapidly shifting technological landscape.

From zoning out to leaning in

Trickel, a Fulton Schools assistant teaching professor, has been leading the charge in reshaping one of the school's foundational courses, [CSE 240 Introduction to Programming Languages](#). He's quick to joke about his old lectures where students might have zoned out in the back row, missing key concepts until exam day.

"Computer science isn't something you can half-get," Trickel says. "You need to be able to sit down and not only write the code, but also know why it works. My goal was to build a class that supports that kind of deep, non-negotiable learning in a way that is supportive and fun."

With the help of instructional innovation coaches [Stefani Jenkins](#) and [Jonathan Baek](#) in the Fulton Schools [Learning and Teaching Hub](#), Trickel transformed his class from a traditional lecture format to a hybrid, flipped model built around collaboration and active problem-solving.

Instead of listening passively, students now spend class time working in small groups to tackle programming challenges. Each group has a trained staff member or teaching assistant who keeps discussions lively and ensures every student participates.

Creating coders who don't quit

One key aspect of the new learning model is its mastery-based grading system. To pass, students must complete live programming challenges in a proctored environment. The twist: They can try as many times as they need until they succeed.

“There’s no way to fake it,” Trickel says. “You can’t just memorize answers or copy code from online. You have to demonstrate the skill.”

Weekly testing sessions, held multiple times per week, allow students to come back, retry and refine their approach. Each time they attempt a challenge, the system generates a problem that is a variation of the same core concept, so students who keep at it eventually succeed, often with a deeper understanding than they expected. That structure has created a cultural shift in the classroom.

“Struggle isn’t failure anymore,” Trickel says. “It’s just part of the process.”

Where group work works

For many students, the collaborative, mastery-driven environment has been a game changer.

“Working in our groups is really helpful,” says Chelsea Allyson Angeles, a computer science student specializing in cybersecurity. “We all have different backgrounds and different ways of approaching problems. When you are thinking alone, you have no access to other perspectives. But in groups, we swap ideas and help each other.”

She added that while generative AI can be useful, it doesn’t replace human interaction.

“AI has a lot of knowledge,” Angeles says. “But it’s not always offered in a way that’s as accessible as learning from another person.”

For Mahin Patel, a sophomore in computer science, the course did more than sharpen coding skills. It built lasting friendships.

“The class really makes learning fun,” Patel says. “I’ve met people I know I’ll stay in contact with even after it ends.”

Darya Riazati, a junior studying computer science, sees clear connections between the course structure and real-world software development.

“Working in these kinds of groups better prepares us for what’s out there,” Riazati says. “It emulates styles like agile or scrum, and gives us experience using team members as resources.”

From learners to leaders

The course also encourages past students to return as mentors. Teaching assistants go through training not only in subject matter but also in leadership skills that help them foster group collaboration.

Sriharsha Silasagaram, a sophomore in computer science, took the class last year and jumped at the chance to return as a TA. “The class was one of my favorites,” he says. “I wanted to come back and offer the same support I received to other students.”

That’s exactly the kind of cycle Trickel hoped to create.

“The best way to solidify your own knowledge is to teach it,” he says. “When students step into that role, they not only help others, they become much stronger programmers themselves.”

Jenkins explains that training students to assume these mentorship roles was a critical part of efforts to overhaul the class.

“We really wanted these student leaders to feel well-supported to run these groups,” Jenkins says. “That meant giving them training not just on the material but also on how to ask good questions and draw out contributions from everyone at the table.”

Preparing for an AI-powered future

At its core, the revamped CSE 240 course isn’t just about learning C, C++, Scheme or Prolog. It’s about preparing students to think critically, collaborate effectively and build confidence in their abilities to solve problems. Those skills matter even more in a world where AI tools are becoming ubiquitous.

Trickel is clear about the balance: Generative AI is welcome as a learning support, but students must build their own coding and debugging abilities.

“If you let AI do the thinking for you, you’re not actually learning,” he says. “We want our graduates to be the ones guiding technology, not the other way around.”

That philosophy reflects a broader vision across the Fulton Schools and ASU. Computer science education must evolve as quickly as the technology it teaches. By blending collaboration, persistence and human mentorship with modern tools, CSE 240 offers a glimpse of what that future could look like.

“Students leave this class knowing they can do it,” Trickel says. “And that confidence will carry them through their careers in an AI-driven world.”

This story originally appeared on [ASU News](#).

Main image



Computer science students in the CSE 240 Introduction to Programming Languages class engage in a discussion about a class assignment. Students work in peer-led groups as part of a new hybrid class format for computer programming. Developed by Erik Trickel, an assistant teaching professor of computer science and engineering in the School of Computing and Augmented Intelligence, part of the Ira A. Fulton Schools of Engineering at Arizona State University, with support from the Fulton Schools Learning and Teaching Hub, the learning model strives to ensure students are ready to face emerging challenges in the tech industry. Photo by Erika Gronek/ASU

Text image(s)



The hustle and bustle of a typical morning where students spend their programming languages class actively engaged in collaborative problem-solving. Photo by Erika Gronek/ASU



Assistant Teaching Professor Erik Trickel discusses an assignment with computer science students. Photo by Erika Gronek/ASU



A student leads her group in a brainstorming exercise to determine the best approach for solving a programming problem. Photo by Erika Gronek/ASU