Exploring the Impact of Assessment Policies on Marginalized Students' Experiences in Post-Secondary Programming Courses

Eman Sherif
University of Washington
Seattle, Washington, USA
emans@uw.edu

Jayne Everson*
University of Washington
Seattle, Washington, USA
everjay@uw.edu

F. Megumi Kivuva*
University of Washington
Seattle, Washington, USA
megumik@uw.edu

Mara Kirdani-Ryan*
University of Washington
Seattle, Washington, USA
marakr@uw.edu

Amy J. Ko
University of Washington
Seattle, Washington, USA
ajko@uw.edu

ABSTRACT

Objectives: Assessments play a crucial role in computer science courses by providing insights into student learning. While previous research has explored various aspects of assessments, little attention has been given to assessment policies that instructors devise and their impact on students' experiences. Our goal was to investigate: *How do assessment policies shape marginalized students' experiences in coding classes?*
Method: We conducted 19 semi-structured interviews with post-secondary students currently enrolled in or completed a class where their code was evaluated. To recruit, we primarily targeted students from underrepresented racial groups in computer science. Many of these students attended large 4-year public universities. During the interviews, we inquired about students' experience with different assessment policies and how those policies affected their lives and experiences completing the assignments.
Results: Our findings revealed ten distinct ways policy and students' lives interacted to create or heighten inequities, which significantly shaped marginalized students' lives. Many policies did not consider the unique experiences of their students and students' needs. Additionally, due to unclear and strict policies, students experienced frustration, confusion, and demotivation, consequently diminishing their sense of belonging in computer science and weakening their self-efficacy as programmers. This reveals the negative consequences of poor assessment policy choices and provides insight into how assessment policies can create barriers to learning computer science for marginalized students.

CCS CONCEPTS

• **Social and professional topics** → **Student assessment**; **Computing Education**.

*Authors contributed equally to this research

ICER '24 Vol. 1, August 13–15, 2024, Melbourne, VIC, Australia

ACM ISBN 979-8-4007-0475-8/24/08.
https://doi.org/10.1145/3632620.3671100

KEYWORDS

Assessment, Assessment Policies, Marginalized populations

ACM Reference Format:
Eman Sherif, Jayne Everson, F. Megumi Kivuva, Mara Kirdani-Ryan, and Amy J. Ko. 2024. Exploring the Impact of Assessment Policies on Marginalized Students' Experiences in Post-Secondary Programming Courses. In *ACM Conference on International Computing Education Research V.1 (ICER '24 Vol. 1), August 13–15, 2024, Melbourne, VIC, Australia.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3632620.3671100

1 INTRODUCTION

Assessments are pivotal tools for computer science instructors, offering vital insights into students' understanding [15]. Assessments provide metrics to help instructors understand what their students know and do not know and allow students to understand what areas of improvement are needed [18, 25]. In many courses, assessments are a large part of course grades, which impact completing pre-requisites and prospective job opportunities [31]. Beyond evaluation, they motivate students and encourage progress in students' learning [5]. Specifically, programming assessments can go beyond technical evaluation, fostering creative thinking and honing problem-solving skills [6, 9, 20]. Overall, assessments are dynamic tools and practices that can shape positive learning environments and cultivate a mindset of continual improvement among students.

Assessment *policies* are the backbone of assessments, shaping the assessments' content, deliverables, and grading schema. Assessment policies are used to describe the purpose of the assessment, the type of assessment(s), what time assessments are distributed and due, the grading rubric, how feedback is given, and how scores are recorded. Assessment policies does not include other parts of course design like how students are able to contact instructors, in class activities, lecture design, etc. There are many different approaches to designing policies. To create assessments, some prior work has drawn from Bloom's taxonomy [30, 45], crafting assessment questions has proven transformative, enhancing student performance in theory-based computer science courses by using a range of cognitive skills to allow students to express their knowledge differently [30]. Assessment policies also dictate what kind of deliverables a student produces from an assessment. For example, in a co-constructed culturally centered computational embroidery course students only had two requirements: 1) produce work that represented their interests and 2) use Turtlestitch, a programming language designed for embroidery. This led to a plethora of creative projects that wove students' identity and interests into programming [19]. Moreover, assessment policies determine how programming assignments will be graded and how feedback is given to students. Autograders allow students to receive feedback fast and help facilitate large-scale instruction [13, 16, 42]. This instant feedback not only elevates correctness levels but also bolsters students' confidence in their programming abilities, especially women [28].

There are many choices that instructors must make when they create assessment policies and in doing so, they may heighten inequities in the classroom. For example, Xie. et.al found bias in assessment questions, by using differential item functioning (DIF) to understand which assessment questions can exhibit bias against women and underrepresented racial groups. They found five areas of bias against women and thirteen areas for underrepresented racial groups. Domain experts had many different interpretations of this data, but these data points gave them guidance on what areas need to be analyzed more closely to remove bias from those types of questions. These findings demonstrate how assessment question choices can be biased against students from marginalized backgrounds more [49]. Additionally, Medel et al. found different ways computer science programming assessments have bias against women. The language, representation, and imagery of women in computer science assessments can lead to lower confidence, negative self-reflection, and continue to normalize bias against women. To combat gender bias, Medel suggested using animals instead of names or replacing images of women with well-known monuments [27]. It is essential to recognize that these biases compound the inequities faced by marginalized students, creating learning environments that may hinder success in computing classes.

Prior work in other fields offers some guidance on how to design more equitable assessment policies. In writing pedagogy, Inoue's *Antiracist Writing Assessment Ecologies: Teaching and Assessing Writing for a Socially Just Future* [17] discusses how writing assessments are a complex system that exceeds the sum of its interconnected parts. Inoue guides instructors to understand the different ways their assessment policies can unintentionally be racist if policies are not designed to be anti-racist. For example, Inoue observed students who were not White encountered language barriers in writing courses. Then, Inoue incorporated labor-based grading contracts to make grades more accessible for all students. In the contract system, grades are determined by the amount of labor that is agreed upon by instructors and students. Within the contract system, students expressed they were participating in a fairer, more predictable, and more democratic system, which provides them the freedom to explore and take risks.

Although Inoue's work gives guidance on assessment policy design, there are still many gaps in our understanding of *how* to design equitable policies for programming assessments in particular. Understanding how policies produce inequities by interacting with student identities and lives may help guide instructors on what possible impacts their policies have and which policies they should redesign. Additionally, we need to understand how these policies interact with students' identities to create new inequities or heighten existing ones. Moreover, Inoue's guidelines are not exactly applicable to computer science because assessment content and type differ between disciplines.

To address these gaps, we ask *How do assessment policies shape marginalized students' experiences in programming classes*? To answer this question, we conducted 19 semi-structured interviews with undergraduate students who have completed or were currently enrolled in a programming class. We then thematically analyzed the interviews and found ten different ways assessment policy and identity interacted to create or heighten inequities. For each interaction, we highlight the policy and provide examples of how it has shaped students' ability to complete assessments. We found that many policies did not consider the lives of students and created frustration, confusion, and additional stress. Additionally, we found some policies weakened students' sense of belonging by isolating students and weakened their self-efficacy as programmers because they did not have adequate support for their needs.

