7-2022

Designing flipped learning activities for beginner programming course

Benjamin GAN
Singapore Management University, benjamingan@smu.edu.sg

Eng Lieh OUH
Singapore Management University, elouh@smu.edu.sg

Designing Flipped Learning Activities for Beginner Programming Course

Completed Research

Benjamin Kok Siew Gan
Singapore Management University
benjamingan@smu.edu.sg

Eng Lieh Ouh
Singapore Management University
elouh@smu.edu.sg

Abstract

This study focuses on designing flipped classroom learning activities across pre-class problem-based exercises; with in-class active discussions and practical problem-solving sessions; and follow up with post-class problem-based labs and assessments. We evaluate the effectiveness of our learning activities based on student surveys, course feedback, grades, and teacher feedback for a beginner programming course with non-IS students. We describe detail programming learning activities with comparisons to existing practices based on related work. Our findings are that majority of students (86%) agreed with flipped classroom, but teachers should be aware of the 14% who disagreed and cater for them. Teachers should avoid overwhelming students with extra exercises (making it optional) but may challenge students who want them and have confidence. With data provided, we hope teachers can make a better-informed decision when choosing flipped classroom, problem-based learning activities for beginner programming course.

Keywords

Learning activities, flipped classroom, problem-based learning, beginner programming course.

Introduction

When designing a beginner programming course, teachers should consider the diversity of student background (Mironova et al., 2016), confidence level (McCord and Jeldes, 2019), engagement level (Rogers et al., 2021; Sprint and Fox, 2020), multiple learning approaches (Kay et al., 2019), available teaching tools (Baniassad et al., 2021; Corley et al., 2020; Fabic et al. 2018), and limited class time (Gan and Ouh, 2019), especially when there are non-IS students (Carr et al., 2020; Khuri et al., 2020). This study focuses on designing the learning activities across pre-class, in-class, and post-class sessions. Our learning activities were based on the flipped classroom and problem-based approaches (Çakıroğlu and Öztürk, 2017; Chiang, 2017; Chis et al., 2018).

There are studies on beginner programming courses for non-IS students (Carr et al., 2020; Dodds et al., 2021; Estell et al., 2021; Khuri et al., 2020; McCord and Jeldes, 2019; Mironova et al., 2016; Salloum et al. 2021), learning activities (Chi, 2009; Fabric et al., 2018; Gan and Ouh, 2019; Irani and Denaro, 2020; Kay et al., 2019; Ouh and Gan, 2021; Sivilotti and Pike, 2007) and flipped classrooms approach (AlJarrah et al., 2018; Çakıroğlu and Öztürk, 2017; Davenport, 2018; Isomöttönen and Tirronen. 2016; Moore at al., 2021; Rosiene and Rosiene, 2015; Souza and Rodrigues, 2015; Zainuddin et al., 2019). However, we are not aware of any with focus on how these flipped classroom learning activities are effective for a beginner programming course with non-IS students. Here are our research questions.

R1. Which flipped classroom learning activities keep student interest?
R2. Which flipped classroom learning activities help students to learn?
R3. What are student views on a flipped classroom approach?

The following sections review the literature for related works, describe the learning activities, present our research method, findings, discussions, and threats to validity.

Related Work

We begin our literature study with the students: beginners in a programming course with non-IS students. It continues with the learning activities and approaches: flipped classroom and problem based.

Beginner programming course

In a beginner programming course with non-IS students, these factors are of concern: confidence, belonging, interdisciplinary team and individualized learning. McCord and Jeldes (2019) stated that a major area of concern in teaching first-year programming to non-majors relates to confidence. Estell et al. (2021) showed confidence in pharmacy students with positive societal impacts in their beginner programming courses. Khuri et al. (2020) studied the importance of belonging for students to continue in their field of study. Belonging correlated with student learning and interest to pursue more IS courses. Dodds et al. (2021) found Biology students taking a beginner programming course retained interest and identity in biology but were not at a disadvantage in, nor discouraged from later IS courses. Carr et al. (2020) and Salloum et al. (2021) emphasized the importance of creating an interdisciplinary faculty team. Mironova et al. (2016) validated individualization of the educational process in an e-environment increased motivation and learning. They considered the level of informatic students' prior knowledge and their preferences in the learning process.

Designing learning activities

Studies on designing learning activities provided course designers with choices. Chi (2009), Irani and Denaro (2020), Fabric et al. (2018), and Ouh and Gan (2021) identified 1) active learning activities to engage learner's attention such as discussions, Kahoots, clicker questions and supplementary mobile tutorials; 2) constructive learning activities that require learners to produce output such as self-reflection and problem solving; and 3) interactive learning activities involving dialogs such as group exercise, mentorship, and online social learning platform. Gan and Ouh (2019) found that learning activities for design thinking courses took a longer time to complete, which reduced available class time. Sivilotti and Pike (2007) suggested that exercises as learning activities for a distributed computing course should involve multiple dimensions of engagement. These learning activities were based on various pedagogy approaches: traditional lecture, active learning, flipped classroom, gamification or problem based. Kay et al. (2019) compared lecture, active learning, and flipped classroom with no significant differences among the approaches with respect to cognitive presence or learning performance. However, recent publications (below) show continued interest in flipped classroom approach for beginner programming courses.

