Comparing the Impact of Programming Assessment Type: In-Class Vs Take-Home

https://doi.org/10.3991/ijep.v10i4.13509

Oscar Karnalim (✉), Gisela Kurniawati, Sendy Ferdian Sujadi, Rossevine Artha Nathasya
Maranatha Christian University, Bandung, Indonesia
oscar.karnalim@it.maranatha.edu

Abstract—In engineering education, educational technologies and teaching interventions are often used to maintain the student retention high. Most of them are strongly related to either in-class or take-home assessment. This paper compares the impact of both assessments on student performance for one academic semester. The findings are expected to re-align the focus of educational technology and teaching intervention research. Fifty-five laboratory sessions were analyzed, involving five Introductory Programming classes with a total of 87 Information Technology and Information System undergraduates. The study shows that, by considering only significant changes, in-class assessment leads to higher assessment mark as the students can easily seek help and focus on the tasks. Technical problems are also unlikely to occur as the classroom (or the laboratory in our case) is well-facilitated.

Keywords—Assessment type, programming, educational technology, teaching intervention, engineering education

1 Introduction

In the academic environment, a faculty or a department is often perceived as successful if they are able to maintain the student retention high [1]. A number of techniques are then developed in favor to such requirement [2], namely: Student Success Course [3], [4], the use of persuasive social media [5], [6], the integration of educational technologies [7], [8], and the introduction of new teaching intervention [9], [10]. For engineering majors, the last two are commonly applied.

A number of educational technologies have been utilized in engineering related major. They are ranging from hardware devices (e.g., electronic voting clicker [11], LEGO MindStorms [12], Arduino [13], or Raspberry Pi [14]) to educational software (e.g., Algorithm Visualization [15], Program Visualization [16], or Visual Programming tool [17]). In general, these technologies share the same objective: to assist students in learning.

There are several teaching interventions that are often used in engineering. Flipped classroom [18], [19] encourages students to learn the instructional material outside the class session so that the session can be used to do the activity or assessment. Team-

based learning [20], [21] requires the students to learn as a group. Game bootstrapping [22], [23] encapsulates the teaching material as games to motivate students. Pair programming [24], [25] lets the students to complete a programming task as pairs. Peer code review [26], [27] encourages each student reviews another' code and provides some feedback.

In applying those educational technologies and teaching interventions, assessments are commonly involved to help students reflecting their understanding toward given materials. These assessments can be classified to two types: in-class and take-home.

To the best of our knowledge, no studies compare the impact of those assessment types on student performance, despite many comparative studies measuring the impact of the educational technologies and teaching interventions [28]–[30]. We believe comparing those two assessment types, finding the most suitable one for students, can realign the focus of educational technology and teaching intervention research. If in-class assessment is preferred, educational technologies should facilitate some in-class assessments [31] or it should be used for completing such assessments [16]. Further, teaching interventions like pair programming and code peer review should be performed during the class sessions. Teaching interventions focusing on in-class assessment like flipped classroom should also gain more attention. If take-home assessment is preferred, educational technologies can focus more on assisting students in understanding the materials. The allocated time for completing take-home assessment is not as tight as the in-class one. Hence, the technologies can be used by the students without time pressure. In addition, teaching interventions in classroom can focus more in understanding the course material (e.g., team-based learning and game bootstrapping). Whereas, teaching interventions outside the classroom can focus more on completing assessments (e.g., pair programming).

This study compares the impact of in-class and take-home assessments on student performance for one academic semester. The recorded data covers undergraduate students' assessment marks, involving three Information Technology classes and two Information System classes. All of them were taking Introductory Programming on their corresponding major, resulting programming as the assessment context.

2 Method

The study was conducted on Introductory Programming, a course where the students first learn how to code. In our faculty, this course was offered in two majors for undergraduates: Information Technology (IT) and Information System (IS). To compare the impact of in-class and take-home assessments, students enrolled in this course were asked to complete both assessments weekly, and their marks for each week would be further analyzed.

