> *through it, then I feel badly about myself and I feel like I'm not as smart as everyone else."*

Self-efficacy theory states that the vicarious experience of watching peers informs students' enactive attainments [6], which could explain why social comparisons arose as a rationale for students' responses. Students also rationalized their responses by citing recommendations given by their professors. For example P4 said:

> *"I put disagree, mainly because of our professors. They always tell us that planning should be first and once you have your plan then you start coding."*

These quotes suggest that peer comparisons and recommendations from professors are additional factors that may contribute to the particular moments that prompt students to negatively self-assess, which should be explored further in future studies. These other factors may help explain why we did not find a correlation between students' responses to the perceptions of professional programmer questions and the self-assessment vignette questions for every moment. Students' inaccurate perceptions of professional programming practice partially explain why students make negative self-assessments at natural parts of the programming process, but other factors also play an important role in determining students' self-assessments.

5 CONCLUSION

In this paper, we contribute the results of a survey study with 214 CS1 students from three universities. We found that some students reported that each of the programming moments prompt them to negatively self-assess, even though these moments occur in professional practice. Interestingly, there was no significant difference in students' responses between the three universities for twelve of the thirteen self-assessment vignette questions despite large differences in the populations of students that these universities serve. This suggests that many self-assessment moments generalize across different university populations. We also found that the frequency with which students negatively self-assess correlates with their overall self-efficacy in their programming course. While there was little difference in the self-assessment moments across the schools, the degree to which self-assessment moments correlated with students' overall self-efficacy significantly differed between the universities. Finally, we found that students' perceptions of professional programmers correlated with their responses to some of the self-assessment vignette questions, suggesting that these perceptions may influence when students negatively self-assess.

Our findings demonstrate that students are negatively self-assessing often and, importantly, in response to moments that occur in professional programming practice. This suggests that students may not have a good understanding of professional practice and the experiences they should expect to encounter while programming. This gap in knowledge may exist because CS1 courses typically do not teach about the cognitive aspects of programming, including problem-solving strategies and programming practices [36]. By explicitly teaching about programming practices in CS1, for example with direct instruction [36] or by showing recordings of professional programmers [10], we may be able to help students build accurate expectations of the programming process and reduce negative self-assessments. This type of intervention might also improve students' self-efficacy in their programming course, since we found that students who negatively self-assess more frequently and strongly tend to have lower self-efficacy. However, while helping students develop more accurate representations of professional practice may be one promising intervention strategy for changing student expectations, other messaging around peer comparisons and instructor expectations may also be needed to reduce negative self-assessments. Since studies show that students factor their perceived ability into their decision to major in CS [34, 39, 51], these types of interventions may help to lower the dropout rates in CS programs.

While these results provide valuable insight into CS1 student experiences, our study has a few important limitations. First, even though we chose the self-assessment moments included in our survey based on previous research and preliminary user studies, it is likely that our set of moments is not comprehensive. There may be other moments in the programming process that prompt students to make negative self-assessments. In particular, cultural differences both within and outside of the US may strongly influence the moments that prompt students to negatively self-assess. Additionally, while our interviews with a small sample of students provide promising initial evidence that our survey accurately captures student self-assessments, we need to conduct a more formal validation of the survey. Finally, our results rely on student self-reports based on remembered experiences triggered by the vignettes. While retrospective assessments are still relevant for understanding students' perceptions of ability, we do not know whether these responses accurately reflect the thoughts that arise during programming episodes. However, we chose this methodology because the survey allowed us to collect a larger sample of data with consistent experiences between students.

We believe there are many opportunities to extend and apply these findings through future work. First, studying the self-assessment moments across a wider variety of contexts and countries would help generalize these findings and allow for interesting cross-cultural comparisons. We are also interested in exploring factors beyond student perceptions of professional programmers, such as class format and social comparisons, to understand why students negatively self-assess in these moments. Finally, since our study was designed to measure correlations, future work could identify the factors that cause students to self-assess and confirm that negative self-assessments have a causal effect on students' self-efficacy. Our findings show that many students negatively self-assess at moments that are natural parts of the programming process, and that these self-assessments negatively correlate with self-efficacy. This research lays a theoretical foundation for designing interventions that reduce unnecessary negative self-assessments for novice programmers.

6 ACKNOWLEDGEMENTS

We thank Rachel Trana for collaborating on the survey design and our community partners for providing access to students. We thank our Delta Lab colleagues for valuable discussions and feedback. This work was supported by the National Science Foundation under Grant IIS-1755628 and by the National Science Foundation Graduate Research Fellowship Program.