Flipped classroom approach

Moore et al. (2021) described flipped classroom as consisting of pre-class independent, self-driven component. Often, requiring the students to watch lecture videos and to complete multiple-choice, auto-graded programming, or reflection questions. The traditional class lecture is typically replaced with active discussions and practical problem-solving sessions. Students can choose not to watch the videos and still learn via reading, active learning, problem-solving, and programming assignments.

Here are some benefits of flipped classroom approach for beginner programming courses. They are more flexible/self-directed/self-regulated (Çakıroğlu and Öztürk, 2017; Isomöttönen and Tirronen. 2016; Souza and Rodrigues, 2015), better availability of resources (Davenport, 2018), and increased in-class engagement/interaction (Kay et al., 2019; Zainuddin et al., 2019). Rosiene and Rosiene (2015) stated that flipping a class allowed for more interaction during in-class meetings. Learning definition and syntax were delegated to pre-class activities, while active and deeper learning took place in the classroom. Here are some drawbacks. Zainuddin et al. (2019) wrote the most significant challenge was a lack of students' motivation to watch the pre-recorded video lectures or to study the contents outside of the class time. Davenport (2018) found a small number of students felt the approach was ineffective and expressed a desire for a brief lecture, particularly for more advanced programming topics. AlJarrah et al. (2018) and Arakawa et al. (2021) faced procrastination. AlJarrah et al. (2018) discovered that students tend to access learning materials late, spiking on the lecture day. Mok (2014) suggested "forcing" students to be engaged in activities in class benefited students who would otherwise not have attempted the programming problems if they had been doled out as homework.

Flipped classroom with problem-based approach

Many flipped classroom approaches incorporated active problem-based in-class sessions. Chis et al. (2018) investigated the effectiveness of combined flipped classroom (FC) and problem-based learning (PBL) teaching. The results showed that FC-PBL approach was effective, and the knowledge acquired by students improved for the weaker students. Çakıroğlu and Öztürk (2017) studied problem-based learning activities in flipped classrooms to promote self-regulation. Self-regulated learning, the goal setting and planning, task strategies and help seeking skills of the students were high. They suggested using problem-based activities in flipped learning. The results from Chiang (2017) indicated that the flipped classroom combined with the problem-solving strategy was more effective than the previous problem-solving behavior sequence.

Some flipped classroom research focused on video watching patterns. Moore et al. (2021) found that students with prior programming experience watched fewer videos and that students that watched more videos performed slightly better on summative assessments. Angrave et al. (2020) found students who watched an above-threshold of video minutes improved their performance with the largest gain for the lowest-performing quartile. AlJarrah et al. (2018) found procrastination. Other flipped classroom research focused on the learning activities. Schwarzenberg et al. (2018) studied when or why flipped class learning activities have a positive effect on students. His study analyzed student experience for class activities, online participation, and out-of-class activities. The improvement in achievement increased when learning activities improved student experience. He suggested the design of a flipped class should consider the effect of different learning activities and select the most appropriate ones for a particular context.

Programming Learning Activities

Our beginner undergraduate Python course was designed based on the flipped classroom with problem-based approach. We categorize our learning activities into 1) non-video flipped classroom pre-class activities: pre-class exercises (PCE) and self-tests; 2) problem-based in-class exercises (ICE) and extra in-class exercises (EICE); 3) active in-class discussions, Kahoots and pop quizzes; and 4) post-class assessments: labs, laptop lab tests and paper finals. The following discusses each in detail.

Flipped PCE and Self-Tests

Our flipped classroom required students to work on problem based PCE and self-tests. Week one PCE was an Anaconda installation guide with exercises to run Python code using command prompt, a file, and Jupyter Notebook. The rest of the weeks involved 1) Python fundamentals: writing expressions, control structures, tuples/lists/dictionaries, functions, and files; and 2) basic programming skills: documentation, debugging, modularization, basic algorithm analysis and iteration. Each PCE came in a Jupyter Notebook which included the learning outcome (e.g., students should be able to define a Python function), code examples, syntax/semantics, and exercises. The self-tests were auto-graded multiple-choice questions (MCQ) using our learning management system (LMS). Typical questions involved understanding Python code. To motivate students to complete the PCE, students were asked to do a self-test before each class to earn participation points.

