PRIMM and Proper: Authentic Investigation in HE Introductory Programming with PeerWise and GitHub

Steven Bradley
Department of Computer Science
Durham University
Durham, UK
s.p.bradley@durham.ac.uk

Anousheh Ramezani
Department of Computer Science
Durham University
Durham, UK
anousheh.ramezani@durham.ac.uk

ABSTRACT

We explore the use of the PRIMM methodology (Predict, Run, Investigate, Modify, Make) within a higher education introductory programming setting, particularly focusing on the three first three steps. Formative prediction questions on the effects of changes to HTML, CSS or JavaScript code are constructed by students using PeerWise system, based on their own investigation. Authenticity of the task is enhanced by presenting the peer prediction questions as pull requests to a GitHub repository, mirroring the code review process followed by professionals working within software development teams. We report on student engagement with the formative practical exercises and analyse the content of the questions they asked.

CCS CONCEPTS

• **Social and professional topics** → **Computer science education**; **Student assessment**.

KEYWORDS

education, programming, assessment

ACM Reference Format:
Steven Bradley and Anousheh Ramezani. 2024. PRIMM and Proper: Authentic Investigation in HE Introductory Programming with PeerWise and GitHub. In *Computing Education Practice (CEP '24), January 05, 2024, Durham, United Kingdom.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3633053.3633062

1 INTRODUCTION: WHY?

In this paper we describe our approach which brings together three different ideas: PRIMM; peer instruction; and authenticity. PRIMM [14] stands for Predict, Run, Investigate, Modify and Make, and has been proposed, adopted and studied as a pedagogical approach for teaching programming, most often used (or at least studied) in schools [15, 16]. Students start with the Predict and Run stages, based on the well-established idea that reading code should be a precursor to writing code [8, 12]. During the Investigate stage students make experimental changes to code that they have been given to develop their understanding of code without "owning" it [7]. This helps to scaffold their learning, and also to alleviate any issues around self-efficacy arising from errors that occur: there are no wrong answers. Also the fact that the code being worked on "belongs" to somebody else reduces any implicit criticism that arises from errors that occur. We are interested in how to use PRIMM within a higher education (HE) environment.

Our second influential idea is peer instruction, which has shown to be effective in teaching programming in multi-institution studies [13, 17]. One common peer instruction practice is for the instructor to ask a question in class which is answered individually by students ("solo vote"), usually electronically. The results are then shown to students who discuss in groups before voting again ("group vote"). Peer instruction is valued by students [13] and there is clear learning gain arising from group discussion [17]. This learning gain varies with question difficulty, so asking the right questions is important — and sharing such questions has become a community activity [19]. It is possible to take peer instruction one stage further by having students create questions for other students to answer, as exemplified by the tool PeerWise [2]. In this approach, rather than students having a "solo vote" and then a "group vote", they answer the questions individually but can comment on and rate the questions being asked. There is evidence from other disciplines that this approach to peer instruction does give learning benefits [3, 10, 18].

The third string to our bow is authenticity, although this is a widely used and often poorly defined term [11]. When first year computing students were asked about how they understood authenticity in assessment, the two most commonly raised themes were "Application to Real-World" and "(Appropriately) Challenging" [9]. In a survey of recent computing graduates, the representation of real-world" issues in undergraduate curricula were explored, and tools such as source-code control and practices such as code reviewing came up as being under-represented in teaching situations [4].

In Section 2 we describe how we combine these three ideas, before evaluating student engagement in Section 3 and drawing conclusions in Section 4. The combination of these three is novel and, we believe, the combination of PRIMM with peer instruction is itself novel, as is the use of GitHub pull requests for PRIMM investigations.

2 WHAT DID WE DO?

The challenge in combining these three ideas is that early PRIMM activities are often — and necessarily, given the school context — presented in an inauthentic way. Predicting the effect of a four line program fragment is far removed from real-world application. Our approach is to make the prediction phase more authentic by

CEP '24, January 05, 2024, Durham, United Kingdom

ACM ISBN 979-8-4007-0932-6/24/01.
https://doi.org/10.1145/3633053.3633062

framing it as a code review, using a standard source-code control tool (git). Rather than having small fragments of code as the basis of the prediction, we offer questions based small changes to an existing code base. And rather than choosing a set of changes fixed by the instructor, the changes and the questions themselves are produced by the students. First we summarise the context, before moving on to a more detailed explanation of what we did.

