"My project team is exploding, and I'm not sure what to do!" As an Engineering Teaching Consultant, computer instructors often contacted me with such requests. In this case, we met and found a solution to stabilize the team. But this wasn't the end of our discussion. We later reconvened to cover strategies for avoiding such crises before they manifest, such as setting up team contracts, organizing group check-ins, and establishing clear communication guidelines.

This proactive stance has been shaped by my experience as a teaching assistant for six classes and a mentor for seven students: I believe the key to effectively supporting students, regardless of background, is *maximizing productive instructional time*. Student contact hours are scarce and I have found that the most important part of teaching and mentoring is often *outside of the classroom, before the course or advising session even starts*. By combining preventative planning with time dedicated to cultivating a friendly environment where students willingly communicate desires or challenges, I can swiftly diagnose and tackle setbacks when they inevitably arise. While this investment is initially expensive, I use the time it saves later to implement educational scaffolding and active learning techniques. In addition to helping enforce course concepts, this guides students in developing generalizable critical thinking skills. I believe these are vital for well-rounded and independent engineers and scientists. In the rest of this statement, I summarize my experiences and detail how I implement my guiding principles, giving examples from both the traditional classroom and also from mentoring and advising undergraduate researchers.

Teaching and Mentorship—Experience Overview: *Classroom Teaching:* I have been *a teaching assistant ten times, for six different courses,* at the University of Michigan. With enrollments from 60 to over 1000, these courses *spanned the undergraduate curriculum* including the complete three-semester introductory programming series, Discrete Math, and two upper-level electives (Software Engineering and Programming Languages). As a teaching assistant, I designed and taught weekly discussion sections to 40 students, wrote and graded exam questions, and fielded student queries both in person and online, receiving positive evaluation scores.

Course design: I helped overhaul and improve Michigan's Programming Languages (EECS 490) syllabus, and I designed and taught a nine-week supplemental tutorial for introductory students which led to improved programming outcomes [5] (also described in my research statement).

Individualized Mentorship: I've acted as the *primary advisor for seven undergraduate students*, leading to *four peer-reviewed publications with mentees*. In addition, as an Engineering Teaching Consultant, I helped other instructors navigate crises and improve their teaching.

Classroom Teaching: *I believe in maximizing productive instructional time with preventative planning, educational scaffolding, and an active learning environment.*

My primary goal in the classroom is to *optimize individual instructional time* so that I can efficiently convey challenging concepts to students with varying prior knowledge. To achieve this, I use dynamic educational scaffolding, manage office hours with an emphasis on transferable skills, and employ active learning techniques to facilitate retention.

One way I efficiently teach core content to diverse groups is by using *educational scaffolding*. I find this technique exciting because it can provide tailored support that is gradually and dynamically removed as students gain independence. For example, I make optional, platform-specific tutorials for core concepts, directing struggling students to them during office hours or via online forums. I have found this particularly effective in introductory courses when teaching fundamental skills, such as file system navigation or debugging. I also scaffold generalizable cognitive skills that are relevant for computing, but may be weaker in students with lower incoming preparation. For example, my semester-long technical reading program led to significantly higher programming gains compared to an active control group [5]. These efforts demonstrate how deliberate scaffolding can help students succeed in computing, regardless of background.

My focus on generalizability extends to office hours, where I commonly encountered queues of over 100 students. When tensions run high, I have found that *balancing long-term advice with relieving short-term stress* helps me promote educational fairness and foster genuine understanding. One strategy I use is pinpointing an underlying skill that helps students solve not only their current error, but also future similar errors on their own. I employ a three-step process: 1) I ask students what they have tried or what they think they should try, 2) I determine familiarity with an approach aligned with course goals, and 3) I guide them in implementing the approach, highlighting misconceptions. I seen how tempting it can be to give into pressure and directly provide answers. However, in my experience, students return for help with the next small issue. With my skills-focused approach, I have observed that students are better equipped to handle subsequent challenges independently. Thus, while initially more time-intensive, I favor such preventive measure that help manage large student volumes in the long term.