MCQ limits the type of learning assessment in our self-tests. A possible improvement is a Python environment with error message feedback (Kohn and Manaris, 2020) or a Jupyter Notebook auto-grader (Manzoor et al., 2020). Manzoor et al. (2020) stated 80% of students believed that getting immediate feedback improved their performance. It significantly reduced teacher workload but required the maintenance of an auto-grader system. Another issue with auto-grader is student reliance on the auto-grader to correct their mistakes. Baniassad et al. (2021) reduced student reliance on the auto-grader by imposing a regression penalty to encourage students to check their own solutions.

Problem-based ICE and EICE

For ICE, students were presented with problem-based learning where they worked on three to eight small problem exercises per week in class. Each problem was marked with one to three stars based on the level of difficulty. To address the inclusivity of a diverse non-IS student cohort and still present a challenge for students with good prior programming skills, the ICE was split to have an extra exercise set (EICE). While each student worked individually on the ICE, the teacher answered student questions and managed the progress of problems to be solved. Due to limited class time, the goal was to complete the ICE. However, the teacher answered questions from the EICE as they were raised during class. A possible suggestion from Mok (2014) was to do pair programming (instead of individual) as the predominant class active learning activity to increase student engagement.

In-class Discussions, Kahoots and Pop-Quizzes

The teacher began the 3-hour class with a quick summary review of the learning outcome from the problem-based PCE. This allows all students to recap the content covered in PCE, including for students who did not do their homework. The class continued with a review of the self-test questions that were incorrectly answered by most of the students and Kahoot questions. Kahoots were meant to be fun, so they were taken anonymously using nicknames (Ouh and Gan, 2021). As gamification, Kahoot winners were rewarded with participation points if they chose to identify themselves. The teacher gave pop quizzes using the LMS as part of class assessment (10%) and encouraged students to work on the PCE. The new pop quiz questions were like the self-test or Kahoot question format and content. These quizzes helped the teacher gauge the number of students fulfilling the learning outcome and identified weak areas that teacher needed to focus on during class. After the pop quiz, students worked on the ICE and EICE as the teacher walked around class to answer questions. Before the class ended, the teacher reviewed the ICE solutions. The review included active discussions of different ways to solve the same problem.

The gamification experience of self-test and Kahoot was like in Sprint and Fox (2020) and Rogers et al. (2021). Students in Sprint and Fox study made better study choices, such as submitting programming assignments and online quizzes significantly earlier; however, these improved study choices did not lead to higher final exam scores. In comparison, our students were challenged, increased engagement, and reduced procrastination during self-test and Kahoot learning activities, but some students were not interested. Since we only had two simple game elements, we did not auto customized/personalized the gamification like in Rogers et al. (2021).

Post-class Labs, Lab-Test and Finals

After class, a student may continue to work on the ICE and EICE. For those who completed the exercises or want additional practice, labs are available with two to six problems like ICE but with higher level of difficulties. Students seek help on labs during office hours and can submit their lab work to get additional participation points. To assess the student learning outcome, majority of the grades were determined by two laptop lab-tests (25% each) and a paper final exam (35%). The lab test involved four to six problems, completed using student laptop without internet connection (via LMS lockdown browser). Student can edit, compile, and run their code while being monitored by a teacher in class. They submitted their final solution in a fixed format (via LMS). For example, they were given a template code to begin which gets input from a file in a specific format and outputs another file in a specific format. The teacher will write a customized script to run black box unit test on student solutions. Auto grading reduced the effort but for some bad format submissions, the teacher must do white box grading. The final exam used pen and paper involving several MCQs and a problem set. The MCQ is easier to grade. However, the problem set required careful grading of the written solution. Pen and paper without access to their laptop reduced the chances for cheating and reliance on notes and tools. The lab tests and finals were scheduled outside regular class time to synchronize all students from different classes to take the same test.

Our lap-test experience is like Kurniawan et al. (2020) study on bring-your-own-device (BYOD) exam. Teachers have some confidence in preventing plagiarism and cheating, student like the familiar tools and auto grading lab test reduces teacher effort. However, a few students submitted their solutions in bad format, as described in Corley et al. (2020). For example, their solution missed an input resulting in misaligned output that is graded as wrong by the auto grading script. If partial credit is to be given, the teacher must do white box grading of the code manually. After the laptop lab test, it might be good to clarify the reason for pen and paper finals: to reduce student reliance on tools and bad format submissions. Some students wanted laptop finals. One student feedback "The pen and paper format for finals does not effectively mimic the actual working environment of professionals in the industry."

Research Method

Our school offered 15 classes of beginner Python courses during 2020-21. There was a total of 518 students, 84 (16%) were non-IS (Information Systems) students. These non-IS students took the course to fulfil their second major requirements in Data Analytics, Digital Business, etc. Other non-IS students may have taken the course as a professional literacy in this digital age, like a beginner writing skill. The focus of this in-depth study was with non-IS students.