REFERENCES

[1] Tuukka Ahoniemi, Essi Lahtinen, and Teemu Erkkola. 2007. Fighting the student dropout rate with an incremental programming assignment. In *Proceedings of the Seventh Baltic Sea Conference on Computing Education Research-Volume 88*. 163–166.

[2] Cheryl S. Alexander and Henry Jay Becker. 1978. The Use of Vignettes in Survey Research. *Public Opinion Quarterly* 42, 1 (1978), 93. https://doi.org/10.1086/268432

[3] Susan A. Ambrose (Ed.). 2010. *How learning works: seven research-based principles for smart teaching* (1st ed ed.). Jossey-Bass, San Francisco, CA. OCLC: ocn468969206.

[4] Eugene Anderson and Dongbin Kim. 2006. Increasing the success of minority students in science and technology. (2006).

[5] Petek Askar and David Davenport. 2009. An investigation of factors related to self-efficacy for Java programming among engineering students. *TOJET: The Turkish Online Journal of Educational Technology* 8, 1 (2009).

[6] Albert Bandura. 1982. Self-efficacy mechanism in human agency. *American psychologist* 37, 2 (1982), 122.

[7] Albert Bandura. 1997. *Self-efficacy: The exercise of control.* Macmillan.

[8] Lecia Barker, Christopher Lynnly Hovey, and Leisa D. Thompson. 2014. Results of a large-scale, multi-institutional study of undergraduate retention in computing. In *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*. IEEE, Madrid, Spain, 1–8. https://doi.org/10.1109/FIE.2014.7044267

[9] Theresa Beaubouef and John Mason. 2005. Why the high attrition rate for computer science students: some thoughts and observations. *ACM SIGCSE Bulletin* 37, 2 (June 2005), 103. https://doi.org/10.1145/1083431.1083474

[10] Jens Bennedsen and Michael E. Caspersen. 2005. Revealing the programming process. In *ACM SIGCSE Bulletin*, Vol. 37. ACM, 186–190.

[11] Jens Bennedsen and Michael E. Caspersen. 2007. Failure rates in introductory programming. *ACM SIGcSE Bulletin* 39, 2 (2007), 32–36.

[12] Jens Bennedsen and Michael E. Caspersen. 2019. Failure rates in introductory programming - 12 Years Later. *ACM Inroads* 10, 2 (2019), 6.

[13] Nancy E Betz and Gail Hackett. 1983. The relationship of mathematics self-efficacy expectations to the selection of science-based college majors. *Journal of Vocational Behavior* 23, 3 (Dec. 1983), 329–345. https://doi.org/10.1016/0001-8791(83)90046-5

[14] Adrian W Bowman and Adelchi Azzalini. 1997. *Applied smoothing techniques for data analysis: the kernel approach with S-Plus illustrations*. Vol. 18. OUP Oxford.

[15] Deborah L. Butler and Philip H. Winne. 1995. Feedback and self-regulated learning: A theoretical synthesis. *Review of educational research* 65, 3 (1995), 245–281. http://journals.sagepub.com/doi/abs/10.3102/00346543065003245

[16] Spencer E. Carlson, Daniel G. Rees Lewis, Elizabeth M. Gerber, and Matthew W. Easterday. 2018. Challenges of peer instruction in an undergraduate student-led learning community: bi-directional diffusion as a crucial instructional process. *Instructional Science* 46, 3 (June 2018), 405–433. https://doi.org/10.1007/s11251-017-9442-0

[17] Ryan Chmiel and Michael C. Loui. 2004. Debugging: from novice to expert. *ACM SIGCSE Bulletin* 36, 1 (2004), 17–21. http://dl.acm.org/citation.cfm?id=971310

[18] Jonathan Cohen, D, Lauren Margulieux, Maggie Renken, and W. Monty Jones. 2020. Conclusions from the Validation of a Vignette-Based Instrument to Measure Maker MIndsets. In *Proceedings of the Fifteenth International Conference for the Learning Sciences (ICLS) 2020*. International Society of the Learning Sciences, Inc.[ISLS]., Nashville, TN, USA.

[19] J. M. Cohoon. 2006. Just get over it or just get on with it. *Retaining women in undergraduate computing. In J. Cohoon & W. Aspray (Eds.), Women and information technology: Research on underrepresentation* (2006), 205–238.

[20] Jessica L. Collett and Ellen Childs. 2011. Minding the gap: Meaning, affect, and the potential shortcomings of vignettes. *Social Science Research* 40, 2 (March 2011), 513–522. https://doi.org/10.1016/j.ssresearch.2010.08.008

[21] Alireza Ebrahimi. 1994. Novice programmer errors: Language constructs and plan composition. *International Journal of Human-Computer Studies* 41, 4 (1994), 457–480.