For IT students, the in-class assessment should be completed within two hours in a laboratory session. The take-home assessment was given at the end of that laboratory session and it was expected to be completed prior the next week's session. Both assessments were made to be comparable; each of them has three different Python programming tasks: easy, medium, and challenging. We are aware that while completing the

take-home one, students probably have understood given course material further. Hence, the difficulty of the take-home assessment was slightly increased.

For IS students, the scenario is somewhat similar except that the course materials were given in a slightly different order. Further, the programming tasks should be solved with both Python and Java at once (i.e., two solutions per programming task). The lecturers believe that completing the tasks in two programming languages can encourage the students not to rely heavily on one programming language. Regardless, as our focus is not on that issue, no further discussion will be given.

The statistics of the recorded data can be seen in Table 1. IT-A, IT-B, IT-C are IT classes while the rest are IS classes. Both IT and IS classes actually had the same number of sessions (14). However, some of the sessions were used for tests, quizzes or review. IS-B has fewer laboratory sessions than another IS class due to human error. In total, there are 55 analyzed laboratory sessions with a total of 87 undergraduate students on board.

Table 1. Data statistics

Metric	IT-A	IT-B	IT-C	IS-A	IS-B	Total
Students	19	17	25	13	13	87
Analyzed laboratory sessions	12	12	12	10	9	55

The analysis was performed by pairing each in-class assessment with its corresponding take-home assessment from the same week. Their average marks would be calculated, and the significance was measured with two-tailed paired t-test with 95% confidence rate.

For each pair, students who only completed one of the assessments was excluded from the comparison. Any students who were indicated to plagiarize or collude would also be removed as their marks do not reflect their real performance. Plagiarism and collusion suspicion were raised manually by the lecturer and/or the tutors, mainly based on the shared similarities from given source code files.

After that, the changes of significant pairs would be observed. If most of them show higher mark for in-class assessment, it can be stated that in-class assessment is preferred. Otherwise, take-home assessment can be more beneficial. A discussion about the result would be given at the end of the analysis.

3 Result and Discussion

Among 55 analyzed laboratory sessions, eighteen of them have statistically significant difference between student marks for in-class and take-home assessments. The details can be seen in Table 2. Each session ID has three components (major, class, and session number) and they are separated with hyphen. In-class and take-home marks are the average value of all eligible corresponding marks for that week (i.e., those from students who completed both types of assessment with no suspicion of plagiarism and collusion). The highest value among those two marks is written in bold and the p-value can be seen in the next right column, followed by the covered material.

Table 2. Statistically significant change between in-class and take-home assessments

Session ID	In-class mark	Take-home mark	p-value	Covered material
IT-A-03	100	97.3	0.04	Branching
IT-A-06	96.2	99.6	0.01	Nested looping
IT-A-07	83.4	96,4	0.008	Function
IT-B-02	92.6	97.4	0.02	Input
IT-B-09	95	75.1	0.03	Array
IT-B-10	90.3	70.7	0.005	Function + Array
IT-B-11	93.2	68.5	0.001	Matrix
IT-C-02	95.7	99.5	0.01	Input
IT-C-04	89	79.5	0.02	Nested branching
IT-C-05	93.6	89.2	0.02	Looping
IT-C-11	86.8	80.9	0.009	Matrix
IS-A-01	57.5	78,3	0.007	Output
IS-A-05	98.2	71.1	0.01	Looping
IS-A-07	96.7	75.2	0.002	Review
IS-A-08	95	76.8	0.04	Array
IS-B-05	91.5	54.1	< 0.001	Looping
IS-B-08	90.4	66.4	0.01	Array
IS-B-10	46.5	100	< 0.001	Function

Only a third of the significant changes favors take-home assessment, even though we expected that kind of assignment would outperform the counterpart as the allocated time is longer. Some students might need a considerable amount of time in understanding the task and then solving it. However, it seems that these students are more in need of guidance rather than longer allocated time. They need some hints in order to solve the tasks. Despite the hints can be gained by asking smart students or tutors outside the class sessions, only few of the students and the tutors are available to help and can be met in person.

Longer allocated time in take-home assessments can also lead more constructive discussion and more access to helpful resources (e.g., textbook and internet). But again, it seems that a little guidance is required to initiate those discussion and resource access.