My emphasis on transferable skills helps free resources to integrate active learning, which is a set of techniques to engage students in the learning process via direct participation, peer to peer interactions, and meaningful reflection on content. I include various techniques that I learned from teaching-related workshops and from the educational research literature. Even simple techniques, such as predictive polls via a straightforward thumbs up or down gesture, can have a big impact. For example, in a software engineering class I might ask students to predict which line is the most suspicious using fault localization. The benefits are two-fold: 1) I can quickly gauge student understanding, dynamically adjusting my lecture if needed, and 2) research has found that making a prediction, even if incorrect, improves learning. I explicitly include active learning in my course design. For example, in the nine one-hour sessions in my technical reading curriculum, I included techniques such as think-pair-share, interactive physical manipulations (e.g., flashcards), and student-led icebreakers. My use of active learning contributes to student satisfaction. For example, when TAing a 400-level Software Engineering Course, I received at least a four out of five on all 22 instructor-specific scores.

Research Advising: *I believe in maximizing the likelihood that a student becomes a successful independent researcher with upfront planning, open communication, and research ownership.*

My favorite part of teaching is research advising. I treat advising like an apprenticeship. Rather than assume students know the basics coming in, I work to build necessary skills from the ground up. *I believe that the key to becoming an effective and independent researcher is not inherent talent, but rather time and experience*. I have observed three main failure modes that prevent students from sticking with research: 1) Students can experience a panic spiral, triggered by research uncertainty, 2) personal crises can lead to communication breakdowns and misaligned expectations, and 3) students can lose motivation, concluding that research is not for them. To combat these barriers, I use planning to diminish uncertainty, foster a communicative and supportive environment, and involve students in creative research aspects to increase ownership.

I have found that making expectations clear and providing individually-tailored research plans is essential for reducing the uncertainty of new researchers. I begin this at my first meeting with a new student, setting aside an hour and a half to introduce the project and set expectations, both mine for them and theirs for me. I further offset uncertainty by planning tentative submission venues up to a year in advance. I routinely revisit timelines at project meetings, and I deliberately include slack in my schedules. This not only builds confidence, but also helps me detect challenges and hurdles far enough in advance that I can adjust the plan if needed. For example, one student-led study hit an unexpected snag when we realized many participants were actually scammers. Because we had contingency plans in place and we caught this early in our regular meetings, we had time to modify the pre-screening form and recruit real participants. This sort of nerve-racking unexpected event could have caused student panic, but my emphasis on proactive planning helped diminish the negative impact. This student-led paper was later accepted at *identifying citation removed*.

I have found that personal issues, rather than research hurdles, can sometimes be the biggest obstacles. Such issues can be hard to detect and ameliorate because students, especially those with imposter syndrome, may not feel their personal issues are relevant. To build an environment where students feel comfortable asking for help, I make the habit of mentioning my own non-research life events at the beginning of each meeting. This can be as simple as trying out a new recipe or a nice bike ride. I also allude to personal struggles when appropriate to normalize failure. These conversations take time. However, I believe the trade-off is worth it: I have seen how promoting communication can lead to students choosing to open up about events impacting their research performance. For example, when one student suddenly became significantly less productive, she felt comfortable enough to share that her father was in the hospital. With this knowledge, I adjusted my expectations, working with her to continue the research while prioritizing her happiness. With these accommodations, she retained her motivation. After her father recovered, *identifying citation removed.* Similarly, a student, who is a native speaker of Chinese, confided in me that she struggled with English idioms and vocabulary. With this insight, I was better equipped to provide targeted support. I encouraged her to interrupt me if she heard a word she didn't understand, and I spent more time explaining the rationale behind my feedback on her written work. I saw significant improvement in her language ability in both informal and formal contexts: a paper *identifying citation removed.*

I am interested in my students retaining motivation, even in the face of unexpected barriers. I have found that one way to do this is by including the student in the creative process: I begin with a core research topic and then encourage students to steer one or more research questions in a direction that resonates with them. This process not only cultivates creative thinking but also instills a sense of ownership. For one example, I proposed a study on cannabis use in software engineering. Through my encouragement of her creativity, my student advocated for us to also explore stimulants and anti-depressants due to personal interest in the intersection of mental health and software. This student later confided to me that at the time I hired her, she was considering dropping out of school entirely, in part due to struggles with managing her mental health. However, Through structured and supportive mentorship, combined with a sense of ownership, she fell in love with research and learning. She not only graduated, but was also an author on (*identifying citation removed*), and decided to pursue graduate school. She has since started her PhD in Software Engineering.

Conclusion: Overall, I believe my ten semesters of traditional classroom teaching experience along with my demonstrated track record of advising seven students on research (resulting in four papers with my advisees) has positioned me to succeed as a tenure-track faculty member.