Improving Student Learning in an Introductory Programming Course Using Flipped Classroom and Competency Framework

Joelle Elmaleh
School of Information Systems
Singapore Management University
Singapore
joellee@smu.edu.sg

Venky Shankararaman
School of Information Systems
Singapore Management University
Singapore
venky@smu.edu.sg

***Abstract*— In this paper we report our study on the impact of implementing a flipped classroom model on student learning in an undergraduate introductory programming course. We use three components to measure student learning namely, final exam scores, competency acquisition and feedback levels. We compare a traditional offering with a flipped offering delivered the following year to a comparable student population, and with no change in the course content and assessments. We observed that in comparison to the traditional, the flipped model increased pass rates in the final exam and also enhanced competency acquisition. In terms of feedback levels, the flipped classroom provided more time for one-to-one in-class personalized feedback. Our approach is unique in the sense that we use a competency framework to be able to pin point the competencies that were improved as a result of the flipped model.**

Keywords— Introductory programming course; student learning; flipped classroom; student competencies; competency framework

I. Introduction

Flipped classroom is a form of blended learning where a student is first exposed to preparatory material outside of class, usually through online videos and practice exercises. When the student attends class in a traditional face-to-face setting, the class time is used to apply the material through problem-solving, feedback and discussion sessions. In this paper we report on a study undertaken to estimate the efficacy of flipped classroom to support acquisition of programming competencies in an introductory programming course.

Students doing the BSc Information Systems (IS) Program are required to do the first programming course in Term 1 of Year 1, titled Information Systems Software Foundation (ISSF). This course introduces students to building blocks of programming concepts such as object manipulation, repetition, decisions, etc. In the past, this course has been delivered through a traditional face-to-face live classroom sessions. Students come to class without any preparation and during the face-to-face sessions, programming concepts are covered through lectures, and students are given a few exercises to solve in class, followed by homework exercises. The teaching team observed that many students failed to complete the exercises given in class due to lack of time, and hence became less motivated to complete them on their own at home without instructor support. Over the weeks, unworked class exercises and home exercises piled up leading to further disillusion and eventually leading to failure in the class quizzes and final exam. After conducting a detailed review and through many discussions with the students, the teaching team came to the conclusion that students needed “more time at task” in class. In 2014, the teaching team decided to experiment with flipped classroom. In this model, each week, prior to attending the class session, the students were required to watch video tutorials that explained the programming concepts and then work on self-help quizzes. In the class session, the students were given problems related to the corresponding concepts covered in the video, and while actively working on the problems, the instructors and teaching assistants provided real-time feedback. Following the class session, the students were given additional homework exercise for further practicing the concepts. Over the weeks, the teaching team observed that most students were completing both the class and home exercises on time and in general seemed to have a better grasp of what they were learning. In comparing the student performance in 2013, when the traditional face-to-face model was used, to that in 2014, there was substantial improvement in both the competencies acquired by the students and the pass rates in the final exam. A key takeaway from our study is that, flipped classroom model does enhance student learning in introductory programming course, where a lot of the learning happens when the students solve the programming exercises with real-time feedback from instructors and teaching assistants.

The paper is structured as follows: In Section 2 we review other related work in the areas of competency based learning and flipped classroom. In Section 3 we describe in detail the traditional and flipped approach with respect to the ISSF course. In Section 4 we analyse and compare the results of both these approaches. Section 5 summarizes the conclusions from our work.

II. RELATED WORK

We review three areas of work that are related to the current research, namely competency based learning and assessment, learning outcomes and competency framework, and flipped classroom learning.

A. Competency Based Learning and Assessment

Many higher education institutions have clearly defined learning outcomes for the program, and competencies for specific courses within the program [1], [2]. Some have also gone further and developed frameworks to successfully leverage the learning outcomes and competencies in a systematic way when designing, delivering or revising a course within the program [3], [4], [5], [6].

Assessment is a crucial component of learning. Hence having defined learning outcomes and competencies, the next step is to define assessments and then to map student performance in these assessments to competencies. For example, the Course Life-Cycle Competency (CLCC) framework developed at the School of Information Systems provides a systematic approach to assess competencies and then uses the results of this assessment to give valuable feedback to both students and instructors teaching the course [5], [6]. Tovar and Soto provide a framework, where they assess basic competencies that high school students must have, before they can embark on a Computer Engineering program [3]. Here the emphasis is on identifying whether the students have the necessary pre-requisite competencies before starting the program. Bekki et al., propose a modified-mastery based learning approach that uses a finite cycle of formative assessments and feedback to demonstrate mastery of the competencies for the course [7]. This is achieved through use of three types of assignments; "evidence assignments", which provide evidence of the students' attempt to learn the topics; "competency assignments", which assess the mastery of a competency; and "enrichment assignments", which present challenges beyond what is covered in the course material and help extend students' understanding of the related topics.

With more and more emphasis on online learning for higher education, e-assessment is also increasingly becoming important. Sitthisak et al., present a system for automatically generating questions from a competency framework, based on question templates, criteria for effective questions, and the instructional content and ability matrix [8]. Ilhai et al., show how a competency based assessment can be extended to online learning environments using assessment grid and feedback [9].

Sharp emphasizes the importance of developing learning outcomes and competencies for the course and postulates that in a flipped classroom approach, the class time is to be devoted to activities that support the knowledge and skills outcomes and competencies [10].