Context

Our institution is small/medium sized research intensive university in the UK, and the students are studying for BSc/MEng in Computer Science Degree, which has a high entry tariff. Students choose between one of two programming modules in their first year of study, and we are reporting here on the module designed for more experienced students i.e. with A-Level Computer Science or equivalent programming experience. Code review is an important part of the module, with later summative work including an aspect of peer review.

In this module there were 94 students in Academic Year 2022/23, and 55 (59%) of them gave consent for us to use their work in our analysis, following institutional ethical approval[1]. All of the teaching for the module took place face-to-face, after the previous two Covid-affected years.

Teaching consisted of one-hour lectures (whole class) and two-hour practicals (smaller groups). PeerWise exercises were added to the previously existing content of two practical classes in the third and sixth week of the first term. In each case students were given tutorial material to explore before looking at the PeerWise questions. The exercises were formative and there were no associated participation marks.

Modified PRIMM/PeerWise process

Having looked through relevant tutorial material for the practical session students were first asked to answer a PeerWise question before moving on to writing their own question. The instructor had added one initial question before the session, to make sure there was at least one question for all students to look at initially.

In PRIMM, "Predict" and "Run" comes before "Investigate", but we then turned this on its head by getting students to investigate the effects of changes they made to some instructor-provided code. They accessed this code via GitHub: an introduction to git was given in the early lectures and in the first practical session (week 2). They were allowed and encouraged to work with somebody else if they preferred. Their instructions were

- Fork the instructor repository and clone the forked version to the local machine. Set the upstream remote to be the instructor repository
- Within the local copy make some changes, and/or add one or more new files. Investigate until you find a change that has a "useful, interesting, beautiful or surprising result". You can see the result of your change by opening the local file with a web browser
- Commit and push your change to your forked repo
- Make a pull request (PR) from your update fork to the main branch of the instructor repo

[1]COMP-2023-04-27T16_25_43-dcs0spb

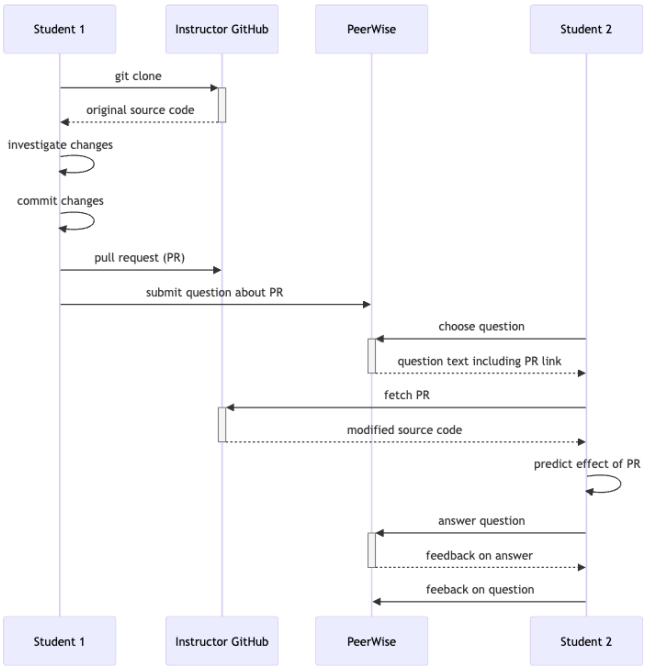

Figure 1: Sequence chart for student interactions with GitHub and Peerwise. In this example Student 1 is asking the question and Student 2 is answering the question, but in practice there are many students who both answer and ask questions

Once the PR based on their investigation was created, students were asked to write a multiple choice PeerWise question about the change, with suggestions of

- how the page looks
- whether this is the only/best way to do something
- whether the page is still valid HTML

They were asked to include a link to their PR in the body of the question so other students could review the changes. The whole process is summarised in Figure 1, albeit simplified to reflect how a single question is produced by one student and answered by another.

