ASEE
125 YEARS AT THE HEART OF ENGINEERING EDUCATION
2018 ASEE ANNUAL CONFERENCE & EXPOSITION
JUNE 24–27, 2018
SALT PALACE CONVENTION CENTER
SALT LAKE CITY, UT

Paper ID #21586

Impact of the Flipped Classroom on Students' Learning and Retention in Teaching Programming

Ms. Shamima Mithun, Indiana University-Purdue University, Indianapolis

Lecturer at Computer Information Technology (CIT) department, IUPUI I received my PhD in Compter Science in 2012.

Dr. Nancy Evans, Indiana University

Nancy Evans is a Lecturer in Communication, Professional, and Computer Skills at the IU Kelley School of Business, Bloomington. She brings her experience in the business world (B.S. in Accounting), Ph.D. in Educational Studies from Ball State University, former consulting as a career coach, and teaching experience in Computer Information Technology at IUPUI to her current teaching of a leadership course. At numerous conferences, Nancy has delivered presentations related to critical thinking, flipped classrooms, online learning, and students' perceptions of meaningfulness. She has been awarded the Frank E. Burley Distinguished Professor Award for service and two Outstanding Teaching Awards. She recently completed a Mosaic fellowship at IUPUI and plans to continue as a Senior Mosaic Fellow at IU Bloomington.

Impact of the Flipped Classroom on Students' Learning and Retention in Teaching Programming

Abstract:

This "evidence-based practice" paper describes the study of a flipped classroom teaching approach compared to a traditional lecture-based teaching approach in an introductory programming course. Many students in the computer information technology (CIT) program at Indiana University–Purdue University Indianapolis (IUPUI) were not retaining fundamental programming concepts from their introductory programming course. Students also procrastinated taking higher-level programming courses because of unfavorable experiences in the introductory course. With the goal of addressing these issues and improving student learning, faculty employed a flipped classroom model to teach the introductory programming course. This flipped model reduced existing content coverage and focused on problem solving techniques, program design/development, logic, and object-oriented terminology/concepts. Evaluation results indicated the class grade average in flipped sections increased compared to traditional sections. Evaluation results also exposed that the flipped classroom has a positive impact on students' motivation and engagement.

Introduction:

Faculty members in the computer information technology (CIT) program at Indiana University–Purdue University Indianapolis (IUPUI) observed that many CIT students in 200-level programming courses were not retaining basic programming skills such as problem solving, program design, logic, and object-oriented concepts from their introductory (100-level) programming course, CIT 14000. As a result, in 200-level programming classes, faculty members spent lots of time reviewing fundamental programming concepts that had already been taught in the introductory course. Another observation is that students often procrastinated taking their higher-level programming courses because of unfavorable experiences in the introductory course. Based on the above observations, the goals of this project were to: (1) improve students' performance, (2) help students retain their programming knowledge/skills, (3) motivate students in learning programming, (4) improve classroom engagement, and (5) give students a better programming experience in the introductory course so that they will not defer enrolling in 200-level programming classes.

Research indicates that active learning environments such as those in a flipped classroom model have had a positive impact on students' learning, engagement, performance, and retention ([1]; [4]; [9]; [12]; [13]). The flipped classroom model also demonstrated higher teacher satisfaction compared to the traditional model [17]. Students often considered it more effective and engaging than the traditional lecture-based approach [3, 11]. In the flipped classroom model, lectures can be replaced by online videos using various screen capture technologies allowing students to use class time for active learning where there is repetition of problem solving, conceptual understanding, and application of what they are learning by practicing with other classmates through group activities and collaboration [2, 7]. Thus, a flipped classroom encourages social interaction among students, which is also a useful skill for students' future careers [15]. In this model, students also receive guidance from instructors as well as fellow students. This pedagogy increases student engagement and facilitates an exchange of ideas with fellow students. In summary, this model provides opportunity for depth without sacrificing content [3].