During Spring 2021, we had a class consisting of 35 non-IS students. After the last day of class, we sent an e-mail survey using Google Form that had been approved by our institutional review board (IRB). Survey participation was voluntary. The survey questions are listed below.

- S1. How effective are these methods to keep your interest in the course topics? [1 Least to 5 Most] Please explain your least and most effective choices. [Free Text]
- S2. How effective are these methods for you to learn the course materials? [1 Least to 5 Most] Please explain your least and most effective choices. [Free Text]
- S3. Flipped learning is where students are introduced to the learning material prior to class and the time in class is used to deepen understanding. Flipped learning course is better than traditional course? [5 Strongly Agree to 1 Strongly Disagree]
- S4. Flipped classroom works for learning to code? [5 Strongly Agree to 1 Strongly Disagree] Please explain your answer. [Free Text]
- S5. What are your programming skills prior to this course? [4 Excellent (done programming before), 3 Good (aware of some programming concepts), 2 Fair (new to programming but confident) and 1 Poor (new to programming and not confident)]
- S6. What is your expectation of your course grade? [5 A, 4 B, 3 C, 2 D and 1 F]

S1 answered R1 on learning activities to keep student interest. S2 answered R2 to learn the course materials. S3 and S4 answered R3 on flipped classroom. S5 and S6 gathered student confidence levels through prior programming skills and grade expectations.

Research Findings

We collected a total of 28 valid survey responses from a random representation of the student population.

Student Interest

The answer to R1 "Which flipped classroom learning activities keep student interest?" is in figure 1. The preferred four learning activities were ICE with an average rating of 4.50, labs 4.46, PCE 4.04, and lab tests 4.04. The least preferred two learning activities were finals 3.50 and EICE 3.57.

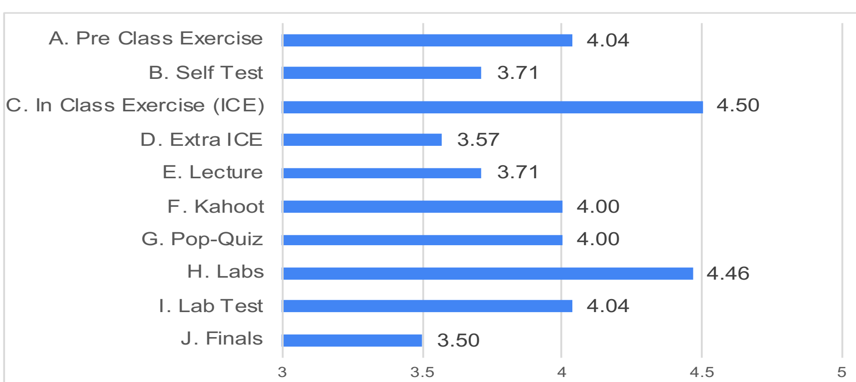

Figure 1. Which flipped classroom learning activities keep student interest?

Here are some comments.

- "Pre-class materials are helpful as we can come in class prepared."

- "ICE & Labs are where I get the most practice on how to code and thus helps me maintain my interest."
- "Lab tests help me to check on the areas which I need improvement on."

Student Learning

The answer to R2 "Which flipped classroom learning activities help students learn?" is in figure 2. *The preferred four learning activities were ICE, labs, PCE and lab tests. The least preferred two learning activities were EICE and finals. Same as R1.*

Here are some comments:

- "ICE and Labs are very useful to test if I have already digested the concepts as it is slightly more challenging than the pre-class exercise but not too challenging, unlike the extra ICE."
- "Finals won't really help in learning as we won't know if we are correct or wrong."

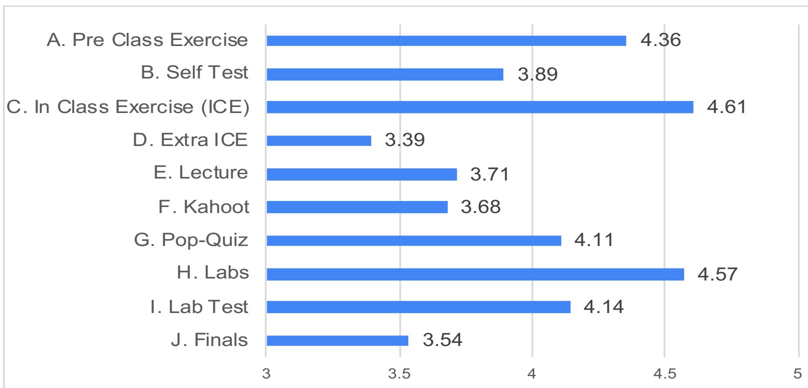

Figure 2. Which flipped classroom learning activities help students to learn?