2 RELATED WORK

Our study is grounded in Inoue's anti-racist assessment framework. This framework discusses how racism is embedded in writing assessment due to writing upholding the dominant white discourse[1] as the ideal text. Inoue's framework first describes an *assessment ecology* as a combination of *people*, *power*, *environments*, *actions*, and *political activities* that impact how students complete a writing assessment. For example, in writing, an assessment ecology could be the interactions between students, instructors, classrooms, campus, language, and racial politics. Next, Inoue discusses the different elements that make up an anti-racist assessment ecology and how to design anti-racist assessment ecologies, proposing seven elements of anti-racist assessment ecology: *power*, *parts*, *purpose*, *people*, *process*, *products*, and *places*. These elements shape policy design and can be used to determine if a policy is equitable.

Inoue describes in detail what each element of an anti-racist assessment ecology entails.

(1) *Power* is used to describe the different power relations between the instructor and students and how instructors will not uphold the dominant White discourse.
(2) *Parts* refers to the artifacts that regulate and embody the assessment. The parts are the most visible entities such as the instruments, scores, grades, essay prompts, etc.
(3) *Purpose* refers to why we are using certain instruments to assess and what learning we are assessing, as well as students' involvement in articulating the assessment purpose.
(4) *People* refers to students' and instructors' ability to cultivate the environment and culture in the course affecting the assessment.
(5) *Process* refers to how instructors anticipate building a rubric, how feedback is given to the student, how students will reflect on their work, and monitoring the amount of labor students put into an assignment.
(6) *Products* refer to the consequences of the assessment such as grades or feedback.

[1]Inoue explains the dominant white discourse as writing courses accepting only the variant of English that is primarily associated with White people rather than other variants that are associated with marginalized communities. Therefore, marginalized students who speak in variations of English must make choices about language and force them to perform their racial identity differently.

(7) *Places* refers to the locations where students produce their assessments such as classrooms, dorm rooms, labs, or online discussion platforms.

Inoue describes how these elements can help instructors understand problems with current assessment ecology to create an anti-racist assessment ecology that is more critical, sustainable, and fair. Inoue explains by adopting this framework he was able to understand his students' learning and experience holistically and he was able to understand what they leave the course with.

Inoue offers a set of questions as a guide for instructors to create anti-racist writing assessment ecologies. Among the strategies proposed, one technique is incorporating labor-based grading techniques into assessments. This was utilized in computer science assessments by incorporating more formative and reflective assessments, which led to reducing student anxiety [24]. Furthermore, Inoue suggested providing mechanisms for students to provide feedback to their peers and reflect on the feedback given to them. The use of peer review on assessments was found to be engaging for students because it allowed students to reflect on their work with guidance from other peers to make their work better and helped them understand areas of weakness in their knowledge [7].

To our knowledge, no prior work in computing education has built upon all of Inoue's anti-racist assessment ecology framework. The closest work is one study that aimed to understand how to use labor-based grading in computer science courses [24]. In this course, everything in the class was a "labor" and all labor equally weighed in the student's grade. They conducted surveys with students and found that labor-based grading made students feel less anxious and had less motivation to cheat. This study suggests that Inoue's work is applicable to the computer science field. However, there are gaps in understanding the most effective ways of applying these guidelines and which ones are lacking in computer science education.

Prior work in computing education has examined many assessment formats and policies, but generally not from an equity lens nor an anti-racist lens. The most common assessment practice in programming classes is automated programming assignments that are evaluated for functionality. This assessment is usually evaluated by using a large dataset on the program and ensuring the output matches the expected results [1]. To design assessments, project-based learning assessments are popular because they increase engagement and motivation for students through developing solutions for real-world problems [10]. Additionally, multiple-choice written exams are common to examine code comprehension through code tracing [21]. To provide feedback, semi-automatic assessments where functionality is assessed automatically, but instructors can give feedback through comments [1]. Moreover, there are multiple different techniques added to current assessment methods [2, 11] to make assessments more engaging and improve performance. Furthermore, prior work has analyzed cheating policies and how to maintain academic integrity within computer science courses [39, 40]. To minimize cheating there are several different practices instructors can take such as revising cheating policies to be more clear or introducing plagiarism detection tools [41]. Although prior research has found different ways to improve assessment practices, none of these consider the impacts of these assessment practices on marginalized students. Moreover, programming courses' unique assessment format makes it difficult to directly apply Inoue's theory because of the way programming assessments are designed.

Some prior work on equitable assessment policies has been about evaluating current curriculum to ensure equitable and effective instruction. One work analyzed the use of the Teacher Accessibility, Equity, and Content (TEC) Rubric by computer science instructors to understand areas of inequity so instructors can create new policies to support students. TEC is used by educators, decision-makers, and designers to craft suitable computing curricula tailored to their students' needs. This rubric helped direct teachers' attention towards inequities within their course such as providing support for ELL students or providing extensions for students [3]. This work provides a tool to understand inequities in their classes and design policies, but this rubric is mainly used to evaluate curricula and is not focused on students' experience with policy. Additionally, the TEC rubric is used to evaluate introductory computing courses in K-12 and is not used in post-secondary institutions.

Some other prior work has focused on designing new grading policies for computer science classes to create more equitable learning environments. Specifically, resubmission policies and non-traditional grading schemes have become more popular because traditional grading increases inequities [8, 38]. Prior work on students' experiences with resubmission policies found that this policy took the pressure off students and focused on learning [23, 33, 35]. There are many different non-traditional grading schemes, one being ungrading. ngrading focuses on de-emphasizing numeric grades [12, 26, 36]. One strategy of ungrading is to label work as complete or incomplete or for larger assignments complete, nearly complete, somewhat complete, or incomplete. For this course, instructors used contract grading to explain to students how their grades would be translated to traditional letter grades (A, B, or C). Spurlock et. al found that this grading scheme increased students' intrinsic motivation and self-efficacy and created a more equitable environment by leveling the playing field by putting less emphasis on numeric grades [44]. Resubmission and ungrading are two assessment policies that may help mitigate inequities marginalized students face, but prior work has not focused on marginalized students' experiences with these policies and if these students experience the benefits from these policies. Additionally, prior work has focused on designing new policies, but investing time into understanding the implications of current assessment policy can ensure the same issues are not translated into new policies and that good current assessment choices are not discarded.

While there is some work on equitable assessment policies, it is still unknown how marginalized students' experience in programming classes in particular is shaped by assessment policies.

3 METHODS

Leveraging Inoue's framework, we investigated the interaction between assessment policies and student lives by conducting semi-structured interviews with undergraduate students in 4-year and 2-year institutions who recently took a programming course. We utilized Inoue's theory to frame questions about assessment policies and how they impacted students' experiences. We received Institutional Review Board approval before conducting this study.

3.1 Positionality and Reflexivity

The first author is a woman of color with experience as a TA for many introductory programming courses. She has seen the negative impacts assessment policies can have on marginalized students and found it necessary to explore how these policies shape students' experiences. She believes computer science courses should consider and incorporate students' unique experiences and identities. As a marginalized person in computer science, she has experienced many inequitable and racist policies but aims to separate her experience from the data by not making assumptions based on her experience.

The second author has a background and formal training in secondary education. She taught middle and high school math, science, and making for more than ten years. This informs her understanding of the assessment and interpretation of the data. Her justice-centered focus, and studies of implicit power structures are a threat to her interpretation of the data. She works to separate her experience from the data provided by students.

The third author has a background in computer engineering and computer science, and has worked as a TA for a variety of courses across the system stack. While they largely benefited from computing culture as an undergraduate student, their doctoral work has surfaced the ways that computing culture exacerbates marginalization. They bring their experiences in computing, both legitimacy and delegitimacy, to balance interpretations of the data provided by students.