Motivated by the success of flipped classroom models, a faculty at our CIT program provided an active learning environment based on the flipped classroom model that allowed students to learn programming by practicing with other classmates during class while also having the instructor available for help in the daily in-class processes. As a result, classroom time was utilized for programming activities instead of traditional theoretical lectures. Lectures were moved outside of the classroom through video lectures to be viewed prior to class attendance with a short recap of the video lectures provided in class before beginning the programming activities. Furthermore, students were also quizzed on the video lectures to promote watching the video lectures and for a baseline assessment of their understanding of content prior to class.

In this project, the faculty member taught two sections of the introductory programming course (CIT 14000) in the Fall 2015 semester using a traditional lecture-based teaching approach and two sections of the same course in the Spring 2016 using the flipped classroom model. We compared flipped sections to traditional lecture-based sections. Several focus groups were also conducted by a staff facilitator from the Center for Teaching and Learning at our university. Students who take this introductory programming course are computer information technology (CIT) majors. CIT programming courses are similar to computer science programming courses, but the CIT major is in the field of computer and information technology, not computer science. The main distinction between CIT and computer science is the applied nature of CIT to solving real-world business problems by studying and using programming, databases, networks, and security versus the more theoretical aspect of computer science. The majority (approximately 85%) of CIT students are off campus residents, traditional-age, white male students. However, the number of residential students is on the increase and CIT attracts returning adult learners often looking for a new career. Students typically take CIT 14000 in their freshman or sophomore year. Ideally all students would take CIT 14000 in their freshman year, but sometimes students do not have the math prerequisites (pre-calculus) to take the introductory programming course. In both the Fall 2015 (traditional lecture) and Spring 2016 (flipped classroom) semesters, the content coverage included problem solving techniques, program design and development, logic, and basic object-oriented concepts. The only difference (change) in the Spring 2016 flipped sections was the instruction model. We did not account for differences among students in our study; however, the student demographics of the courses from the fall to subsequent spring semesters are quite similar.

Related Work:

The flipped classroom model is increasingly being utilized in higher education but the use of the flipped classroom has received less attention in STEM (science, technology, engineering, and mathematics) [1]. Historically, due to difficult subject matter and the technical nature in STEM disciplines, educators mostly rely on traditional lecture-based teaching methods [6]. In lecture-based models, educators spend class time explaining fundamental technical concepts with the use of examples, walking through the solutions to various problems, and then they are assigning homework to supplement this learning. In this model, the practice and application happen outside of the classroom in the form of homework with delayed feedback from the faculty. This traditional model does not permit sufficient time for student-student or student-faculty interactions during class. It has been observed by STEM educators that a lecture-based approach does not engage some students or keep them interested [6]. Researchers found it is even harder to keep students interested by competing with the invasion of social networks and other smartphone notifications for a longer class with traditional teaching [11]. Students also have difficulty retaining and applying their knowledge. To address these issues, educators have begun utilizing a flipped model in STEM education ([3]; [7]; [11]; [13]; [14]).

In a flipped classroom, students gain technical knowledge through online videos that prepare them to participate in class activities. These videos contain explanation of concepts, examples, and problem walk throughs. In-class activities are designed to answer questions or uncover common misconceptions, discuss complex topics, and work with students [7, 15]. Class activities also contain tasks which help students to learn and practice skillsets for mastery of concepts. In this learning environment, students get immediate real-time feedback from other students and faculty during class. Educators notice that students learn the concepts better when applied directly to a specific problem, and in the group learning environment, they learn to teach each other, which is the highest level of learning comprehension [7, 14].