Flipped classroom approach

The answer to R3 "What are student views on flipped classroom approach?" is in figures 3 and 4. 86% of students agreed or strongly agreed that flipped learning is better than traditional, answer to S3. 82% of students agreed or strongly agreed that flipped classroom works for learning to code, answer to S4. There was a high correlation of 0.84 between responses to S3 and S4 with a R2 of 0.70. This showed a strong positive linear relationship. *Students who agreed that flipped learning approach was better than traditional approach, agreed that it worked for learning to code as well.*

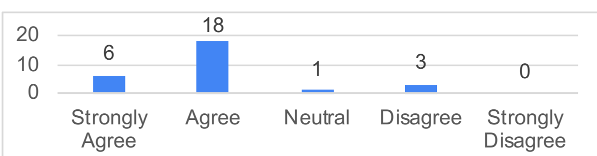

Figure 3. Flipped is better than traditional

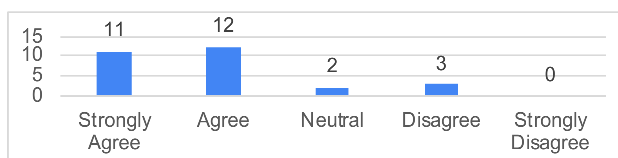

Figure 4. Flipped classroom works for coding

Here are some comments:

- "Coding is very hands-on, hence, flipped learning allows for more efficient use of class time."
- "It makes us prepare for the class and come with questions which we might need help with and therefore deepen our learning."

Here are comments from students who were neutral or disagreed that flipped classroom is better or works for coding.

- "Beginners to programming like myself might not be able to understand how to code by self-learning and would prefer a more direct and hands-on learning experience."

- "Time should be used to explain and go through the concepts in detail."

The comments are like those described in Davenport (2018). One suggestion for student was to present their solution to learn from one another. Studies (McCord and Jeldes, 2019; Rogers et al., 2021) suggested that these students needed more guidance or personalized support.

Programming skills

Responses to S5, shown in figure 5, grouped our students to excellent, good, fair, and poor prior programming skills. As the first programming course which can be exempted, excellent had a zero response as expected. We noticed two major groups of students: good (11) and poor (12). Student's prior skills suggested a confidence level corresponding to good and poor confidence levels respectively. There was a low correlation between prior skills and grade 0.40. However, the good group average grade of 4.36 was higher than the poor group average grade of 3.58. Thus, we investigated the statistical significance to test the alternate hypothesis: *The good group, student with good prior programming skills (confidence) on average, did better in grades than the poor group*. The grade histograms had slightly skewed distributions. Instead of a t-test, the ordinal grade value between independent subjects led us to use Mann-Whitney U test to calculate the p-value. The p-value for two-tail was 0.043, below the traditional significant level α of 5%, rejecting the null hypothesis and accepting the above alternate hypothesis.

Student Grades

Figure 6 showed student expected grade vs actual grade, answer to S6. Student's expected grade suggested confidence level. The correlation was close to a strong simple linear regression of 0.74. The ANOVA significance F value was 5E-06, showing a reliable statistically significant result, not by chance. The T-test paired two sample two tail p-value was 0.018 with a slightly skewed histogram. The Wilcoxon signed rank test two tail p-value was 0.025. Both were below the traditional significant level α of 5%, not by chance. Thus, *student expected grades (confidence level) were significantly correlated to actual grades*.

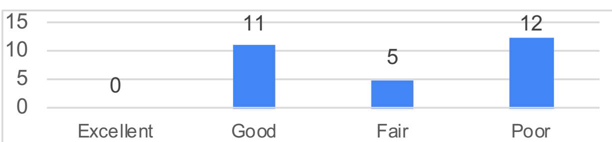

Figure 5. Student Prior Programming Skills

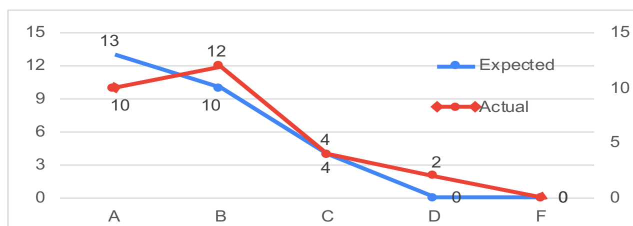

Figure 6. Expected vs Actual Grades

Research Analysis

We correlated all the S1: learning activities to keep student interest. For correlations >= 0.60, we calculated the p-value using Wilcoxon signed-rank test for statistically significant since the same student evaluated both Likert scale ordinal value (within subject).

- Flipped classroom: PCE and self-test correlation of 0.60. Wilcoxon p-value 0.042 < 5%, significant.
- Short quiz: Self-test and pop-quiz correlation of 0.60. Wilcoxon p-value 0.052 > 5%, not significant.
- In-class: Discussions and Kahoots correlation of 0.65. Wilcoxon p-value 0.094 > 5%, not significant.
- Large problem-based: EICE and labs correlation of 0.65. Wilcoxon p-value 0.0004 < 5%, significant.

