OPINION

EngageCSEdu

by Briana B. Morrison, *University of Nebraska Omaha*, Michelle Craig, *University of Toronto*, and Beth A. Quinn, *University of Colorado at Boulder*

Programming Assignments: Identifying Red Flags

In our previous column [2] we discussed the qualities and criteria for assessing programming assignments. As the editors-in-chief of EngageCSEdu, this topic is very much of interest to us: we ask our reviewers to evaluate and critique submissions of computing instructional materials based on a rubric. In this column, we will discuss the value of "red flags"—aspects of an instructional approach that give pause, or, as the Oxford English Dictionary defines it, are "sign[s] of danger, a warning; a signal to stop." What aspects of your instructional materials should give you pause? Here we discuss the types of "red flags" and how to avoid them in your own assignments. Removing "red flags" can help you improve the quality of the assignments and engage more students.

A set of "red flags" has been a part of the review rubric for EngageCSEdu since its inception in 2014. All reviewers—computer science educators and social science reviewers—are asked to assess whether a submission has any "red flags" associated with diversity and inclusion. These include problematic stereotypes [4] or denigration of particular groups, material likely to trigger a stereotype threat for some groups, polarizing content, and outdated references. Examples include cultural references or slang that may seem "cool" and fun for the instructor (e.g., science fiction references) but may confuse or exclude students who aren't part of that particular subcultural group [1]. In addition to unnecessarily increasing cognitive load [5] for some students, these in-group references may also unintentionally reinforce ideas about who "belongs" in computing [3].

A red flag does not necessarily indicate a problem with the assignment that would make it unusable. It isn't wrong to use graphics in an assignment or to tell students that they can assume the preconditions are met, but each of these things requires some explicit thought on when, where, and how these might play out in a particular classroom.

Another subtle example is an assignment that only uses typically white, male names or only white male musicians, artists, or scientists in a data set. On the other end are assignments where the author is trying to engage women but ends up reinforcing gender stereotypes.

Red flags aren't just about inclusion. We have noticed in this first year of our editorship of EngageCSEdu that there are also "red flags" associated with the computer science portions of the assignments. These are things that set off alarm bells for an experienced instructor—issues that might foreshadow a potential disaster. They could be descriptions or starter code that might confuse students or issues that would cause a lot of headaches or extra work for the instructor. Here is a short list that comes to mind as CS red flags.

1. Does an algorithm used in the assignment have unspecified edge cases?
2. Does the handout make it clear what to do if preconditions are not met? This is a broad class of red flags including things like what to do if a user doesn't provide input as expected, or what to do if a file is missing, unreadable or doesn't have the expected format.
3. Does the code depend on specific libraries or a specific operating system?
4. Does the assignment use graphics or interactive user input such that it can't be automatically tested?

A red flag does not necessarily indicate a problem with the assignment that would make it unusable. It isn't wrong to use graphics in an assignment or to tell students that they can assume the preconditions are met, but each of these things requires some explicit thought on when, where, and how these might play out in a particular classroom. What is acceptable for a small class of 20 students that meets daily does not necessarily work well for a

PHOTO: ©WWW.ISTOCKPHOTO.COM/OSKANOV

class of 400 students or for a course using auto-grading with systems that have strict test cases. Having to answer a few student questions on how to handle an empty or ill-formatted input file is no big deal. Having to answer several hundred student questions—whether by yourself or with the help of your TAs—is lost productivity time. Having to adjust the assignment requirements right before the due date because of an unforeseen test case can cause great frustration for students.

Part of the power of EngageCSEdu is the *generalizability* or *reusability* of the assignments we publish. Just as you may read a research paper and think that you can borrow a research method, survey protocol, or measurement instrument, we want you to see an EngageCSEdu assignment and realize that it could be used within one of your classes. Yet we know that no assignment is a one-size-fits-all and each assignment would need some adaptation to be used. We want the published submissions to be easily adaptable. This means identifying issues that may be problematic depending on the size, style, or delivery method of your class. The key is disclosing this information up front so it can be included in the consideration of whether or not it is worth the effort to adapt the assignment for your own use.

We now ask our computer science reviewers to identify CS red flags in submissions. Although we are partially counting on their experienced intuition when they spot a possible issue, we would also like to provide them with a more comprehensive list than the examples above. Do you have specific signs of danger that you are careful to watch for when you are creating your own programming assignments? Perhaps they have come from a bad experience and a personal vow to "never do **that** again." We would love to hear your stories about the red flags that you wish you had noticed in the past. ❖

References

1. Cheryan, S., Plaut, V. C., Davies, P. G., and Steele, C. M. Ambient belonging: How stereotypical cues impact gender participation in computer science. *Journal of Personality and Social Psychology*, 97, 6 (2009), 1045–1060; https://doi.org/10.1037/a0016239.
2. Craig, M., and Morrison, B. B. How should we assess assignments? *ACM Inroads*, 11, 2 (2020), 17–19; https://doi.org/10.1145/3381024
3. Master, Allison, Cheryan, Sapna, and Meltzoff, Andrew N. Computing whether she belongs: Stereotypes undermine girls' interest and sense of belonging in computer science. *Journal of Educational Psychology*, 108, 3 (2016), 424–437.
4. Schmader, T., Johns, M., and Forbes, C. An integrated process model of stereotype threat effects on performance. *Psychological Review*, 115, 2 (2008), 336.
5. Sweller, J. (2011). Cognitive load theory. *Psychology of learning and motivation*, 55 (2011), 37–76.

Briana B. Morrison
Computer Science Department
University of Nebraska Omaha
Omaha, NE 68182
bbmorrison@unomaha.edu

Michelle Craig
Computer Science Department
University of Toronto
Toronto, Canada
mcraig@cs.toronto.edu

Beth A. Quinn
National Center for Women & Information Technology (NCWIT)
University of Colorado
Campus Box 417 UCB
Boulder, CO 80309
beth.quinn@ncwit.org

DOI: 10.1145/3397774