The first exercise (week 3) was based on HTML/CSS. Many students would have some familiarity with these concepts, so probably the hardest part of the exercise was the git forking/clone/PR process. The second exercise (week 6) was on client-side JavaScript code, and in the intervening weeks the students engaged in a collaborative exercise using git to jointly develop a simple set of web pages. This collaborative exercise also required them to review the code produced by another group, so they will have gained experience of using. By the time they reached the second PeerWise exercise (week 6) the main challenge was anticipated to be in the understanding of the JavaScript code, rather than the associated git and reviewing process.

Later on in the module students were given Modify tasks to complete i.e. add required functionality to an existing JavaScript program, and during their first piece of summatively assessed coursework they had to create a dynamic website based on a topic of their own choosing.

3 EVALUATION: DOES IT WORK?

Denny et al. had four measures of student engagement with PeerWise that they correlated with subsequent exam performance: Number of questions created (Q); Number of questions answered (A); Total length of comments (C); Number of active days (D)[2]. They found that each of these separately were significantly correlated with performance in a final MCQ exam, and that a combined meaure of them all had the strongest correlation both with MCQ exam score and also a final non-MCQ exam. Here we look at the first two of these measures, as they are relatively easy to derive, but first we examine the extent to which students engaged at all, given the common problem of non-engagement with formative assessment.

Of the 55 students that gave consent for us to use their data, 44 (80%) submitted an answer to at least one of the questions. There are multiple reasons why students may not have answered a question, including:

- The student was absent from the session
- The student didn't get through the tutorial material in the time allotted to the practical session, so did not get to the PeerWise exercise
- Students worked collaboratively on the practical session, so one submission was made on behalf of two or more students
- There were issues registering with PeerWise

Taking all of these possibilities into account, 80% seems like a good rate of engagement. Other work exploring the use of formative online quizzes (albeit in a different discipline) had a participation rate of 50-60%, only rising to 90% after specific interventions were made to increase participation [6]. When the issues affecting participation in formative assessment were studied, four key areas were identified: inadequate feedback; curriculum organization and mistrust; time constraint; and fear of judgment [1]. Within PeerWise students are allowed to choose their own username for display, so could anonymise themselves if they desired, hopefully allaying some of the fear of judgment. There were certainly time constraints in place, but the quality of feedback and relevance to the curriculum seems to have been strong enough to encourage students to engage. shows how many questions were answered (by consenting students who answered any). The practical work set only asked students to answer one question and to ask one question, but it is apparent from Figure 2 that many students answered more than one question. The average number of questions answered was 2.7 and indeed one student chose to answer 23 questions, indicating engagement outside of the timetabled practical class.

Questions were submitted by 26 (47%) distinct students. Given that students were allowed/encouraged to work together to write questions (which they weren't for answering questions) this is almost certainly an underestimate of the number of students engaging in question writing.

Of the 35 submitted questions 9 (26%) did not include a link to a PR, which may have been either because they did not make a PR or

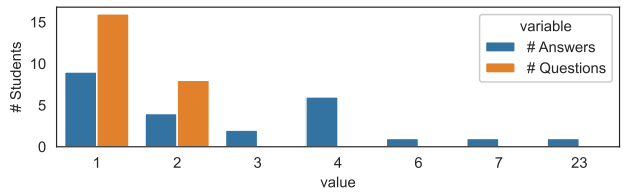

Figure 2: Distribution of the number of questions set and answers given by students. Note that the x-axis is not linear.

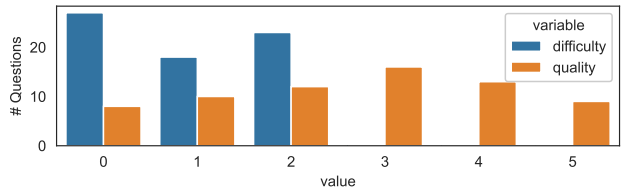

Figure 3: Distribution of student ratings of question difficulty and quality. For difficulty the range goes from Easy (0) to Hard (2). For quality the range goes from Very Poor (0) to Excellent (5)

they forgot to include a link to it. So at least 74% did include a PR, indicating engagement both with GitHub and PeerWise.

Earlier work on peer instruction found that the quality and difficulty of the questions asked had a significant impact on the learning gain due to the question [17]. Figure 3 shows the student ratings of the difficulty and quality of the questions asked. The questions had a range of difficulty, with "Easy" (0) being the most common, followed by "Hard" (2), and overall slightly skewed towards "Easy". The quality of questions was skewed towards "Excellent" (5), with the most common being "Good" (3) and "Very Good" (4).