However, research on flipped classrooms shows that positive outcomes from flipped learning models depend on its design [1, 8]. Instructional design plays a very important role in student experience and performance. Researchers [6] found that educators have “flipped” incorrectly (for example, by having students watch three hours of video lecture outside of class), and have been really frustrated with the flipped classroom concept. As a result, a key focus of current flipped classroom research is what factors educators need to consider when designing a flipped learning environment [8]. Presently, to design a successful flipped classroom, educators include various factors such as a pre-recorded video lecture with pre-class activity, a follow-up quiz on the pre-class work, an in-class activity, and an in-class group activity. Studies [1, 15] show that the following factors have significant impact on the success of a flipped classroom model: 1) out-of-class and in-class elements must be carefully integrated for students to understand the model and be motivated to prepare for class; 2) shorter, rather than longer videos; and 3) pre-class activities must be coupled with quizzes or follow-up tasks to assess the student's understanding. A successful flipped classroom design requires significant effort and time on the part of faculty [15]. Currently, educators generally evaluate the effectiveness of the whole flipped classroom learning environment rather than the individual components of a flipped classroom [14, 16]. To evaluate the effectiveness of the whole flipped classroom learning environment, educators typically conduct surveys to collect students’ perceptions about experiences or compare students’ performance in flipped classroom with those in traditional classroom. On many occasions, it has been found that students performed better in flipped classroom models compared to the traditional lecture-based models. For example, Balaban, Gilleskie, and Tran found that in a large-lecture principles of economics course the flipped classroom format increased students' final exam performance relative to the traditional instructional format [14].

In STEM education, the flipped classroom model has been found to be used mostly used to teach pure science and mathematics courses. This model has rarely been used to teach subjects from the applied science, technology, and engineering fields [5]. In a recent study of fifty-eight peer-reviewed research studies on flipped learning in the higher education STEM disciplines, results indicate that only about 6% of these studies were conducted in the applied science, technology, and engineering domains [5]. There are a few works where a flipped model has been used successfully in the applied science, technology, and engineering disciplines ([3]; [7]; [10]; [11]; [13]; [10]; [16]). Similar to the non-STEM domains, existing studies on technology and engineering reported high student satisfaction and increased performance [1]. For example, Liu found that the flipped section yielded much more A's, less B's, and less C's compared to lecture-based section in teaching Engineering Computation [16]. Redekopp and Ragusa also flipped a course, a computer organization and architecture course, and found that the flipped approach increased students' problem solving and modeling skills in computer engineering [10]. Paez used a flipped classroom model to teach a software engineering course. Students considered it more effective and engaging than the traditional lecture-based approach [11]. Cupak and Riabov utilized this model successfully to teach different computer science topics such as comparative analyses of algorithm implementations in different languages, object-oriented designing [7]. Davenport taught a computer programming course using this model [3]. The availability of resources, including online videos and classroom collaboration, was specified as a key component of students being able to succeed and many students felt that the flipped classroom improved their critical thinking skills. Some students indicated that they preferred attending lectures face-to-face, however, they also preferred using class time for problem solving and hands on activities. Our work is most comparable to this work; however, they used this approach to teach a different programming language called python and since their student population was very small, they could not evaluate the flipped classroom model effect on students' learning or retention. Bachnak and Maldonado effectively used this model to teach the principles of electrical engineering course and found that supplemental materials such as videos, illustrative problem solving, and application examples should be made available for student success [13].