The statistically significant suggest that correlation between both learning activities is not by chance. Thus, there could be a relationship between both learning activities from the student's survey. *The significant correlations suggested that teacher can choose both these learning activities together for students in a beginner programming course.*

The following are correlations of >= 0.60 for all S2: learning activities that helped students learn.

- Flipped was better S3 and ICE correlation of 0.70. Wilcoxon p-value 0.0001 < 5%, significant.
- Flipped worked for coding S4 and ICE correlation of 0.69. Wilcoxon p-value 0.0021 < 5%, significant.

These two correlations suggested that our active *ICE worked in relation with flipped classrooms for coding*.

The following are correlations of >= 0.60 for large problem-based activities with actual grades.

- EICE and grades correlation of 0.68. Normal distribution. Both T-test p-value 0.0003 and Wilcoxon p-value 0.001 are < 5%, significant.
- Labs and grades correlation of 0.60. Skewed distribution. Wilcoxon p-value 0.004 < 5%, significant.

These two correlations suggested that our active *large problem-based learning activities such as EICE and labs were significantly correlated with grades*. We noticed students who agreed or strongly agreed that they learn from EICE had an average grade of 4.5, much higher than students who disagreed or strongly disagreed 2.84. These students may be more motivated as EICE is optional. With a normal distribution, we used t-test paired two sample between the independent subject. The two-tail p-value was 0.002. Similarly for labs, average grade for the agreed group was 4.2 vs disagreed group was 3.0. Normal distribution t-test p-value was 5E-07. The data showed that *students who agreed with EICE and lab had better grades than those who disagreed, with EICE showing a bigger difference*.

Threat to Validity

Our research findings were based on student survey, feedback, grades, and teacher feedback for one beginner to programming class with all non-IS students. Our sample size (n=28) is within the range of 20 to 50 samples for each minor sub-group in a relational survey research as suggested by Delice (2010, p. 2008). This sample size (n=28) is a ratio of 33% of the sub-group population (28/84 non-IS students) which is sufficient representation. The relationships between learning activities are generalizable to other implementations because they are grounded in established pedagogy approaches. However, the specific context of our learning activities such as a Python course and the use of Kahoot and Jupyter Notebook may reduce the generalizability of our findings. While we acknowledge the threat to external validity in our study, we feel that sharing our experience is still research worthy, considering the emerging importance of beginner programming courses with non-IS in a flipped classroom course where the learning activities are designed for IS students. Please compare the context of your learning activities with ours.

Finding Insights

Flipped classroom worked for beginner programming course. Here are insights into our research findings.

- Students who agreed that flipped learning approach was better than traditional approach, agreed that it worked for learning to code as well. Majority (86%) agreed with flipped classroom. However, 14% students disagreed. Teacher considering doing flipped classroom with non-IS students should be aware of these 14% students who disagreed and cater for them.
- *The preferred four learning activities were pre, in-class exercises (P/ICE), labs, and laptop lab tests. The bottom two learning activities were finals and extra exercises (EICE). ICE worked in relation with flipped classrooms for coding*. Keep ICE.
- *The good group, non-IS student with good prior programming skills on average did better in grades than the poor group. Student expected grades were significantly correlated to actual grades*. Student's prior programming skills and expected grades suggested their confidence level which correlated with actual grades.
- *Large problem-based learning activities such as EICE and labs were significantly correlated with grades. Students who agreed with EICE and lab had better grades than those who disagreed, with EICE showing a bigger difference*. Teachers should avoid overwhelming students with extra exercises (making it optional) but may challenge students who want them and have confidence.
- *The significant correlations suggested that teacher can choose both these learning activities together for students in a beginner programming course*: Flipped classroom PCE with self-test; and large problem based EICE with labs.

Conclusion

This paper evaluated a set of learning activities for beginner programming course. By sharing our learning activity experience and quantitative analysis, we hope teachers can make a better-informed decision when choosing flipped classroom, problem-based learning activities for beginner programming course. Our research finding discussions provided insights on specific preferred learning activities and its correlation to grade performance. Our diverse student population agreed with flipped classroom and problem-based learning activities. This support similar study (Çakıroğlu and Öztürk, 2017; Chiang, 2017; Chis et al., 2018). Flipped pre-class exercise and active in class problem-based exercise combination were good. Students who agree on doing labs and extra exercise performed better. Confidence level was significantly correlated to grades. This support similar result in McCord and Jeldes (2019). Teacher decisions should be based on student prior programming skills and learning preferences, as suggested by Mironova et al. (2016). The personalized learning activities for non-IS students may be best implemented by an interdisciplinary team as suggested by the following studies (Carr et al., 2020; Dodds et al., 2021). This considers non-IS teacher perspectives.