Majority of the significant changes favor in-class assessment. During the completion of the assessment, it is easier to seek help, as the tutors have allocated that time for helping the students and they can be met in person. It is also easier to discuss how to solve the tasks with friends as all students are focused on solving the same tasks.

Another benefit of having in-class assessment is that the allocated time should be mainly used for completing the assessment. They cannot be distracted by other activities as they are being monitored by the lecturer and/or the tutors. Outside the classroom, the students can jump into other activities as no one monitors them.

In-class assessment also enforces the students to complete the tasks in the same place (e.g., the classroom or the laboratory), and that place is commonly well-facilitated. Many technical problems like hardware or software errors are unlikely to occur and delay the completion of the assessment. This can work in reverse for take-home assessment as student-owned computers are not guaranteed to facilitate that completion due to a wide range of computer specifications and variants. Further, not all students have

their own computer. Some of them need to share a computer with their family, which can also delay the completion of take-home assessment.

No strong findings about the covered materials can be gained from the significant changes, even though some materials seem to be frequently occurred. However, we can say that in-class assessment is likely to be beneficial on looping (but not the nested one) and array-related materials (array, matrix, and function + array).

4 Limitations of the Study

Our study has three limitations. First, the study was performed only on introductory programming students in which the assessments are purely programming. The findings can be different if the students enrolled in other courses are taken into consideration, especially when the assessments are not programming-related. Second, the suspicion of plagiarism and collusion was raised manually, and it is possible that some of the perpetrators were not caught. The use of automated similarity detection tool like JPlag [32] is recommended to solve this limitation in the future. Third, our study ignores laboratory sessions with insignificant changes as our intention is to know which assessment type leads to higher assessment marks, not whether the change of assessment type affects assessment marks. Further study is needed to confirm the second statement.

5 Conclusion and Future Work

This study compares the impact of in-class and take-home assessments based on student performance for one academic semester. With only significant differences on board, it shows that in-class assessment can be more beneficial as the students can easily seek help and they can focus on completing the tasks with minimum distraction. Further, this kind of assessment mitigates the occurrence of technical problems as the classroom or the laboratory is commonly well-facilitated.

Based on the findings, it is suggested that in-class educational technologies should be able to assess the students' performance, while pair programming and code peer review should be conducted during the class or the laboratory session. We also believe that flipped classroom can be beneficial as the assessment is commonly completed during the class session.

For future work, we plan to conduct a larger study involving students enrolled in more advanced courses like Advanced Object-Oriented Programming or Artificial Intelligence. We also plan to do similar study on courses which assessments are not purely programming like Project Management or Calculus.

6 References

[1] L. Wild and L. Ebbers, "Rethinking Student Retention in Community Colleges," Community College Journal of Research and Practice, vol. 26, no. 6, pp. 503-519, 2002. https://doi.org/10.1080/2776770290041864