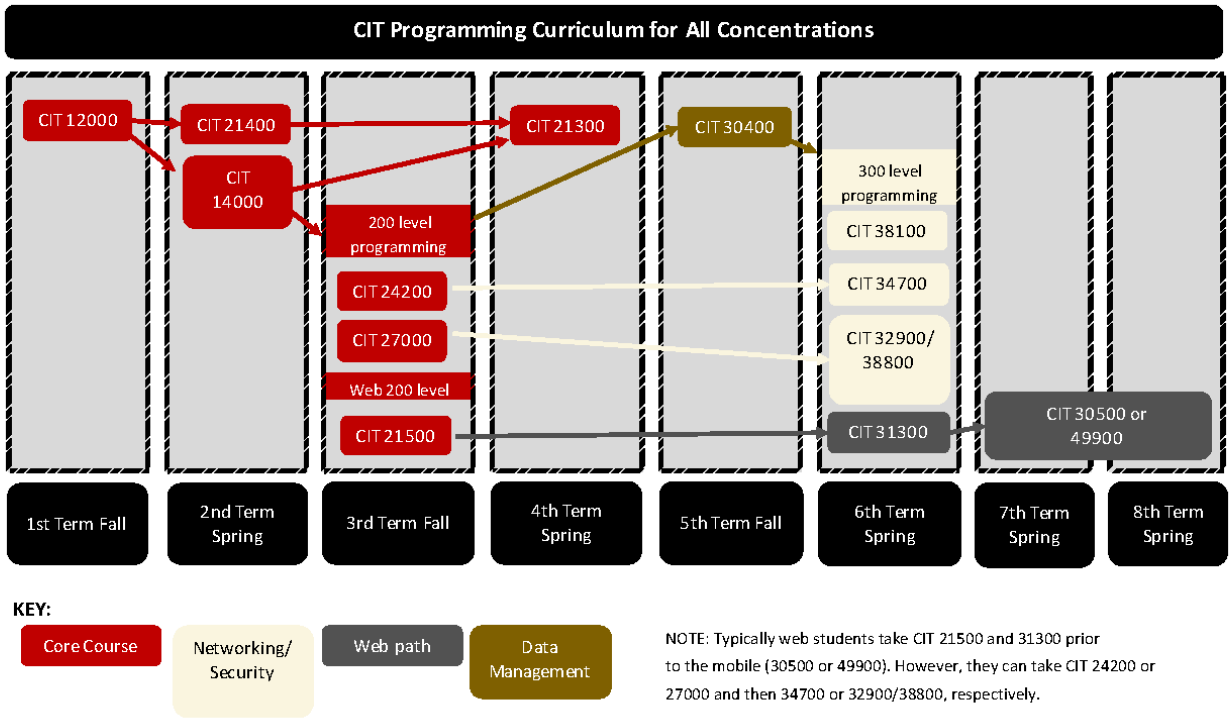
In our project, the flipped classroom model to teach introductory programming supports a future research direction in the technology domain of STEM education, specifically computer information technology. All students in our CIT program must take programming courses similar to those in the computer science field, so this project potentially applies to a broad audience of introductory programming courses as well as higher level programming courses where it has been determined that class interaction with students and faculty would be beneficial to students. We have found that first and second-year students find learning programming to be a challenging task. Students find it is difficult to learn technical information from textbooks and traditional classroom lectures. We also know that beginners need to write many programs where they must use newly-learned programming concepts and programming constructs. To address these needs, we entered the emerging research in the STEM technology domain by applying a flipped classroom model to teach introductory programming.

Flipped Classroom Model for Introductory CIT Programming Course:

CIT 14000 is the introductory programming course that all students in our computer information technology program are required to take. This course is an introduction to problem-solving techniques, program design and development, programming logic, and object-oriented terminology and concepts. This course is the prerequisite for all 200-level programming courses as shown in Figure 1. This course is often the first programming course students have ever taken. While many of the students have taken a pre-programming course (CIT 12000) that focuses on fundamental programming concepts using flowcharts and pseudocode (a structured, English-like language used to represent steps in an algorithm), CIT 12000 did not seem to be helping students in the CIT 14000 introductory programming course.

The traditional lecture-based structure and instructional pedagogies of CIT 14000 had not changed in over a decade. Previously, the course covered too much content without enough practice of the basic concepts. First-year students in our CIT program find programming to be a particularly challenging task and need multiple attempts with similar problems for which they must use newly learned programming constructs. They also need significant assistance to successfully write programs. Lecture-based teaching cannot allocate sufficient time for students to practice newly learned concepts in class.

Figure 1: Typical flow of programming curriculum



NOTE: Typically web students take CIT 21500 and 31300 prior to the mobile (30500 or 49900). However, they can take CIT 24200 or 27000 and then 34700 or 32900/38800, respectively.

As an intervention, the CIT 14000 instructor flipped the classroom beginning with Spring 2016 sections. For the flipped classroom model, the faculty member changed the focus of the course content from breadth to depth of coverage. Outside of and prior to class, students needed to watch video lectures recorded by the instructor, read the chapter, and do a quiz as prep work. This low stakes quiz provided indication on students' understanding of topics, and their performance on the quizzes suggested that they do indeed watch the video lectures outside of class. Outside of class, students often also require to attempt a problem where they had to write an algorithm or pseudocode and then bring that to class for feedback from the instructor and students. It was helpful because it provided them with an idea of what they understood and did not understand immediately on a low stakes assignment. From that point, it was easier for them to figure out what they needed to do to fix their mistakes. In class, the instructor could see students discussing with each other what they did not understand in the pre-class activity which aided in self-directed learning.