The fourth author has experience learning and teaching computing in and out of traditional school contexts. In their education, they often felt as though the assessments they were given were not an accurate representation of their learning or knowledge. During data analysis, they found that some experiences aligned with their experiences learning computer science, while others did not. They found conversations with the analysis team to be helpful to disentangle their own biases during data analysis. Much of their work aims to broaden participation in computing for historically underrepresented minorities in computing.

The **fifth author** is a biracial woman of color and had been post-secondary faculty for more than 15 years at the time of this study. She has always had a deep skepticism about summative assessments and the way they are often designed to erode student agency, in and outside of computing. She has frequently resisted dominant assessment policy norms around cheating, lateness, and other punitive framings of assessment. When she learned of more equity-focused approaches to assessment more than a decade ago, she sought to examine their impacts more directly, which partially led to her supervising this research. She managed this perspective by helping the first author shape an interview protocol that centered student voices, rather than the research team's skepticism of dominant programming assessment norms.

3.2 Pilots

Due to the limited prior work on assessment policies' impacts on students, the author decided to conduct pilot interviews with 6 undergraduate and graduate students majoring in computer science who are from different marginalizaed identities. These pilot interviews were used to refine the method and gain more insight into which interview questions could effectively answer our research question. Our goal with the pilots was not to answer the research question, but to shape the research design and methods; we focused the interviews on participants' worst experiences, to understand how we might develop rapport with participants.

3.3 Recruiting and Participants

We recruited students who attended 2 or 4-year colleges in the Seattle area. To recruit students we contacted the advisors at institutions for computer science, informatics, electrical engineering, and similar majors, who sent out mass emails to their students. Additionally, we recruited through affinity organizations such as the LEAP alliance, National Society of Black Engineers (NSBE), Society of Hispanic Professional Engineers (SHPE), and Women in Computing (WIC). We asked interested students to fill out an interest form to collect their demographic data and contact information. Moreover, we asked students if they had a good or bad assessment experience they would like to talk about. Additionally, we asked students if they are currently enrolled in or completed a programming class, defined as any class that contains assessments that evaluate the correctness of a student's code. The classes were not limited to computer science or engineering classes but could be informatics or lab science classes. We chose to adopt this broad stance on what counts as a programming class because we did not find any literature that discussed how assessment policies for classes that evaluate students' code differ based on discipline, and we did not want to artificially limit the types of classes we investigated and risk gathering incomplete data. In our study, we were interested in interviewing marginalized students, so we heavily recruited from underrepresented racial minorities (URM). In total, we recruited a diverse group of 19 students who experienced varying forms of marginalization.

We focused on recruiting students from Black, Native American, Latine, Pacific Islander, and Native Hawaiian backgrounds because they are severely underrepresented with less than 13% total computer science bachelors graduates in 2022 (CRA). There are a plethora of different reasons why there is such a small percentage of computer science graduates such as lack of resources, little exposure to computing, and lack of sense of belonging [29, 37, 50]. Additionally, underrepresented racial groups often experience additional inequities such as income inequality that can cause students to work other jobs and not be able to own a computer [48, 49]. These inequities disturb a student's learning experience which can cause them to become disengaged with their classes and fall behind.

Table 1 shows demographic data for all participants.

3.4 Interviews

Through our pilot interviews, we found that many participants internalized their misfortunes and blamed themselves for having a negative assessment experience. Some participants did not feel that they have anyone to reach out to and when they do they are often told it is "too late" or "nothing that can be done". Lastly, the multiple inequalities they faced seemed to work in a way that made their experiences extremely difficult and they believed their instructors were unaware or did not care about their experiences. During these interviews, we found that due to the large amount of different

Participant ID	Race/ Ethnicity	Gender
1	Black	Woman
2	Latino/ Middle Eastern	Man
3	Black	Woman
4	Black	Man
5	Black	Woman
6	Asian/Pacific Islander	Woman
7	Asian/Pacific Islander	Woman
8	Asian/ Pacific Islander	Man
9	White	Man
10	Asian/ Pacific Islander	Woman
11	Asian/ Pacific Islander	Woman
12	Asian/ Pacific Islander	Woman
13	Middle Eastern	Woman
14	Asian/ Pacific Islander	Woman
15	Asian/ Pacific Islander	Woman
16	White	Man
17	Black	Woman
18	Asian/ Pacific Islander	Woman
19	Asian/ Pacific Islander	Woman

Table 1: Self-reported demographic data of Race/ Ethnicity and gender of all participants

views on assessment policies, a semi-structured interview approach would help answer the research question.

Thus, the first author designed and conducted semi-structured interviews over Zoom during the summer of 2023. The interviews were recorded and transcribed with participant consent. Through the preliminary findings and questions posed by Inoue, they devised guiding interview questions. First, the interviewer would ask the participants about their overall experience with programming classes. Then the interviewer would ask the participant if they wanted to discuss a good or bad experience first. After this, the interviewer asked questions based on participants' experiences and how that impacted participants' lives in and outside the course. Questions in our protocol included:

- What is a good experience you had with an assessment policy?
- What is a bad experience you had with an assessment policy?
- Did the instructor have certain ways to demonstrate what you are being assessed on?
- What tools did you use for your assessment?
- Were you given the opportunity to go back and work out any of the things you got wrong?
- Was there any benefit to reflecting on your work? Was there any time?
- How did they demonstrate that grade to you? Canvas, gradescope, etc. What did you see/ how did you feel?
- How did instructors take feedback?
- How were your needs addressed by the instructor?

3.5 Analysis Plan

First, the first author read through transcripts and corrected transcription errors to match the video recordings of the participants. Additionally, they anonymized any identifiable information such as names and institutions by replacing them with pseudonyms. Next, the analysis team read through transcripts to familiarize themselves with the data. The second and fourth authors read through 10 transcripts. The first and third authors read through the other 9 transcripts. Then, the analysis team met and discussed initial thoughts on the data.

To analyze the data, we inductively coded our data to identify significant statements and interpret claims about the data. Our analysis process follows these steps:

Round One of Analysis

- In pairs, the authors read through the transcripts and gathered significant statements that described the impacts of assessment policies. Then created a code that described the significant statement.
- The first author gathered all significant statements in a main document and removed any duplicate statements.
- The analysis team met and performed affinity diagramming where we grouped significant statements based on the policy and its impact on the student
- The analysis team met to discuss policies and their impacts on students.

During affinity diagramming, there was disagreement on what impact late submission policies had on students. Very few students mentioned this policy and the analysis team could not think of a coherent impact, so they decided to not consider the impact of this policy.

Round Two of Analysis

- The first author wrote short paragraphs, summarizing participants' statements about identities, policies, and impacts. Even though all members of the analysis team had access to full transcripts, these paragraphs served as a reference with important information about specific experiences with policies participants mentioned.
- Each author read the short paragraphs and inductively created statements descriptions of how the inequities in each student's lives interacted with assessment policies. To do this, they linked policies students mentioned and how they affected students' experience. The impacts found in the first round of analysis informed the analysis team on what these interactions could be.

The interactions from round two formed the basis of our results. During this round we did not have any disagreements because the team finalized these statements together.

We conducted our analysis using guidelines outlined by Hammer and Berland [14], who advocate for qualitative thematic analysis as a means to generate interpretative claims about data. We did not collect agreement measurements such as inter-rater reliability instead we resolved disagreements through discussion.

4 RESULTS

Here we describe ten interactions between policy, identity, and students' lives, each emerging from our analysis. Each of these created or heightened inequities that students faced, harming their learning, self-efficacy, grades, and attitudes toward computer science.