The content of the questions is summarised in the word cloud of Figure 4[2]. We can see that question content is relevant to the subject of study (HTML and JavaScript), addressing the "curriculum organization and mistrust" concern, particularly "since misalignment between formal curriculum and quizzes was a major source of mistrust" [1].

There are multiple threats to the validity of the data presented including the usual issues of generalisation from small numbers at a single institution. Also, as noted above, students may well have worked together to write questions, and we have no good way to correct for that. Finally, it is possible that the students who consented to their data being used were not a representative sample of the whole class: it is not inconceivable that students who do consent are more compliant in general, and hence more likely to engage with formative exercises.

4 CONCLUSIONS: IS THIS IMPORTANT?

We have seen that students engaged well in our PRIMM/ PeerWise/ GitHub combination. Evidence from studies on PeerWise suggests that engagement with this kind of peer instruction leads to learning

[2]Generated by wordclouds.com

Figure 4: Word cloud of question content. Size of words corresponds to frequency of occurence, common "stop words" are removed. Questions are relevant to content (HTML, JavaScript)

gain [3]. There is also evidence that in at least some forms of peer instruction the difficulty of the questions asked has an impact on learning gain (easier questions are better) [17], so it is encouraging to see that within the range of difficulties is skewed towards the easier end of the spectrum. Question quality is most often good or better.

The methods of teaching programming have recently come under some scrutiny, with some emphasising "direct instruction" over the "discovery-based" approaches which have been influential for many years [5]. A balance between these two approaches is achieved through the PRIMM method, with students moving from "direction instruction" of predicting the outcome of a given programme execution through to eventually making their own programmes independently. Involving students in the earlier stages of prediction through investigation moves the balance back towards a discovery approach, which may be more appropriate at a higher education level, as opposed to the primary/secondary school level where PRIMM has mostly been implemented before [15].

The real-world aspect of authentic teaching and assessment is known to be valued by students [9], and the use of code reviews through GitHub addresses recent graduates' concerns with preparation for the real world through computing degrees [4]. We found that the large majority of contributed questions demonstrated an ability to use GitHub, which is promising.

5 WHAT NOW: FURTHER WORK

We hope to extend this experience into a more detailed research study in several possible directions: development of more sessions and modules; correlation of engagement with summative assessment outcomes; interviews to see how/why students engage and what their experience is; comparison between student-generated and instructor-generated questions; evaluation of learning gain.

REFERENCES

[1] Ancil J. Abney, Sarina Amin, and Jonathan D. Kibble. 2017. Understanding factors affecting participation in online formative quizzes: an interview study. *Advances in Physiology Education* 41, 3 (Sept. 2017), 457–463. https://doi.org/10.1152/advan.00074.2017

[2] Paul Denny, John Hamer, Andrew Luxton-Reilly, and Helen Purchase. 2008. PeerWise: students sharing their multiple choice questions. In *Proceedings of the Fourth international Workshop on Computing Education Research (ICER '08)*. ACM, 51–58. https://doi.org/10.1145/1404520.1404526

[3] Paul Denny, Brian Hanks, Beth Simon, and Spencer Bagley. 2011. PeerWise: exploring conflicting efficacy studies. In *Proceedings of the seventh international workshop on Computing education research (ICER '11)*. ACM, 53–60. https://doi.org/10.1145/2016911.2016924

[4] Marisa Exter, Secil Caskurlu, and Todd Fernandez. 2018. Comparing Computing Professionals' Perceptions of Importance of Skills and Knowledge on the Job and Coverage in Undergraduate Experiences. *ACM Transactions on Computing Education* 18, 4 (Nov. 2018), 21:1–21:29. https://doi.org/10.1145/3218430

[5] Robin K. Hill and Mark Guzdial. 2019. Pondering variables and direct instruction. *Commun. ACM* 62, 4 (March 2019), 6. https://doi.org/10.1145/3311717

[6] Jonathan D. Kibble. 2011. Voluntary participation in online formative quizzes is a sensitive predictor of student success. *Advances in Physiology Education* 35, 1 (March 2011), 95–96. https://doi.org/10.1152/advan.00053.2010