In class, there was also a brief review from the instructor and then there were in-class activities. The unique aspect of this flipped model is that for the in-class activities students had a programming partner to consult, which we called peer programming. There were many instances in class sessions where students worked individually to develop pseudocode and could consult their partner for help. In more complex problem situations, students worked together in a group of up to 5 students to develop pseudocode. Individually, students implemented the pseudocode (applied JavaScript syntax to pseudocode), ran their programs individually and if they encountered errors, they could consult their peer programming partner for debugging. Toward the end of class, the instructor asked for a volunteer to show/type up code so the students who did not get their program to work could see a working solution before going on to do individual homework that was assigned out of class.

Results and Discussion:

In this project, a faculty member taught two sections of CIT 14000 in the Fall 2015 using a traditional lecture-based teaching approach and two sections of the same course in the Spring 2016 using the flipped classroom model. In this experiment, the same assignments (see Appendix), including exams which covered programming skills and conceptual understanding of programming constructs, were used for all sections and all the tasks were graded by the same faculty member using the same rubrics (see Appendix). Students' performance data from the Fall 2015 sections were compared to Spring 2016 sections. A total of 86 students' data was analyzed (N = 45 for Fall 2015 traditional classroom; N = 41 for Spring 2016 flipped classroom). Results showed the class grade average in flipped sections increased from a solid C in traditional sections to a high C+ in the flipped sections. There were no students in the Spring 2016 sections who were repeating the course from the Fall 2015 semester, and the students in the Spring 2016 sections would not have had more course preparation than those who took the course in the Fall 2015 sections.

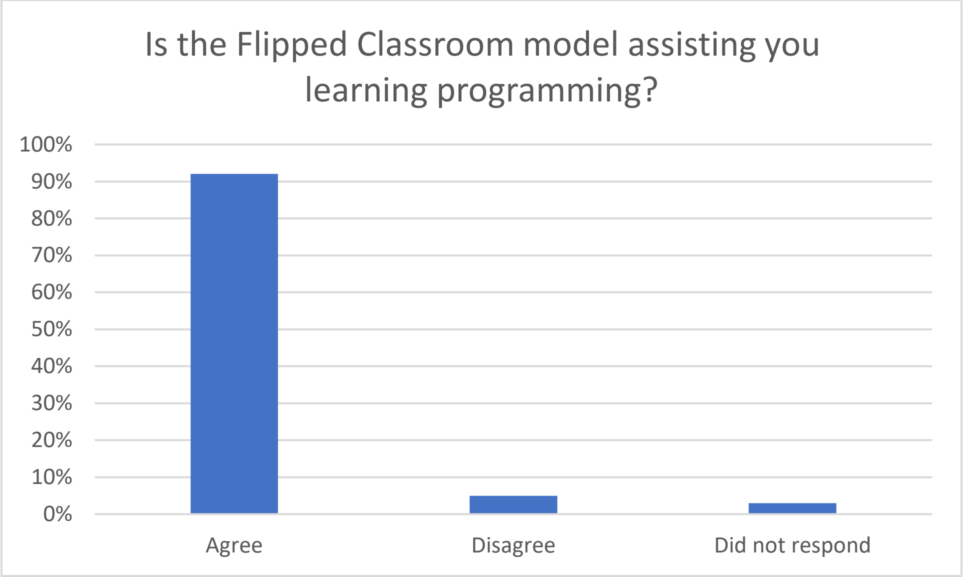
An independent-samples t-test was conducted to compare students' final grades of the Fall 2015 traditional classroom (M = 74.38, SD = 19.32) and students' final grades of the Spring 2016 flipped classroom (M = 79.36, SD = 17.97). There was not a statistically significant difference in the final grades, t(84) = -1.06, p = .29. These results suggest that the flipped classroom does not have an effect on students' final grades; however, we find the increase from C to C+, almost B-) to be a vast improvement. The increase in grades could be explained by students being more acclimated to school in a spring versus fall semester, or the instructor could be more used to the new content coverage and course materials which could have a positive effect on the final grades. Regardless of the nature of the statistically non-significant increase in grades, there are other indicators that suggest the flipped classroom is improving student learning and meeting our project goals.