[2] K. S. Hone and G. R. El Said, "Exploring the factors affecting MOOC retention: A survey study," Computers & Education, vol. 98, pp. 157-168, 2016. https://doi.org/10.1016/j.compedu.2016.03.016
[3] K. Kimbark, M. L. Peters, and T. Richardson, "Effectiveness of the Student Success Course on Persistence, Retention, Academic Achievement, and Student Engagement," Community College Journal of Research and Practice, vol. 41, no. 2, pp. 124-138, 2017. https://doi.org/10.1080/10668926.2016.1166352
[4] S.-W. Cho and M. M. Karp, "Student Success Courses in the Community College," Community College Review, vol. 41, no. 1, pp. 86-103, 2013. https://doi.org/10.1177/0091552112472227
[5] S. Zheng, K. Han, M. B. Rosson, and J. M. Carroll, "The Role of Social Media in MOOCs: How to Use Social Media to Enhance Student Retention," in Proceedings of the Third (2016) ACM Conference on Learning @ Scale - L@S '16, 2016, pp. 419-428. https://doi.org/10.1145/2876034.2876047
[6] C. Woodley and C. Meredith, "Supporting Student Transition Through Social Media," American Journal of Distance Education, vol. 26, no. 2, pp. 86-95, 2012. https://doi.org/10.1080/08923647.2012.655055
[7] H. M. Truong, "Integrating learning styles and adaptive e-learning system: Current developments, problems and opportunities," Computers in Human Behavior, vol. 55, pp. 1185-1193, 2016. https://doi.org/10.1016/j.chb.2015.02.014
[8] L. De-Marcos, A. Domínguez, J. Saenz-de-Navarrete, and C. Pagés, "An empirical study comparing gamification and social networking on e-learning," Computers & Education, vol. 75, pp. 82-91, 2014. https://doi.org/10.1016/j.compedu.2014.01.012
[9] D. Fung, "Promoting critical thinking through effective group work: A teaching intervention for Hong Kong primary school students," International Journal of Educational Research, vol. 66, pp. 45-62, 2014. https://doi.org/10.1016/j.ijer.2014.02.002
[10] L. Dacre Pool and P. Qualter, "Improving emotional intelligence and emotional self-efficacy through a teaching intervention for university students," Learning and Individual Differences, vol. 22, no. 3, pp. 306-312, 2012. https://doi.org/10.1016/j.lindif.2012.01.010
[11] Q. I. Cutts and G. E. Kennedy, "Connecting Learning Environments Using Electronic Voting Systems," in Proceedings of the 7th Australasian Conference on Computing Education - Volume 42, 2005, pp. 181-186.
[12] H.-N. Liang, C. Fleming, K. L. Man, and T. Tillo, "A first introduction to programming for first-year students at a Chinese university using LEGO MindStorms," in The 2013 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE), 2013, pp. 233-238. https://doi.org/10.1109/TALE.2013.6654435
[13] M. El-Abd, "A Review of Embedded Systems Education in the Arduino Age: Lessons Learned and Future Directions," International Journal of Engineering Pedagogy (iJEP), vol. 7, no. 2, pp. 79-93, 2017. https://doi.org/10.3991/ijep.v7i2.6845
[14] M. Wirth and J. McCuaig, "Making Programs With The Raspberry Pi," in The 2014 Western Canadian Conference on Computing Education - WCCCE '14, 2014, p. 17. https://doi.org/10.1145/2597959.2597970
[15] S. Halim, Z. Chun KOH, V. Bo Huai LOH, and F. Halim, "Learning Algorithms with Unified and Interactive Web-Based Visualization," Olympiads in Informatics, vol. 6, pp. 53-68, 2012.
[16] R. A. Nathasya, O. Karnalim, and M. Ayub, "Integrating program and algorithm visualisation for learning data structure implementation," Egyptian Informatics Journal, vol. 20, no. 3, pp. 193-204, Nov. 2019. https://doi.org/10.1016/j.eij.2019.05.001

[17] M. Kölling and Michael, "The Greenfoot programming environment," ACM Transactions on Computing Education, vol. 10, no. 4, pp. 1-21, Nov. 2010. https://doi.org/10.1145/1868358.1868361

[18] J. Moffett, "Twelve tips for 'flipping' the classroom," Medical Teacher, vol. 37, no. 4, pp. 331-336, 2015. https://doi.org/10.3109/0142159X.2014.943710

[19] C. Demetry, "Work in progress - An innovation merging 'classroom flip' and team-based learning," in 2010 IEEE Frontiers in Education Conference (FIE), 2010, p. T1E-1-T1E-2. https://doi.org/10.1109/FIE.2010.5673617

[20] P. Lasserre, "Adaptation of team-based learning on a first term programming class," in The 14th annual ACM SIGCSE conference on Innovation and technology in computer science education, 2009, vol. 41, p. 186. https://doi.org/10.1145/1595496.1562937

[21] P. Lasserre and C. Szostak, "Effects of team-based learning on a CS1 course," in Proceedings of the 16th annual joint conference on Innovation and technology in computer science education - ITiCSE '11, 2011, p. 133. https://doi.org/10.1145/1999747.1999787

[22] G. Kiss and Z. Arki, "The Influence of Game-based Programming Education on the Algorithmic Thinking," Procedia - Social and Behavioral Sciences, vol. 237, pp. 613-617, 2017. https://doi.org/10.1016/j.sbspro.2017.02.020