[7] Irene Lee, Fred Martin, Jill Denner, Bob Coulter, Walter Allan, Jeri Erickson, Joyce Malyn-Smith, and Linda Werner. 2011. Computational thinking for youth in practice. *ACM Inroads* 2, 1 (Feb. 2011), 32–37. https://doi.org/10.1145/1929887.1929902

[8] Raymond Lister, Colin Fidge, and Donna Teague. 2009. Further evidence of a relationship between explaining, tracing and writing skills in introductory programming. *ACM SIGCSE Bulletin* 41, 3 (July 2009), 161–165. https://doi.org/10.1145/1595496.1562930

[9] Roger McDermott, Mark Zarb, Mats Daniels, and Ville Isomöttönen. 2017. First Year Computing Students' Perceptions of Authenticity in Assessment. In *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '17)*. ACM, 10–15. https://doi.org/10.1145/3059009.3059062

[10] Steven Moore, Huy Anh Nguyen, and John Stamper. 2021. Examining the Effects of Student Participation and Performance on the Quality of Learnersourcing Multiple-Choice Questions. In *Proceedings of the Eighth ACM Conference on Learning @ Scale (L@S '21)*. ACM, 209–220. https://doi.org/10.1145/3430895.3460140

[11] Valentina Nachtigall, David Williamson Shaffer, and Nikol Rummel. 2022. Stirring a Secret Sauce: A Literature Review on the Conditions and Effects of Authentic Learning. *Educational Psychology Review* 34, 3 (Sept. 2022), 1479–1516. https://doi.org/10.1007/s10648-022-09676-3

[12] Greg L. Nelson, Benjamin Xie, and Amy J. Ko. 2017. Comprehension First: Evaluating a Novel Pedagogy and Tutoring System for Program Tracing in CS1. In *Proceedings of the 2017 ACM Conference on International Computing Education Research (ICER '17)*. ACM, 2–11. https://doi.org/10.1145/3105726.3106178

[13] Leo Porter, Dennis Bouvier, Quintin Cutts, Scott Grissom, Cynthia Lee, Robert McCartney, Daniel Zingaro, and Beth Simon. 2016. A Multi-institutional Study of Peer Instruction in Introductory Computing. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education (SIGCSE '16)*. ACM, 358–363. https://doi.org/10.1145/2839509.2844642

[14] Sue Sentance and Jane Waite. 2017. PRIMM: Exploring pedagogical approaches for teaching text-based programming in school. In *Proceedings of the 12th Workshop on Primary and Secondary Computing Education (WiPSCE '17)*. ACM, 113–114. https://doi.org/10.1145/3137065.3137084

[15] Sue Sentance, Jane Waite, and Maria Kallia. 2019. Teachers' Experiences of using PRIMM to Teach Programming in School. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19)*. ACM, 476–482. https://doi.org/10.1145/3287324.3287477

[16] Sue Sentance, Jane Waite, and Maria Kallia. 2019. Teaching computer programming with PRIMM: a sociocultural perspective. *Computer Science Education* 29, 2-3 (July 2019), 136–176. https://doi.org/10.1080/08993408.2019.1608781

[17] Cynthia Taylor, Jaime Spacco, David P. Bunde, Andrew Petersen, Soohyun Nam Liao, and Leo Porter. 2018. A multi-institution exploration of peer instruction in practice. In *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE 2018)*. ACM, 308–313. https://doi.org/10.1145/3197091.3197144

[18] Youness Touissi, Ghita Hjiej, Abderrazak Hajjioui, Azeddine Ibrahimi, and Maryam Fourtassi. 2022. Does developing multiple-choice Questions Improve Medical Students' Learning? A Systematic Review. *Medical Education Online* 27, 1 (Dec. 2022), 2005505. https://doi.org/10.1080/10872981.2021.2005505 Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/10872981.2021.2005505.

[19] Craig Zilles, David P. Bunde, Jaime Spacco, Cynthia Lee, Leo Porter, and Cynthia Taylor. 2023. Spiffy Peer Instruction Questions. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 2 (SIGCSE 2023)*. ACM, 1226–1227. https://doi.org/10.1145/3545947.3569590