Despite the flipped classroom model not showing statistically significant improvement on students’ performance, students' attendance in flipped classroom increased by 9% compared to that of traditional classroom. The flipped classroom was the only structural class variable that changed regarding attendance, meaning that the attendance policy did not change from 2015 to 2016. Not only did attendance increase, but the faculty member observed that student participation improved. Students’ attendance and participation suggest that the flipped classroom has a positive impact on students’ motivation and engagement toward the class and its content. The rating on course evaluations for overall course satisfaction for the Spring 2016 sections also improved from 3.16 (Fall 2015) to 3.60 (Spring 2016) on a 4-point Likert scale which is an increase of 12% from the traditional classroom to the flipped classroom. This improvement in overall course satisfaction indicates students perceive they are getting a better learning experience.

Student comments also suggest that the learning experience was better from the Fall 2015 traditional class to the Spring 2016 semester flipped class. A student comment from the Fall 2015 traditional class stated that the “instructor just read the PowerPoint slides and didn't try any explaining methods. That class was supposed to be my intro to programming, but it moved too fast." Contrast that with a student comment from Spring 2016 flipped class that stated, “watching the lectures prior to class was very helpful, as it made the in-class lecture more of a review. In-class assignments were invaluable in gaining hands on experience."

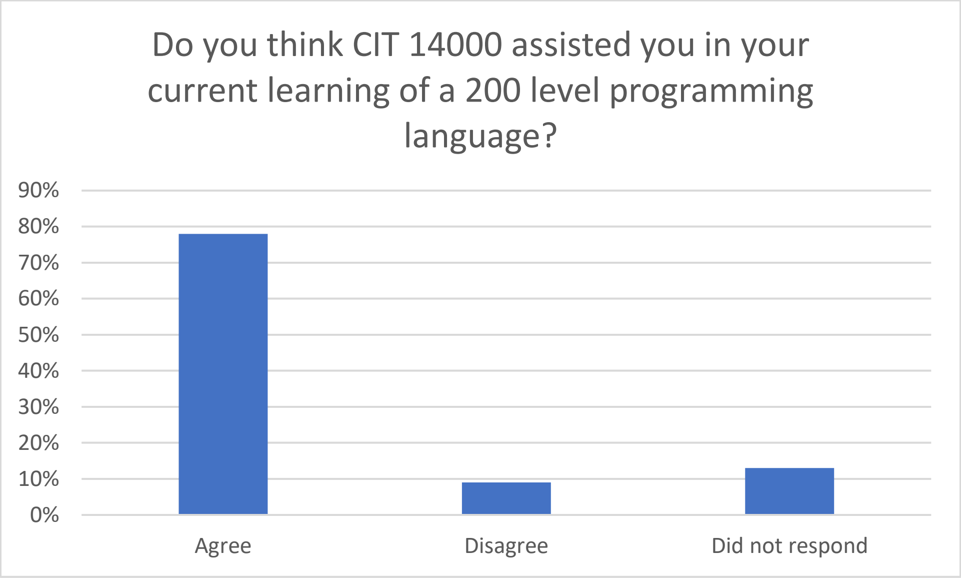
In addition to looking at course evaluations, several focus groups were conducted in the Fall 2016 by a staff facilitator from the Center for Teaching and Learning from our institution. In these focus groups, fifty students from two separate sections of CIT 14000 sections taught using the flipped classroom participated. The reports illustrate that 92% of students perceived their understanding was deepened by the "flipped" approach and it helped them learn programming. About 5% students disagree with the flipped approach and 3% students did not respond (shown in Figure 2). Students who thought the flipped classroom deepened their understanding also mentioned that the flipped classroom allowed more time to grasp and understand concepts. On the other hand, students who did not like the flipped classroom stated that they preferred the traditional lecture provided by the instructor, and some of them mentioned they did not get sufficient time to work through the activities. Since the vast majority of students perceive that their understanding and learning is deeper, they should be able to retain the programming knowledge better. Focus group results also suggested that 88% of CIT 14000 students are "happy" with the "new" instructional methods and course resources.