[23] K. Puritat, "Enhanced Knowledge and Engagement of Students Through the Gamification Concept of Game Elements," International Journal of Engineering Pedagogy (iJEP), vol. 9, no. 5, pp. 41-54, 2019. https://doi.org/10.3991/ijep.v9i5.11028

[24] N. Salleh, E. Mendes, and J. Grundy, "Investigating the effects of personality traits on pair programming in a higher education setting through a family of experiments," Empirical Software Engineering, vol. 19, no. 3, pp. 714-752, 2014. https://doi.org/10.1007/s10664-012-9238-4

[25] M. Ayub, O. Karnalim, R. Risal, W. F. Senjaya, and M. C. Wijanto, "Utilising pair programming to enhance the performance of slow-paced students on Introductory Programming," Journal of Technology and Science Education, vol. 9, no. 3, pp. 357-367, 2019. https://doi.org/10.3926/jotse.638

[26] C. Hundhausen, A. Agrawal, D. Fairbrother, and M. Trevisan, "Integrating pedagogical code reviews into a CS 1 course," in The 40th ACM technical symposium on Computer sci-ence education, 2009, vol. 41, pp. 291-295. https://doi.org/10.1145/1539024.1508972

[27] C. D. Hundhausen, P. Agarwal, and M. Trevisan, "Online vs. face-to-face pedagogical code reviews," in The 42nd ACM technical symposium on Computer science education, 2011, pp. 117-122. https://doi.org/10.1145/1953163.1953201

[28] A. Luxton-Reilly et al., "Introductory programming: a systematic literature review," in 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education, 2018, pp. 55-106. https://doi.org/10.1145/3293881.3295779

[29] T. Unruh, M. L. Peters, and J. Willis, "Flip This Classroom: A Comparative Study," Computers in the Schools, vol. 33, no. 1, pp. 38-58, Jan. 2016. https://doi.org/10.1080/07380569.2016.1139988

[30] O. Karnalim and M. Ayub, "A Quasi-Experimental Design to Evaluate the Use of PythonTutor on Programming Laboratory Session," International Journal of Online and Biomedical Engineering (iJOE), vol. 14, no. 2, pp. 155-164, 2018. https://doi.org/10.3991/ijoe.v14i02.8067

[31] O. Debdi, M. Paredes-Velasco, and J. Á. Velázquez-Iturbide, "GreedExCol, A CSCL tool for experimenting with greedy algorithms," Computer Applications in Engineering Education, vol. 23, no. 5, pp. 790-804, Sep. 2015. https://doi.org/10.1002/cae.21655

[32] L. Prechelt, G. Malpohl, and M. Philippsen, "Finding plagiarisms among a set of programs with JPlag," Journal of Universal Computer Science, vol. 8, no. 11, pp. 1016-1038, 2002.

7 Authors

Oscar Karnalim graduated with a Bachelor of Engineering from Parahyangan Catholic University in 2011 and completed his Master at Bandung Institute of Technology (ITB) in 2014. He works at Faculty of Information Technology,Maranatha Christian University as a full-time lecturer and he is currently pursuing a PhD in Information Technology at University of Newcastle, Australia. His research interests are computing education, software engineering, information retrieval, and artificial intelligence.

Gisela Kurniawati graduated with a Bachelor of Computer from Maranatha Christian University in 2019. She works at Faculty of Information Technology, Maranatha Christian University as a lecturer. Her research interests are computing education and software engineering.

Sendy Ferdian Sujadi graduated with a Bachelor of Computer from Maranatha Christian University in 2011 and completed his Master at Bandung Institute of Technology (ITB) in 2017. He works at Faculty of Information Technology, Maranatha Christian University as a full-time lecturer. His research interests are software engineering and information system.

Rossevine Artha Nathasya graduated with a Bachelor of Computer from Maranatha Christian University in 2019. She works at Faculty of Information Technology, Maranatha Christian University as a lecturer. Her research interests are computing education and software engineering.

Article submitted 2020-02-02. Resubmitted 2020-03-30. Final acceptance 2020-03-31. Final version published as submitted by the authors.