4.1 Unclear cheating policies created a fear of collaboration which resulted in students feeling isolated and teaching assistants feeling overburdened

In many courses, participants discussed how cheating policies were not very clear because they did not know what they were or were not allowed to discuss with peers. This created a large fear of collaboration, and many participants felt isolated when completing their assignments. Students wanted to collaborate to be able to engage in discussions and learn from each other. This consequence also placed an undue burden on teaching assistants who were left to manage the fallout of this isolation, struggling with increased workload and student inquiries resulting from the lack of clarity surrounding collaboration guidelines.

Participant 1 discussed how cheating policies made them fear collaboration due to the lack of clarity. She did not feel comfortable understanding whether or not something would be considered cheating:

> *I feel like the collaboration part of it, the policy is always unclear. It's like collaborate, but don't do this. But so then I remember, especially in the first couple of months, those first introductory courses, is always like should we? I don't know if this counts as cheating, it was like walking around tip toes and stuff like that. So those that that's always usually the unclear bit. And then you just pray , okay, hopefully, nothing like, I don't get any messages*

Participant 3 discussed feeling isolated in classes and not communicating with other people because of cheating policies. She did not have many friends who would take computer science courses, so she could not seek help from their peers:

> *I had like I knew classmates, but I wasn't talking to anyone for the assignments. It was more like everyone was scared. Oh, my God is what I'm doing, going to be considered cheating. So unless you had a very close knitted groups that were like helping each other in past quarters no one was helping each other with that homework or talking about it. From what I knew.*

4.2 When policies disallowed TAs from helping students in the way they need, students found help elsewhere, weakening the relationship the students had with the course instructors

When policies prevented TAs from providing the assistance students required, whether due to rigid guidelines or limitations on the type of help they could offer, it created a gap in support. Consequently, students were forced to seek help from other sources, such as alternative forms or external resources. This reliance on other assistance weakened the connection between students and their course instructors. By relying on other forms of support, students were less engaged in assessments and had a diminished sense of belonging in the course.

Participant 13 discussed how she felt uncomfortable asking for more help from TAs even when she needed it. She resorted to asking friends for help instead and felt guilty about seeking their assistance:

> *So I definitely felt like I had to go to multiple office hours. And then as much as the TAs were helpful. And they tried to help you understand stuff. I have this mentality where it is like I don't want to keep the TA. If I feel like I generally don't understand something I'll probably just say, "Oh, I got it" after like the third time they try to explain something to me, and then I'll probably just ask - I had to ask a lot of my friends who were in the CS program, or like friends who took CS. At like other tech schools, to like help me with my assignments, which I probably shouldn't have done looking back*

Participant 3 explained how they needed help with syntax, but they were not able get help with syntax due to strict policies that limited what TAs could help them with:

> *My problem is the syntax and not the concept understanding wise. We are not allowed that kind of help.*

Participant 5 explained how instead of approaching TAs in their computer science courses she would resort to other forms of support:

> *They are kind of stripped on what they can tell you. And with the [support center] it's less. There aren't that many rules. If you go there for help, and let's say you're stuck with a method or something. If you go to someone. And you're like, okay, I'm stuck on this. They will sit there until you sort of like figure it out so they will stay with you. And, another thing I forgot to mention is, there are a lot more students at the TAs office hours, and that's only 1 hour of office hours. The [support center] it's like you have the whole day, so you can come any time, and you're more likely to yeah, get help there.*

4.3 Help seeking through office hours was not always possible which made students feel less psychologically safe in courses

Students had very limited ways to get help on their programming assignments. They expressed that TA office hours were the most helpful resource because TAs were very knowledgeable about the programming assignments. Sometimes students were not able to attend office hours due to scheduling conflicts, limited availability, or discomfort seeking help, making them feel less psychologically safe. Psychological safety refers to one's ability to feel safe to express themselves, speak up, or disagree openly [4]. When students felt unable to access support through office hours, they felt isolated, uncertain, and anxious about their assessment performance. A lack of psychological safety could weaken students' confidence in their programming abilities and seeking help.

Participant 19 explained how she was not been able to take advantage of TA office hours because she did not found them helpful.

The TAs were not well trained and gave conflicting answers, which led to a lot of confusion about what steps she should take when completing the assessment. She spent more time trying to understand what the question on the assessment was asking rather than working on the assessment:

> *It was frustrating because none of the TAs were really well trained, I think, to support students in that class. So I would go to office hours like multiple times every week and try to ask questions about coding assignments, because both my friend and I were having a lot of issues sometimes even understanding what the questions were about. We would get conflicting answers from TA or TAs, or straight up tell us, "I don't know". And yeah, it was just sort of like overall, a pretty frustrating experience. And it was like a lot more time dedicated to trying to understand what the problems were as opposed to actually like build coding skills. And you know that sort of thing. So yeah, that was overall like, not the best experience. And I did end up deciding to S/NS [Satisfactory/ Not Satisfactory] that class because I was spending so much time on it every week. I was also working part time. I don't have you know the energy to deal with this.*

Conversely, Participant 6 described a positive relationship with her TA which resulted to her feeling comfortable asking for help because of the TA's positional identity:

> *My TA, I remember her. I would bug her all the time with emails. But I also feel I'm really grateful because I felt comfortable asking her for help. I think you know she is a woman in this field. I think there's also a lot of intersecting positionalities when it comes to being a woman in this field. Being a woman of color in this field, she was a woman of color, TAing this class. So I'm sure that probably added to how I felt comfortable speaking to her.*

4.4 When life and school responsibilities conflicted, policy favored school responsibilities, creating inequities

Many students had different life challenges that they were facing alongside pursuing a college degree. Sometimes students needed to decide between important personal commitments and navigating policies to complete an assessment. When faced with this choice, students chose to follow policy.

Participant 18 discussed how she was sick with COVID-19 and still felt that she was required to attend class sections. She did not want to miss opportunities for extra credit because previously the professor did not explain expectations for assignments well, causing her to perform poorly on the majority of assignments:

> *I even remember my family was like "What are you doing? You're sick with Covid!". Yeah I gotta go to quiz section in case he gives extra credit*

Participant 1 discussed how they were unsure what kind of accommodations they were able to get due to a religious holiday. They wanted to fully engage in the course assessments and their holiday, so they asked for accommodations on exams. However, sometimes they did not feel comfortable asking for accommodations because the policy was not clear:

> *Because I feel like the university has policies in place for Ramadan. They're like, "Oh, like you can ask for accommodations," but there's no guidance or policy already about what you can and can't ask where we should ask for. If that makes sense. It's hard to guide them like should I just ask for, like unlimited late days, for 30 days? Should I ask for like extended due day? It's so hard to navigate what to ask for, because there's not usually like how to do it, you know. So I also don't want to be another thing like it, like I mentioned before. I don't want to ask for too much, so I have a hard time asking like. Last year, I just took a test during Ramadan once I was fasting I didn't want to ask for anything. I didn't. I didn't know what the other option would be like, I feel like it would have been easier for me. If it's like, Hey, we have alternate test times after sunset or something. It would have been easy to ask. But since I didn't have that, I just took the test.*

4.5 Accommodations for extenuating circumstances or disability oftentimes did not meet students' needs, leading them to internalize failure

Many universities had existing accommodations for disabilities. However, students expressed how these accommodations were not sufficient for them to complete an assignment. Similarly, many students who had extenuating circumstances, such as falling ill or facing personal problems, had the barrier of needing to ask for accommodations informally. As a result, students viewed their inability to complete their assignments as a failure, which was detrimental to their self-esteem, motivation, and performance.