Figure 2: Students' perceptions regarding flipped classroom



To further evaluate students' retention, we conducted a survey towards the end of the Fall 2016 semester to collect 200-level programming students' opinions with students who had taken the flipped introductory course in the Spring 2016. In this survey, 38 students from two different classes participated (see Figure 3). Figure 3 displays that in this survey, 78% of students agreed that CIT 14000 prepared them well for their 200-level programming class; 13% did not respond because they did not take the class at our institution; and 9% disagreed about CIT 14000 preparing them for their 200-level programming course. Students who disagreed admitted that either they did not spend sufficient time in CIT 14000 or they took the class too long ago.

Figure 3: Students' perceptions regarding their preparedness for 200-level programming course



We have also received positive verbal feedback on students' retention from 200-level faculty members. For example, a senior faculty member teaching 200-level programming in our program stated, "In my 9 years of teaching here for the first time I am noticing students are correlating concepts from their 100-level programming course to their 200-level programming topics." Further, students are no longer procrastinating enrolling in their 200-level programming courses. Students are enrolling in their 200-level programming classes the semester after completing their introductory programming course rather than putting off their 200-level programming. Following the advised plan of study by taking the 200-level class immediately after the 100-level helps students with flow in their studies and reduces the need for remediation later on. In conclusion, both the quantitative results from the focus groups and qualitative results from 200-level faculty suggest that the flipped classroom has had a positive impact on students' learning and retention in 100 and 200-level programming classes.

Conclusion and Future Work:

In this project, we used a flipped classroom model to teach introductory programming. As part of the flipped classroom, we used pre-recorded video lecture as pre-class activity, follow-up quizzes, in-class activities, and group activity and discussion. We found that follow-up quizzes or discussion is required to encourage students to complete their assigned pre-class activity.

Evaluation results show that flipped classroom model improves students' performance, motivation, and engagement. Most of the students perceived their understanding was deepened by the "flipped" approach. Students also agreed that CIT 14000 prepared them well for their 200-level programming class. In conclusion, we found that the changes we have made to the introductory programming course are leading to improvements in learning for our students.

In the future, we would like to study if student differences account for differences in grades, course evaluation ratings (if possible), and/or student perceptions related to the surveys and focus group results. We also would like to conduct research on the 200-level programming classes to evaluate the impact of flipped classroom model on students' actual versus perceived retention of programming concepts from their 100-level course. Further, we will be revising the pre-programming course (CIT 12000) as a result of this project because it does not appear to be helping students in their introductory programming course, CIT 14000. Finally, because the flipped classroom worked well in introductory programming, we will be looking at its utility in other CIT courses.

Acknowledgement:

This project was funded by Curriculum Enhancement Grant (CEG) 2016 received from Center for Teaching and Learning (CTL), IUPUI.

Bibliography:

[1] B. Kerr, "The flipped classroom in Engineering Education: A survey of the research", *2015 International Conference on Interactive Collaborative Learning (ICL)*, Florence, Italy, 2015.

[2] B. Simon, "Why the flip should I flip my classroom: Results on the use of peer instruction in computing courses," *Journal of Computing Sciences in Colleges*, 28(4), 92, 2013.

[3] C. E. Davenport, "Evolution in Student Perceptions of a Flipped Classroom in a Computer Programming Course", *Journal of College Science Teaching,* 2018.

[4] D. Berrett, "How "flipping" the classroom can improve the traditional lecture," *The Chronicle of Higher Education*, 2012. Retrieved from http://chronicle.com/article/How-Flipping-theClassroom/130857/, Last Accessed: February 04, 2018.

[5] E. Huber and A. Werner, "A review of the literature on flipping the STEM classroom: Preliminary findings," *In S. Barker, S. Dawson, A. Pardo, and C. Colvin (Eds.), Show Me The Learning, Proceedings ASCILITE 2016 Adelaide*, 2016. pp. 267-274.