[22] K Anders Ericsson and Herbert A Simon. 1984. *Protocol analysis: Verbal reports as Data*. MIT Press, Cambridge, MA.

[23] Allan Fisher and Jane Margolis. 2002. Unlocking the clubhouse: the Carnegie Mellon experience. *ACM SIGCSE Bulletin* 34, 2 (2002), 79–83. http://dl.acm.org/citation.cfm?id=543836

[24] Jamie Gorson and Eleanor O'Rourke. 2019. How Do Students Talk About Intelligence? An Investigation of Motivation, Self-efficacy, and Mindsets in Computer Science. In *Proceedings of the 2019 ACM Conference on International Computing Education Research - ICER '19*. ACM Press, Toronto ON, Canada, 21–29. https://doi.org/10.1145/3291279.3339413

[25] Desmond Wesley Govender and Sujit Kumar Basak. 2015. An investigation of factors related to self-efficacy for Java programming among computer science education students. *Journal of Governance and Regulation* 4, 4 (2015), 612–619. https://doi.org/10.22495/jgr_v4_i4_c5_p6

[26] Gail Hackett and Nancy E Betz. 1981. A self-efficacy approach to the career development of women. *Journal of Vocational Behavior* 18, 3 (June 1981), 326–339. https://doi.org/10.1016/0001-8791(81)90019-1

[27] Brian Hanks and Matt Brandt. 2009. Successful and unsuccessful problem solving approaches of novice programmers. In *ACM SIGCSE Bulletin*, Vol. 41. ACM, 24–28.

[28] Bassam Hasan. 2003. The influence of specific computer experiences on computer self-efficacy beliefs. *Computers in Human Behavior* 19, 4 (July 2003), 443–450. https://doi.org/10.1016/S0747-5632(02)00079-1

[29] Amanpreet Kapoor and Christina Gardner-McCune. 2018. Considerations for switching: exploring factors behind CS students' desire to leave a CS major. In *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education - ITiCSE 2018*. ACM Press, Larnaca, Cyprus, 290–295. https://doi.org/10.1145/3197091.3197113

[30] Päivi Kinnunen and Beth Simon. 2012. My program is ok – am I? Computing freshmen's experiences of doing programming assignments. *Computer Science Education* 22, 1 (March 2012), 1–28. https://doi.org/10.1080/08993408.2012.655091

[31] Thomas D. LaToza, Gina Venolia, and Robert DeLine. 2006. Maintaining mental models: a study of developer work habits. In *Proceeding of the 28th international conference on Software engineering - ICSE '06*. ACM Press, Shanghai, China, 492. https://doi.org/10.1145/1134285.1134355

[32] Robert W Lent, Steven D Brown, and Kevin C Larkin. 1984. Relation of self-efficacy expectations to academic achievement and persistence. *Journal of counseling psychology* 31, 3 (1984), 356. Publisher: American Psychological Association.

[33] Robert W Lent and Gail Hackett. 1987. Career self-efficacy: Empirical status and future directions. *Journal of Vocational Behavior* 30, 3 (June 1987), 347–382. https://doi.org/10.1016/0001-8791(87)90010-8

[34] Colleen M. Lewis, Ken Yasuhara, and Ruth E. Anderson. 2011. Deciding to major in computer science: a grounded theory of students' self-assessment of ability. In *Proceedings of the seventh international workshop on Computing education research*. ACM, 3–10.

[35] Alex Lishinski, Aman Yadav, Jon Good, and Richard Enbody. 2016. Learning to Program: Gender Differences and Interactive Effects of Students' Motivation, Goals, and Self-Efficacy on Performance. In *In Proceedings of the 2016 ACM Conference on International Computing Education Research*. ACM Press, 211–220. https://doi.org/10.1145/2960310.2960329

[36] Dastyni Loksa, Amy J. Ko, Will Jernigan, Alannah Oleson, Christopher J. Mendez, and Margaret M. Burnett. 2016. Programming, Problem Solving, and Self-Awareness: Effects of Explicit Guidance. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM Press, 1449–1461. https://doi.org/10.1145/2858036.2858252

[37] Robert McCartney, Anna Eckerdal, Jan Erik Mostrom, Kate Sanders, and Carol Zander. 2007. Successful students' strategies for getting unstuck. In *ACM SIGCSE Bulletin*, Vol. 39. ACM, 156–160.