Participant 18 had a learning disability for which she received university accommodations, but she did not feel that the accommodations were adequate for the exam because the format was new, and she was isolated from the rest of the class, unable to get a lot of clarification on the exam:

> *I felt rushed, even though I had extended time for my disability. If I had received the normal length of time. I don't even know what I would have done to be honest. Because not only do you have to like think about the question and kind of try to remember things or try to think about the best way to answer the questions, both like the programming one and the written like response ones, but also, like actually, like physically, writing everything down is a slower process for them, even typing it. So I felt very rushed, and I think that it took me a couple of seconds to kind of remember certain topics and just like, but what was really rushing was like, just physically like writing everything down*

Participant 13 discussed how she was sick halfway through the course and was in the hospital for a week. She asked her professor for an extension on an assignment, but he said no, so she needed to navigate recovering from her sickness and finishing an assignment:

> *I felt almost helpless, like I wanted to finish the assignment. I know that I'm all on my own because, I had like a health emergency. I remember I had a health emergency couple of weeks into that quarter. And I had to be in the hospital for a whole week. And I emailed, all my professors being like, "hey I'm sorry I can't come to class. I'm gonna try to find someone to take notes for me. But is there any way like I can get an extension on this week's assignment". We have like coding assignments every week. And I remember him sending me a 2 sentence response, being like "I'm so sorry like, but you can't get an extension. Just watch the lecture notes." Like it was so brief compared to all the other professors. They had been like, " I'm so sorry. Please take time off like, really, school is not that important compared to what you're going through?" And I just felt like I almost started crying because I was like, how are you so inhumane like I explained in such vivid detail and like, how are you so like you. Don't you have any sympathy?*

4.6 If assessment purpose was not communicated well, students lose motivation

If the rationale behind assessments is unclear or poorly communicated, students struggle with understanding the significance of the task. Without a clear explanation of what they were being assessed on, why, and how it related to the learning objectives, students might have perceived assessments as disconnected from their learning. This impacted students' motivation to complete the assessment and engagement throughout the course.

Participant 4 took a computer science course that was being revamped, which had many assessment policy issues. He explained how the course final was a 20-page exam, but most of it was comprised of story lines to explain the problem. He did not understand why the professor added these story lines to the exam:

> *I believe it was like a 20 page exam. And like an hour and like there were so many unnecessary. He would like make storylines like a whole anime about a character. He would keep the whole story for each section of the assignment. And I was like, you're getting way too creative with this it'd be like Johnny is making a web app. His webapp has this, this, this Johnny has a friend. Who's this, this this. For the first question, test his code, and then, like you would make a test. And then, like that whole, it will take 2 pages to just explain the question. And then, like another 2 pages, just to like, write down the answer.*

4.7 Unclear grading schemes caused students to spend more time on assignments leading them to sacrifice personal time

When students lacked clarity on how their work would be assessed, they felt compelled to invest additional time into their assignments to try to meet these unclear expectations. This pressure caused students to spend less time on personal activities or other classes. Unclear policies made it difficult for students to balance academic and personal time because they were unable to manage time effectively.

Participant 18 described how she did not understand what was missing in her function comments, which resulted in her getting marked down. She created very long function comments, which had taken a lot of extra time, and she did not believe that it made her program better:

> *I would always get marked down on my function comments ... It's not like I ever forgot to write them I always wrote them. But they would always mark me down from like the exceeding standards to like meeting standard, because it wasn't long enough. So I got to a point where I was writing like almost a paragraph. You're not gonna catch me anymore. And looking back on those assignments, I was like, this makes my code not readable at all.... and I thought that was really ridiculous, like during that class, I was like, I finally cracked the code. I'm such a genius like, I'm like playing the system. This is great. But then, when I look back on my code, I was like, this is awful code.*

4.8 Resubmission policies encouraged students to read feedback but also caused fatigue because they demanded students to do extra work that interfered with other coursework and personal obligations

Usually, students did not pay attention to feedback because they did not find reviewing it to be helpful. However, with resubmission policies, students would review their work because they knew there would be some benefit from it. Overall, we found that students enjoyed the resubmission policies and found that they were fair and took pressure off of students. Some students had explained some annoyances with resubmission policies, such as getting more points off and how completing a resubmission had taken a lot of extra time. Students were not been able to balance personal obligations with resubmissions, which made it difficult for them to take advantage of this policy.

Participant 13 discussed how her favorite policy was the resubmission policy because it encouraged her to look at her work and revise it. She explained that it made her feel less stressed and reassured because she was always able to get points back. However, she explained how during the final and midterm, she was very stressed because she had not been able to resubmit:

> *Yeah, my favorite thing was definitely resubmissions, because I could actively see the mistake that I made, or the error. Like what I got my points were just reduced*

> *on, and then fix it. Seeing how that translated definitely helped me feel more reassured and less panic. For the final and for the midterm, literally my hand was like shaking so bad because I was like I was telling myself, like my grade literally depends on these tests because of how much I did that on the assignments.*

Participant 2 discussed how he was less stressed because he was able to have regrades on quizzes:

> *I also liked how they had quiz re-grades in that one. So like if you did like poorly on a quiz you can just sign up for like a retake. I feel like in that way it made like the assessments, a lot less like stress inducing.*

Participant 2 also discussed how resubmission policies could be frustrating because they may resubmit an assignment, but will get more points taken off:

> *Another thing that really annoyed people is that sometimes they would let you do regrade. So in that way it kind of like made it a little bit more fair. But then, on the other hand, something that would happen is they mark you off, for one thing, on the original submission and then on the on your resubmission, they'll like, fix that one in your grade but then they have to like ding you for something else that they didn't even mention before. So I think that also kind of frustrated some people because they feel like it was thrown at them last minute.*

Participant 7 discussed how resubmissions took a lot of time to fill out because if she wanted to resubmit a change to one function header she would need to fill out a form with multiple questions explaining her change:

> *There was resubmission, so I guess that was good. Then I could just try again and get Es but I think it was just really bothersome. In the first place, to have to do all those resubmissions*

Participant 19 discussed how resubmissions were too time consuming. If she focused on resubmissions she would fall behind in other course materials:

> *And they did allow resubmission. But there was so much work every week that, if like, you just would have to try to turn things in, because otherwise you would fall behind.*

4.9 Students expected their grade to correlate with their labor; this misalignment created frustration

If students spent a lot of time completing an assessment, they believed they should have obtained a higher grade based on the effort they had put into the assignment. However, in computer science classes, they did not often see that their labor reflected in their grade, creating frustration because students were not sure what they needed to do to get a higher grade. This constant frustration created fatigue within students and caused them to be unmotivated to complete future assessments.