[6] “Flipped classrooms turning STEM education upside down.” Retrieved from http://news.psu.edu/story/413452/2016/06/06/academics/flipped-classrooms-turning-stem-education-upside-down Last Accessed: February 04, 2018.

[7] J. J. Cupak and V. V. Riabov, “APPLYING "FLIPPED CLASSROOM" METHODOLOGY IN COMPUTER SCIENCE COURSES”, *Rivier Academic Journal* , Vol. 13 Issue 2, pp. 1-6. 2017.

[8] L. Liu, D. Ripley and A. Lee, A., “Flipped Learning and Influential Factors: Case Analysis.” *Journal of Educational Technology Development and Exchange (JETDE):* Vol. 9: Iss. 2, Article 5, 2016.

[9] M. Estes, R. Ingram, and J. C. Liu, “A review of flipped classroom research, practice, and technologies,” *International Higher Education Teaching & Learning Review*, July 29, 2014.

[10] M.W. Redekopp and G. Ragusa, “Evaluating Flipped Classroom Strategies and Tools for Computer Engineering,” *Proceedings of the 120th American Society of Engineering Education Annual Conference & Exposition*, 2013.

[11] N. M. Paez, “A Flipped Classroom Experience Teaching Software Engineering”, *2017 IEEE/ACM 1st International Workshop on Software Engineering Curricula for Millennials (SECM),* Buenos Aires, Argentina, pp. 16-20, May 20 - 28, 2017.

[12] N. Sarawagi, “Flipping an introductory programming course: Yes, you can!” *Journal of Computing Sciences in Colleges*, 28(6), pp. 186-188, 2013.

[13] R. Bachnak and S. C. Maldonado, “A Flipped Classroom Experience: Approach and Lessons Learned”, *proceedings of the 121st ASEE Annual Conference & Exposition*, Indianapolis, IN, June 15-18, 2014.

[14] R. A. Balaban, D. B. Gilleskie and U. Tran, “A quantitative evaluation of the flipped classroom in a large lecture principles of economics course,” *The Journal of Economic Education*, 47 (4), pp. 269–287, 2016.

[15] S. Elazab and M. Alazab, “The Effectiveness of the Flipped Classroom in Higher Education”, *2015 Fifth International Conference on e-Learning,* Manama, Bahrain, 18-20 October, 2015.

[16] S. Liu, **“**Teaching Engineering Computation Using a Flipped Classroom Model,” *Proceedings of the 2015 ASEE North Central Section Conference,* 2015.

[17] Z. Unal and A. Unal, “Comparison of Student Performance, Student Perception, and Teacher Satisfaction with Traditional versus Flipped Classroom Models”, *International Journal of Instruction,* Vol.10, No.4, pp. 145-164, 2017.

Appendix

Sample Class Activity:

Write a program that randomly generates an integer greater than or equal to 0 and less than 100. The program then prompts the user to guess the number. If the user guesses the number correctly, the program outputs an appropriate message. Otherwise, the program checks whether the guessed number is less than the random number. If the guessed number is less than the random the number generated by the program, the program outputs the message, ‘‘Your guess is lower than the number’’; otherwise, the program outputs the message, ‘‘Your guess is higher than the number’’. The program then prompts the user to enter another number. The user is prompted to guess the random number until the user enters the correct number.

Sample Rubric:

Criteria	**Rating**			**Points**
Variables Declared correctly	3.0 pts Full marks	2.0 pts Missed 1-2	0.0 pts Missed more than 2	3.0 pts
Correct Usage of Decision Statement	10.0 pts Full marks	5.0 pts Missed 1	0.0 pts Missed 2 or more	10.0 pts
Correct Usage of Functions	5.0 pts Full marks	3.0 pts Missed 1	0.0 pts Missed 2 or more	5.0 pts
Correct Usage of Arguments	5.0 pts Full marks	3.0 pts Missed/wrong 1	1.0 pts Missed 2 or more	5.0 pts
Correct Usage of Parameters	5.0 pts Full marks	3.0 pts Missed/wrong 1	2.0 pts Missed 2 or more	5.0 pts
Error Checking Performed	3.0 pts Full marks	0.0 pts If not done		2.0 pts