[38] Carol Midgley, Avi Kaplan, Michael Middleton, Martin L. Maehr, Tim Urdan, Lynley Hicks Anderman, Eric Anderman, and Robert Roeser. 2013. *Patterns of Adaptive Learning Scales*. Technical Report. American Psychological Association. https://doi.org/10.1037/t19870-000 type: dataset.

[39] Irene T. Miura. 1987. The relationship of computer self-efficacy expectations to computer interest and course enrollment in college. *Sex Roles* 16, 5-6 (March 1987), 303–311. https://doi.org/10.1007/BF00289956

[40] Laurie Murphy, Gary Lewandowski, Renée McCauley, Beth Simon, Lynda Thomas, and Carol Zander. 2008. Debugging: the good, the bad, and the quirky–a qualitative analysis of novices' strategies. In *ACM SIGCSE Bulletin*, Vol. 40. ACM, 163–167. http://dl.acm.org/citation.cfm?id=1352191

[41] Frank Pajares and M David Miller. 1994. Role of self-efficacy and self-concept beliefs in mathematical problem solving: A path analysis. *Journal of educational psychology* 86, 2 (1994), 193. Publisher: American Psychological Association.

[42] Ilias O. Pappas, Michail N. Giannakos, and Letizia Jaccheri. 2016. Investigating Factors Influencing Students' Intention to Dropout Computer Science Studies. In *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education*. ACM Press, 198–203. https://doi.org/10.1145/2899415.2899455

[43] Brian Parkinson and A. S. R. Manstead. 1993. Making sense of emotion in stories and social life. *Cognition & Emotion* 7, 3-4 (May 1993), 295–323. https://doi.org/10.1080/02699939308409191

[44] Michael Perscheid, Benjamin Siegmund, Marcel Taeumel, and Robert Hirschfeld. 2017. Studying the advancement in debugging practice of professional software developers. *Software Quality Journal* 25, 1 (March 2017), 83–110. https://doi.org/10.1007/s11219-015-9294-2

[45] Leo Porter and Beth Simon. 2013. Retaining nearly one-third more majors with a trio of instructional best practices in CS1. In *Proceeding of the 44th ACM technical symposium on Computer science education - SIGCSE '13*. ACM Press, Denver, Colorado, USA, 165. https://doi.org/10.1145/2445196.2445248

[46] Chris Quintana, Meilan Zhang, and Joseph Krajcik. 2005. A Framework for Supporting Metacognitive Aspects of Online Inquiry Through Software-Based Scaffolding. *Educational Psychologist* 40, 4 (Dec. 2005), 235–244. https://doi.org/10.1207/s15326985ep4004_5

[47] Vennila Ramalingam, Deborah LaBelle, and Susan Wiedenbeck. 2004. Self-efficacy and mental models in learning to program. In *ACM SIGCSE Bulletin*, Vol. 36. ACM, 171–175.

[48] Dale H. Schunk. 1989. Attributions and Perceptions of Efficacy during Self-Regulated Learning by Remedial Readers. (1989).

[49] Dale H. Schunk. 1995. Self-efficacy, motivation, and performance. *Journal of Applied Sport Psychology* 7, 2 (Sept. 1995), 112–137. https://doi.org/10.1080/10413209508406961

[50] Sabine Sonnentag. 1998. Expertise in professional software design: A process study. *Journal of applied psychology* 83, 5 (1998), 703. Publisher: American Psychological Association.

[51] F. Boray Tek, Kristin S. Benli, and Ezgi Deveci. 2018. Implicit Theories and Self-Efficacy in an Introductory Programming Course. *IEEE Transactions on Education* 61, 3 (Aug. 2018), 218–225. https://doi.org/10.1109/TE.2017.2789183

[52] Nanette Veilleux, Rebecca Bates, Cheryl Allendoerfer, Diane Jones, Joyous Crawford, and Tamara Floyd Smith. 2013. The relationship between belonging and ability in computer science. In *Proceeding of the 44th ACM technical symposium on Computer science education*. ACM, 65–70.

[53] Willemien Visser. 1987. Strategies in programming programmable controllers: A field study on a professional programmer. In *Empirical Studies of Programmers: Second workshop (ESP2)*. Ablex, 217–230.

[54] Christopher Watson and Frederick W.B. Li. 2014. Failure rates in introductory programming revisited. In *Proceedings of the 2014 conference on Innovation & technology in computer science education - ITiCSE '14*. ACM Press, Uppsala, Sweden, 39–44. https://doi.org/10.1145/2591708.2591749

[55] Aharon Yadin. 2011. Reducing the dropout rate in an introductory programming course. *ACM Inroads* 2, 4 (Dec. 2011), 71–76. https://doi.org/10.1145/2038876.2038894