Participant 10 discussed how she would often go to office hours and work very hard on assignments, but she did not get the grade she believed she deserved. She did not know why she did not get the grade she wanted or what she could have done better:

> *I don't know, because I just I found the class really hard, so I guess I feel like I deserve the grade I got I don't know, though, like I feel like I should have gotten a higher score just because, like I went to office hours like I tried like I really tried, but I get it. But then, at the same time again, I don't know how the class it's also curved, I believe so. I don't even know how the curve is like. and there's no transparency on the curve either, like usually in like chem classes I've taken gen chem and they give you like full transparency of what the curve is going to be.*

Participant 4 discussed how they spent the majority of their time that quarter on completing assessments for a redesigned class. He did not get the grade he expected because TA grading was very inconsistent and the instructor believed the average grade of a 65% was appropriate for the course:

> *He gave us screenshots. He gave us literal screenshots, and he told us to code this out. And then, like, I want you to make this out from code and like, we're gonna post that. And then you're gonna code this into a blank document where he didn't give us any like skeleton code. And he expected us to like code this up. And this took me like probably 30 hours to finish that assignment.... A really bad experience at the end everybody's grade was like tanked. And he was like I'm just gonna keep it the way it is.*

4.10 Exam formats differed from classroom norms which created stress for students

Sometimes exam formats deviated from exam norms and expectations, leaving students feeling unprepared and uncertain about how to approach the exam. For example, students were comfortable writing code on their computer, but felt nervous handwriting code during the final exam because they never practiced it. The change in exam formats caused students to focus on practicing how to take an exam in that format rather than on the content the exam covered.

Participant 17 was added to a course late, and within her first week, there was a quiz. She did not know how to use the Ed platform that other students were already familiar with from previous courses. She had spent more time trying to understand the platform than studying for the quiz:

> *I had skipped [pre-req course]. And went straight to [course] So I didn't actually know how Ed worked or anything like that. And I just was straight into it, like [pre-req course], probably like, gave everyone an introduction to how Ed works and stuff. So when I first took the first test, I was so terrified I was messing up so much, because, like I don't know, it was just stress of being timed when I did the the coding like I wasn't super fast. I wasn't super good. It takes a while to process it. So, like I messed up the first exam*

> *really badly, and it was the most easy exam too when I look back at it.*

Participant 17 also discussed feeling nervous completing an exam on paper. They were not used to doing this and had to sacrifice time studying to practice writing code on paper:

> *It was actually yeah. but I guess they did tell us beforehand that it was going to be on paper. But it was like, I think, like a week before, and they told us that. But everyone else already knew that because they took [course] and they did a final exam on paper, too. So they already knew that. So like I was like, Oh, shoot! So I had I guess they gave us time to practice how to do on code. But I didn't really practice much on that. I just practiced actually coding on my computer instead of like actually writing it down, which is something I probably should have done to prepare myself for the exam.*

Participant 13 discussed how a change in exam format for the final exam was stressful. She explained how the format change was uncomfortable and was very different than anything she did throughout the course:

> *A handwritten exam for the midterm or the final. Which I really didn't like, because we never really did anything with pen and paper. On quiz sections we sort of did, but it just felt so weird to do pen and paper exams when, the entire class you've been doing coding with our computers for assignments, and that so I had to like literally practice, just to see if my hands can, keep up with my mind, it was a timed exam, so that was definitely something I really didn't like about the class. If it was an exam that mimicked what we were doing for assignments, I would have liked that more.*

5 DISCUSSION

Our research revealed many different ways assessment policies could shape marginalized students' experiences in programming classes. We observed ten different interactions between policy and students' identities that resulted in new inequities or heightened existing inequities. Across the set, we found that students did not understand the reasoning behind certain assessment choices, leading to confusion. Students discussed how rules that limited access to help, led to a frustrating lack of support. The combination of different strict policies caused students to feel less motivated to complete their assessments. Moreover, these interactions explained how inequitable policy choices negatively impacted students' sense of belonging in computer science courses through policies that isolated them. Policy choices also weakened self-efficacy and made many students feel like they were not good programmers. Overall, these interactions demonstrate how assessment policies create confusion and frustration, and demotivate students when completing assessments.

5.1 Implications

Despite these limitations, our findings are broadly consistent with how Inoue has conceptualized assessment ecologies, demonstrating how essential it is that instructors consider their students' identities and contexts when designing assessment policies. For example, section 4.9 suggests that students' labor is not accurately reflected within their grades. Inoue's work suggests the use of labor-based grading contracts which would acknowledge students labor as an important factor in students grades [17]. Moreover, section 4.6 discussed how without understanding the assessment purpose it is difficult for students to perform well on the exam. One of Inoue's tenants of equitable assessment ecologies is assessment *purpose*. Assessment purposes are constantly changing and can be different for instructors, students, and institutions. Our findings and Inoue's theory suggests students need instructors to take time to explain why they are choosing certain assessment practices. Additionally, these results are consistent with Inoue's claims about *power* relationships shaping assessment dynamics. In section 4.5 students asked for accommodations, but instructors had the power to decide if students should receive them or not. Additionally, our results suggest that students feel powerless and do not have any agency in deciding assessment policies. This creates a learning environment where it is sometimes impossible to complete assessments. Inoue describes how when power is not shared assessments follow racist practices to determine what is success or failure. He explains a anti-racist assessment ecology would interrogate power dynamics by considering the identities of students when creating the assessment. Therefore, our results suggest that many of Inoue's guidelines on designing equitable assessment policies could be a good tool for computer science instructors to create new assessment policies.

Many different elements of Inoue's work were reflected in our results, but some aspects were not consistent with Inoue's findings. For example, Inoue discusses the importance of *place* and how many academic places impact the way students write. In our results, students did mention places such as lecture halls or lab classes, but not in the context of an assessment or an assessment policy. For example section 4.5, a student completed an exam in another room due to university accommodations, but the impact they mentioned comes more from the people they could interact with rather than the physical space. Additionally, Inoue discusses *people* and how the relationship with the instructor creates culture. Our findings reveal that instructors have some impact on the culture, but most of the culture is created through other students and TAs. This discrepancy may be attributed to how many students discussed not having any relationship with the instructor. These differences highlight how Inoue's might not be directly applied to programming. There are many possibilities for these differences such as Inoue's framework being designed for K-12 writing assessments, differences in the type of assessment, or cultural differences.

Beyond Inoue, our findings suggest that grading schemes that deviate from students' expectations, such as ESNU (E - excellent, S - satisfactory, N - needs improvement, U - unsatisfactory) grading in our study, are unclear when implemented in practice creating inequities. Although the purpose of these grading schemes are to promote equity, students believed they were unfair and did not understand the benefits of using them. When adopting a grading scheme different from the norm of standard based grading, prior work has suggested that it is imperative to communicate the purpose and motivation of this new grading scheme [47]. Students expressed frustration over unconventional grading schemes because they did not have clarity on what was required by them to get a certain grade and how their grade will translate in the 4.0 grading scale the university used. Moreover, the 4.0 grading scheme and non-traditional grading schemes are in tension and the dominant 4.0 grading scale is considered to be more important to students, causing there to be little space for other grading schemes.

Moreover, we found policies surrounding TA interactions were a important part of students' experiences in programming courses. Prior work has suggested teaching assistants help with grading, debugging [32], and creating culture and community in computer science classes [34], our findings suggest TAs are used to communicate and reinforce policy, often acting as a proxy for the professor. Many participants expressed having little communication with the instructor and solely relying on the TA for information about the assessment policies. However, TAs do not design these policies and can give students conflicting answers on questions about policy. In section 4.3, participant 19 expressed TAs giving conflicting answers on assessment content which led to her feeling very confused. Moreover, when policy limited TAs ability to help a student, students turned to other resources for help. In section 4.2, participant 5 explained how they used another support center on campus to get help. Some of these resources were allowed by course policies, but some violated academic integrity. For instance in section 4.2, participant 13 discussed how they contacted other students for help on assignments because they did not understand how the TA was explaining something. From section 4.2 and4.3 we can infer students' inability to get help from TAs is a likely motivation to cheat.