One direction for future study is to analyze the right synergistic combination of flipped pre-class, active in-class, and problem-based exercises. The quantitative analysis can look at other independent variables besides confidence, prior skills, learning preferences, and other dependent variables besides survey, engagement, grades, and feedback. An alternate independent variable is the problem-based exercise could include more problems related to non-IS field of study such as Accounting. The survey dependent variable with student preferences may seem subjective, however, it is important to understand the views of non-IS students being taught using learning activities designed for IS students. More empirical data using learning activity specific objective variables such as quiz scores can be used to provide a stronger validation for causality to grade performance. Another direction is the teaching tools for each learning activity, such as auto-grader, gamification, laptop exam, real-time student feedback, and machine learning to predict weak students.

REFERENCES

AlJarrah, A., Thomas, M. K. and Shehab, M. 2018. "Investigating temporal access in a flipped classroom: procrastination persists," in *International Journal of Educational Technology in Higher Education* (15:1), pp. 1–18. https://doi.org/10.1186/s41239-017-0083-9

Angrave, L., Zhang, Z., Henricks, G. and Mahipal, C. 2020. "Who Benefits? Positive Learner Outcomes from Behavioral Analytics of Online Lecture Video Viewing Using ClassTranscribe," in *SIGCSE '20*, pp. 1193–1199. https://doi.org/10.1145/3328778.3366953

Arakawa, K., Hao, Q., Greer, T., Ding, L., Hundhausen, C. D. and Peterson, A. 2021. "In Situ Identification of Student Self-Regulated Learning Struggles in Programming Assignments," in *SIGCSE '21*, pp. 467–473. https://doi.org/10.1145/3408877.3432357

Baniassad, E., Zamprogno, L., Hall, B. and Holmes, R. 2021. "STOP THE (AUTOGRADER) INSANITY: Regression Penalties to Deter Autograder Overreliance," in *SIGCSE '21*, pp. 1062–1068.

Çakıroğlu, Ü. and Öztürk, M. 2017. "Flipped Classroom with Problem Based Activities: Exploring Self-regulated Learning in a Programming Language Course," in *Educational Technology & Society*, (20:1), pp. 337–349.

Carr, V., Jones, M. and Wei, B. 2020. "Interdisciplinary Computing: Applied Computing for Behavioral and Social Sciences," in *SIGCSE '20*, pp. 400–406. https://doi.org/10.1145/3328778.3366799

Chi., M. T. H. 2009. "Active-Constructive-Interactive: A Conceptual Framework for Differentiating Learning Activities," in *Topics in Cognitive Science*, (1:1), pp. 73–105.

Chiang., T. H. 2017. "Analysis of Learning Behavior in a Flipped Programming Classroom Adopting Problem-Solving Strategies," in *Interactive Learning Environments*, (25:2), Routledge, pp. 189–202.

Chis, A. E., Moldovan, A., Murphy, L., Pathak, P. and Muntean, C. H. 2018. "Investigating Flipped Classroom and Problem-based Learning in a Programming Module for Computing Conversion Course," in *Educational Technology & Society*, (21:4), pp. 232–247.

Corley, J., Stanescu, A., Baumstark, L. and Orsega, M. C. 2020. "Paper Or IDE? The Impact of Exam Format on Student Performance in a CS1 Course," in *SIGCSE '20*, pp. 706–712.

Davenport, C. E. 2018. "Evolution in Student Perceptions of a Flipped Classroom in a Computer Programming Course," in *Journal of College Science Teaching*, (47:4), pp. 30–35.

Delice, A. 2010. "The Sampling Issues in Quantitative Research," in *Educational Sciences: Theory and Practice*, (10:4), pp. 2001-2018, Aut. https://files.eric.ed.gov/fulltext/EJ919871.pdf

Dodds, Z., Morgan, M., Popowski, L., Coxe, H., Coxe, C., Zhou, K., Bush, E. and Libeskind-Hadas, R. 2021. "A Biology-based CS1: Results and Reflections, Ten Years," in *SIGCSE '21*, pp. 796–801.

Estell, J. K., Coffman-Wolph, S., Sieg, J. and Musser, M. 2021. "Supporting Pharmaceutical Healthcare Outreach: A Culminating First-Year Programming Experience," in *SIGCSE '21*, pp. 376–382.

Fabic, G. V. F., Mitrovic, A. and Neshatian, K. 2018. "Investigating the Effects of Learning Activities in a Mobile Python Tutor for Targeting Multiple Coding Skills," in *Research and Practice in Technology Enhanced Learning*, (13:23). https://doi.org/10.1186/s41039-018-0092-x

Gan, B. and Ouh, E. L. (2019). "Designing learning activities for experiential learning in a design thinking course," in *IEEE International Conference on Engineering, Technology and Education (TALE)*.

Irani, S. and Denaro, K. 2020. "Incorporating Active Learning Strategies and Instructor Presence into an Online Discrete Mathematics Class," in *SIGCSE '20*, pp. 1186–1192.