Our findings also suggest that unclear collaboration policies may cause students to cheat unknowingly and create isolation in computer science courses. Prior work show that many students are unsure of cheating policies especially policies surrounding collaboration [22, 43]. We found students feared academic integrity violations so much that they isolated themselves from other people in the class to make sure they were not cheating (4.1. Prior work found that peer networks increase retention of students and help students confidence [46]. Unclear cheating policies that outlaw collaboration is worsening the sense of community in computer science.

We also found that some students make inaccurate assumptions about policy. For example, whereas work suggests that students find it difficult to ask for accommodations in courses, we found that many students made assumptions that they were not able to get support for the challenges they were going through. For example in section 4.4, participant 18 discussed how they were sick with COVID-19 but felt obligated to attend quiz section so she can receive extra credit. There was no policy that stated she could not receive extra credit from different avenues, but she felt it was obligatory to attend this quiz section. Another example in section 4.4, participant 1 discussed how she did not feel comfortable asking for some accommodation during Ramadan because she did not want to ask for too much. There were no formal limits to how many late days she could ask for, but she believed that there could be some sort of limit. This put more burden on her and created a situation where she needed to work at her normally even with a unique circumstance.

Our work found there are many different ways policy shapes the experience of marginalized students, but there is more work to do to get to the goal of equitable assessment policies. Much of this work describes negative implications of these policies because that is what students highlighted. Future work should consider utilizing Inoue's design framework for equitable assessment policies and these findings to design new policies. A similar study should take place that aims to understand how these new policies shape marginalized students experience. Additionally, other future work should understand how often these inequitable assessment policies are being utilized by instructors. Once we understand the prevalence of these policies, we can understand how common bad policies are

Finally, our work also has implication for practice. For example, students explained how grading was inconsistent between TAs which made them feel frustrated. Instructors should consider investing in more grading training to minimize discrepancies between graders. Additionally, if there are discrepancies, students should have a means for submitting regrade requests. Additionally, instructors should consider creating assessments that allow collaboration. Students wanted to be able to learn from each other, but were not able to and had to rely on TAs. This made TA office hours very long, but if collaboration was allowed this could help students work faster and put less burden on TAs. Additionally, instructors should consider allowing students to give input on assessment policies throughout the class. This relates to Inoue's claims about *power* where he describes reconstructing power dynamics between students and instructors by getting students fully involved in the creation of assessment policies [17]. By having a open dialogue, instructors will be able to understand the needs to students and less inequities will arise from their policies.

Overall, these implications for research and practice demonstrate that assessment policy design, like all policies, can have unintended consequences, especially on marginalized students experiences in programming, amplifying inequities that exist, or creating new ones. As the computer science community continues to strive for equity, it is essential that assessment policies are created to support the diverse lives of students, especially those who are marginalized. We hope this work, and the work it builds upon, offers some guidance for how.

5.2 Limitations & Future Work

Our work has valuable contributions, but there are some limitations to the research. With respect to internal validity, the first author conducted all the interviews and had limited interview experience. She also comes from a similar undergraduate computer science experience as many of the participants, so she may have used her personal experience to understand what types of questions to ask. Some of the participants had a relationship with the first author as they were both members of the same outreach organizations. Moreover, participants 9 and 16 did not explicitly disclose if they were a part of any marginalized group, but possibly still could be because we recruited specifically for marginalized students. Future work could explore other populations and focus on recruiting participants from specific marginalized groups.

With respect to external validity, there are some limitations based on the sample population. Many of these participants come from a large research-intensive public school in North America. Therefore, our findings may not be generalizable to students who attend smaller and/or liberal arts colleges. Moreover, students came from a very diverse set of different identities and experiences, which could make it difficult to generalize these results to other students. Future work could explore conducting interview studies at multiple different institutions to ensure these findings are representative of all students.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1539179, 1703304, 1836813, 2031265, 2100296, 2122950, 2137834, 2137312, 2023338658 and unrestricted gifts from Microsoft, Adobe, and Google.

REFERENCES

[1] Kirsti M Ala-Mutka. 2005. A survey of automated assessment approaches for programming assignments. *Computer science education* 15, 2 (2005), 83–102.

[2] Nicole Clark. 2004. Peer testing in software engineering projects. (2004).

[3] Merijke Coenraad, Connor Hopcraft, Jane Jozefowicz, Diana Franklin, Jen Palmer, and David Weintrop. 2021. Helping teachers make equitable decisions: effects of the TEC Rubric on teachers' evaluations of a computing curriculum. *Computer Science Education* 31, 3 (2021), 400–429.

[4] McKinsey & Company. 2023. What is psychological safety? *McKinsey & Company* (2023).

[5] Linda Darling-Hammond, Frank Adamson, and Jamal Abedi. 2010. *Beyond basic skills: The role of performance assessment in achieving 21st century standards of learning.* Stanford Center for Opportunity Pollcy in Education.

[6] Sonal Dekhane, Xin Xu, and Mai Y Tsoi. 2013. Mobile app development to increase student engagement and problem solving skills. *Journal of Information Systems Education* 24, 4 (2013), 299–308.

[7] V Dropčová. 2016. *Peer Review and Peer Assessment in Higher Computer Science Education.* Ph.D. Dissertation. Ph. D. thesis, Comenius University in Bratislava.

[8] J Feldman. 2018. School grading policies are failing children: A call to action for equitable grading. *Oakland, CA: Crescendo Education Group* (2018).

[9] José Figueiredo and Francisco José García-Peñalvo. 2020. Increasing student motivation in computer programming with gamification. In *2020 IEEE Global Engineering Education Conference (EDUCON)*. IEEE, 997–1000.

[10] Mark Frydenberg and Kevin Mentzer. 2020. From engagement to empowerment: project-based learning in Python coding courses. EDISG Conference, Information Systems & Computing Academic Professionals.

[11] Daniel Gebremichael. 2016. An evaluation of gamification to assess students' learning on their understanding of first year computer science programming module. (2016).

[12] Tim Gorichanaz. 2022. "It made me feel like it was okay to be wrong": Student experiences with ungrading. *Active Learning in Higher Education* (2022), 14697874221093640.

[13] Georgiana Haldeman, Andrew Tjang, Monica Babeş-Vroman, Stephen Bartos, Jay Shah, Danielle Yucht, and Thu D Nguyen. 2018. Providing meaningful feedback for autograding of programming assignments. In *Proceedings of the 49th ACM Technical symposium on computer science education.* 278–283.

[14] David Hammer and Leema K Berland. 2014. Confusing claims for data: A critique of common practices for presenting qualitative research on learning. *Journal of the Learning Sciences* 23, 1 (2014), 37–46.

[15] Wynne Harlen, Caroline Gipps, Patricia Broadfoot, and Desmond Nuttall. 1992. Assessment and the improvement of education. *The curriculum journal* 3, 3 (1992), 215–230.

[16] Jack Hollingsworth. 1960. Automatic graders for programming classes. *Commun. ACM* 3, 10 (1960), 528–529.

[17] Asao B Inoue. 2015. *Antiracist writing assessment ecologies: Teaching and assessing writing for a socially just future.* Parlor Press LLC.

[18] SP King and C Amon. 2008. Assessment data: A tool for student and teacher growth. *Data-driven school improvement: Linking data and learning* (2008), 71–86.

[19] F Megumi Kivuva and Camilo Montes De Haro. 2024. Cultural-Centric Computational Embroidery. (2024).

[20] Heru Kuswanto et al. 2018. Android-Assisted Mobile Physics Learning through Indonesian Batik Culture: Improving Students' Creative Thinking and Problem Solving. *International Journal of Instruction* 11, 4 (2018), 287–302.

[21] Raymond Lister, Elizabeth S. Adams, Sue Fitzgerald, William Fone, John Hamer, Morten Lindholm, Robert McCartney, Jan Erik Moström, Kate Sanders, Otto Seppälä, Beth Simon, and Lynda Thomas. 2004. A multi-national study of reading and tracing skills in novice programmers. In *Working Group Reports from ITiCSE on Innovation and Technology in Computer Science Education* (Leeds, United Kingdom) *(ITiCSE-WGR '04)*. Association for Computing Machinery, New York, NY, USA, 119–150. https://doi.org/10.1145/1044550.1041673

[22] Michael Liut, Anna Ly, Jessica Jia-Ni Xu, Justice Banson, Paul Vrbik, and Caroline D Hardin. 2024. " I Didn't Know": Examining Student Understanding of Academic Dishonesty in Computer Science. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1.* 757–763.

[23] Lauri Malmi, Ville Karavirta, Ari Korhonen, and Jussi Nikander. 2005. Experiences on automatically assessed algorithm simulation exercises with different resubmission policies. *Journal on Educational Resources in Computing (JERIC)* 5, 3 (2005), 7–es.

[24] Chris Marriott, Menaka Abraham, and Heather E Dillon. 2023. Labor-based Grading in Computer Science: A Student-Centered Practice. In *2023 ASEE Annual Conference & Exposition.*

[25] Julie A Marsh, John F Pane, and Laura S Hamilton. 2006. Making Sense of Data-Driven Decision Making in Education: Evidence from Recent RAND Research. Occasional Paper. *Rand Corporation* (2006).

[26] Lindsay C Masland. 2023. Ungrading: The joys of doing everything wrong. *Zeal: A Journal for the Liberal Arts* 1, 2 (2023).

[27] Paola Medel and Vahab Pournaghshband. 2017. Eliminating gender bias in computer science education materials. In *Proceedings of the 2017 ACM SIGCSE technical symposium on computer science education.* 411–416.

[28] Joydeep Mitra. 2023. Studying the Impact of Auto-Graders Giving Immediate Feedback in Programming Assignments. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1* (<conf-loc>, <city>Toronto ON</city>, <country>Canada</country>, </conf-loc>) *(SIGCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 388–394. https://doi.org/10.1145/3545945.3569726

[29] Yordanos Mogos and Shasta Ihorn. 2021. Computer Science Identity Development in Diverse Student Populations: A Qualitative Study. In *SIGCSE.*

[30] Shareeda Mohammed and Permanand Mohan. 2019. A Method for Creating Assessment Items for Computer Science. In *2019 14th International Conference on Computer Science & Education (ICCSE)*. IEEE, 1025–1029.

[31] Philip Nagy. 2000. The three roles of assessment: Gatekeeping, accountability, and instructional diagnosis. *Canadian Journal of Education/Revue canadienne de l'éducation* (2000), 262–279.

[32] Christopher O'neal, Mary Wright, Constance Cook, Tom Perorazio, and Joel Purkiss. 2007. The impact of teaching assistants on student retention in the sciences: Lessons for TA training. *Journal of College Science Teaching* 36, 5 (2007), 24.

[33] Leah Perlmutter, Jayne Everson, Ken Yasuhara, Brett Wortzman, and Kevin Lin. 2022. Reading between the lines: Student experiences of resubmission in an introductory CS course. In *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education V. 2.* 1137–1137.

[34] Leah Perlmutter, Jean Salac, and Amy J Ko. 2023. " A field where you will be accepted": Belonging in student and TA interactions in post-secondary CS education. In *Proceedings of the 2023 ACM Conference on International Computing Education Research-Volume 1.* 356–370.

[35] Leah R Perlmutter. 2023. *Student Belonging in Teaching Assistant Interactions and Course Policies in Post-Secondary Computer Science.* Ph.D. Dissertation. University of Washington.

[36] Marcia Rapchak, Africa S Hands, and Merinda Kaye Hensley. 2023. Moving Toward Equity: Experiences With Ungrading. *Journal of Education for Library and Information Science* 64, 1 (2023), 89–98.

[37] Adrian Salguero, William G. Griswold, Christine Alvarado, and Leo Porter. 2021. Understanding Sources of Student Struggle in Early Computer Science Courses. In *Proceedings of the 17th ACM Conference on International Computing Education Research* (Virtual Event, USA) *(ICER 2021)*. Association for Computing Machinery, New York, NY, USA, 319–333. https://doi.org/10.1145/3446871.3469755

[38] Jeffrey Schinske and Kimberly Tanner. 2014. Teaching more by grading less (or differently). *CBE—Life Sciences Education* 13, 2 (2014), 159–166.

[39] Judy Sheard and Martin Dick. 2011. Computing student practices of cheating and plagiarism: a decade of change. In *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education.* 233–237.

[40] Judy Sheard, Martin Dick, Selby Markham, Ian Macdonald, and Meaghan Walsh. 2002. Cheating and plagiarism: Perceptions and practices of first year IT students. In *Proceedings of the 7th annual conference on Innovation and technology in computer science education.* 183–187.

[41] Judy Sheard, Simon, Matthew Butler, Katrina Falkner, Michael Morgan, and Amali Weerasinghe. 2017. Strategies for maintaining academic integrity in first-year computing courses. In *Proceedings of the 2017 ACM Conference on Innovation*

and Technology in Computer Science Education. 244–249.

[42] Mark Sherman, Sarita Bassil, Derrell Lipman, Nat Tuck, and Fred Martin. 2013. Impact of auto-grading on an introductory computing course. *Journal of Computing Sciences in Colleges* 28, 6 (2013), 69–75.

[43] Simon, Beth Cook, Judy Sheard, Angela Carbone, and Chris Johnson. 2013. Academic integrity: differences between computing assessments and essays. In *Proceedings of the 13th Koli Calling International Conference on Computing Education Research* (Koli, Finland) *(Koli Calling '13)*. Association for Computing Machinery, New York, NY, USA, 23–32. https://doi.org/10.1145/2526968.2526971

[44] Scott Spurlock. 2023. Improving student motivation by ungrading. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*. 631–637.

[45] Christopher W Starr, Bill Manaris, and RoxAnn H Stalvey. 2008. Bloom's taxonomy revisited: specifying assessable learning objectives in computer science. *ACM Sigcse Bulletin* 40, 1 (2008), 261–265.

[46] 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*. 65–70.

[47] Robbie Weber. 2023. Using alternative grading in a non-major algorithms course. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*. 638–644.

[48] Kyle M Whitcomb, Sonja Cwik, and Chandralekha Singh. 2021. Not all disadvantages are equal: Racial/ethnic minority students have largest disadvantage among demographic groups in both STEM and non-STEM GPA. *AERA Open* 7 (2021), 23328584211059823.

[49] Benjamin Xie, Matt J Davidson, Baker Franke, Emily McLeod, Min Li, and Amy J Ko. 2021. Domain Experts' Interpretations of Assessment Bias in a Scaled, Online Computer Science Curriculum. In *Proceedings of the Eighth ACM Conference on Learning@ Scale*. 77–89.

[50] Benjamin Xie, Alannah Oleson, Jayne Everson, and Amy J Ko. 2022. Surfacing equity issues in large computing courses with Peer-Ranked, Demographically-Labeled student feedback. *Proceedings of the ACM on Human-Computer Interaction* 6, CSCW1 (2022), 1–39.