Isomöttönen, V. and Tirronen, V. 2016. "Flipping and Blending-An Action Research Project on Improving a Functional Programming Course," in *ACM Transactions on Computing Education*, (17:1), pp. 1–35.

Kay, R., MacDonald, T. and DiGiuseppe, M. 2019. "A Comparison of Lecture-Based, Active, and Flipped Classroom Teaching Approaches in Higher Education," in *Journal of Computing in Higher Education*, (31:3), pp. 449–71. https://doi.org/10.1007/s12528-018-9197-x

Khuri, N., Lee, W., Lehmkuhl-Dakhwe, K. V., VanHoven, M. and Khuri, S. 2020. "Interdisciplinary Minor in Bioinformatics: First Results and Outlook," in *SIGCSE '20*, pp. 407–412.

Kohn, T. and Manaris, B. 2020. "Tell Me What's Wrong: A Python IDE with Error Messages," in *SIGCSE '20*, pp. 1054–1060. https://doi.org/10.1145/3328778.3366920

Kurniawan, O., Lee, N. T. S. and Poskitt, C. M. 2020. "Securing Bring-Your-Own-Device (BYOD) Programming Exams," in *SIGCSE '20*, pp. 880–886. https://doi.org/10.1145/3328778.3366907

Manzoor, H., Naik, A., Shaffer, C. A., North, C. and Edwards, S. H. 2020. "Auto-Grading Jupyter Notebooks," in *SIGCSE '20*, pp. 1139–1144. https://doi.org/10.1145/3328778.3366947

Mironova, O., Amitan, I., Vendelin, J., Vilipõld, J. and Saar, M. 2016. "Maximizing and personalizing e-learning support for students with different backgrounds and preferences," *Interactive Technology and Smart Education*, (13:1), pp. 19-35. https://doi.org/10.1108/ITSE-09-2015-0025

Mok, H. N. 2014. "Teaching tip: The flipped classroom," in *Journal of Information Systems Education*, (25:1), pp. 7-11.

Moore, C., Battestilli, L. and Domínguez, I. X. 2021. "Finding Video-watching Behavior Patterns in a Flipped CS1 Course," in *SIGCSE '21*, pp. 768–774. https://doi.org/10.1145/3408877.3432359

McCord, R. and Jeldes, I. 2019. "Engaging non-majors in MATLAB programming through a flipped classroom approach," in *Computer Science Education*, (29:4), pp. 313–334.

Ouh, E. L. and Gan, B. (2021). "Effective Digital Learning Practices for IS Design Courses during COVID-19," in *AMCIS 2021*, Proceedings, 1.

Rogers, M., Yao, W., Luxton-Reilly, A., Leinonen, J., Lottridge, D. and Denny, P. 2021. "Exploring Personalization of Gamification in an Introductory Programming Course," in *SIGCSE '21*, pp. 1121–1127. https://doi.org/10.1145/3408877.3432402

Rosiene, C. P. and Rosiene, J. A. 2015. "Flipping a programming course: The good, the bad, and the ugly," in *IEEE Frontiers in Education Conference (FIE)*, pp. 1-3. https://doi.org/10.1109/FIE.2015.7344151

Salloum, M., Jeske, D., Ma, W., Papalexakis, V., Shelton, C., Tsotras, V. and Zhou, S. 2021. "Developing an Interdisciplinary Data Science Program," in *SIGCSE '21*, pp. 509–515.

Schwarzenberg, P., Navon, J., Nussbaum, M., Pérez-Sanagustín, M. and Caballero, D. 2018. "Learning experience assessment of flipped courses," in *Journal of Computing in Higher Education*, (30:2), pp. 237–258. https://doi.org/10.1007/s12528-017-9159-8

Sivilotti, P. A. G. and Pike, S. M. 2007. "A Collection of Kinesthetic Learning Activities for a Course on Distributed Computing: ACM SIGACT News Distributed Computing Column 26," in *ACM SIGACT News*, (38:2), pp. 56–74. https://doi.org/10.1145/1272729.1272741

Souza, M. J. D. and Rodrigues, P. 2015. "Investigating the effectiveness of the flipped classroom in an introductory programming course," in *The New Educational Review*, (40:2), pp. 129–139.

Sprint, G. and Fox, E. 2020. "Improving Student Study Choices in CS1 with Gamification and Flipped Classrooms," in *SIGCSE '20*, pp. 773–779. https://doi.org/10.1145/3328778.3366888

Zainuddin, Z., Haruna, H., Li, X., Zhang, Y. and Chu, S. K. W. 2019. "A Systematic Review of Flipped Classroom Empirical Evidence from Different Fields: What Are the Gaps and Future Trends?" in *On the Horizon*, (27:2), Emerald Publishing Limited, pp. 72–86.