B. Learning Outcomes and Competency Framework

In Figure 1, we show the key components of the Learning Outcomes and Competency Framework (LOCF) implemented at the School of Information Systems, Singapore Management University [4], [5] and [6].

The LOCF consists of three major components: learning outcomes, competencies and assessments. While the learning outcomes have been established at the program level, competencies and assessments are defined at the individual course level.

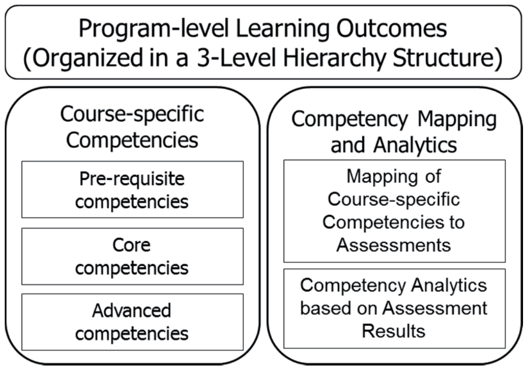

Fig. 1. Learning Outcomes and Competency Framework

For each 1st level learning outcome, several 2nd level learning outcomes have been defined, and each 2nd level learning outcome has several competencies attached to it.

The second important component of the LOCF is competencies. Contrary to the learning outcomes which are defined at the program-level (and are, thus, common for all core as well as elective program courses), the competencies are defined at the individual course level. These competencies are defined by the teaching staff to describe "what the student is capable of doing" on completing the course. Core competencies refer to those competencies that all students are expected to acquire and demonstrate on completing the course.

For a specific course, in addition to the core competencies, two additional competencies can be defined namely, pre-requisite and advanced. Pre-requisite refers to the competencies that a student must acquire and demonstrate before starting a course; these are used as building blocks for the course in question. Advanced refers to those competencies that a subset of students doing the course may acquire and demonstrate on completing the course.

When designing a curriculum it is best practice to have some higher level courses build on competencies acquired in the lower level courses. For example, in the School of Information Systems, course progression is designed such that competencies acquired in the Introduction to Software Foundations (ISSF) become the pre-requisites for the subsequent course namely, Object Oriented Applications Development (OOAD).

The third component of the LOCF is assessments. The competencies are mapped to individual assessments in a course and the results of the assessments are analysed. This analysis provides insights into the extent to which the competencies have been acquired by the students. Several methods of assessments are used in the student evaluation process namely labs, quiz, project, exam, and case studies. For measuring the alignment within a course, in our framework, we use the course level competencies and assessments defined in the course.

C. Flipped Classroom Learning

Flipped classroom is a form of blended learning, where online learning is systematically integrated with periodic face-to-face interaction with instructor. In essence it comprises of two components (see Figure 2), the first component is the directed computer-based individual instruction outside the classroom, followed by the second component of interactive learning activities inside the classroom [11]. Careful design and implementation of both these components are essential for the ultimate success of the flipped classroom.

There is strong research evidence that supports the flipped model. Bransford et al., argue that in order to develop competence in an area of inquiry, students must firstly have a deep foundation of factual knowledge, secondly, understand facts and ideas in the context of a conceptual framework, and thirdly, organize knowledge in ways that facilitate retrieval and application [12].

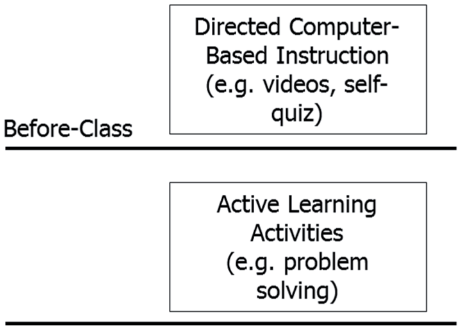

Fig. 2. Flipped Classroom Pedagogy

In a traditional classroom setting, most of the time is spent on transmitting the factual knowledge. With the flipped model, more time is spent in class on student working on problems with peer and faculty interaction, giving them the ability to correct misconceptions, and begin to build frameworks for organizing the material themselves [13]. Additionally, Bransford et al., also emphasize the need for a 'metacognitive' approach to instruction so that students learn to take control of their own learning by defining learning goals and monitoring their progress in achieving them. The flipped model provides opportunity for students to use higher cognitive functions in the classroom with guidance from the instructor, and with clearly defined competencies, students can monitor their learning [14].

There have been number of attempts in implementing flipped classroom learning in programming and other information systems courses. In some instances the entire course has been flipped and in others some specific sessions within a course. However, there have been mixed result in terms of effectiveness, where some have reported positive learning impact and others see neutral or not much improvement in learning experience when compared to traditional face-to-face learning [15].

Maher et al., describe their experiences flipping four different computer science classes across multiple semesters over two years [16]. They further share the instructional design challenges of a flipped classroom namely structuring preparatory work, effectively delivering the instructional content out-of-class, designing active learning activities for students to practice critical competencies during in-class sessions, and structuring student interaction to best leverage social learning and peer instruction. In their work they do not report any comparisons on learning improvements due to the flipped classroom in terms of competency acquisition or exam results. However, they conducted a survey to gather student perceptions about the flipped classroom approach and report that majority of students were pleased with the flipped course approach and the students were more engaged with the material and with their peers as a result of the flipped approach. One of the major concerns was that many students did not realize that the content provided prior to the class sessions through online and textbook reading was part of the learning experience. This highlights that the importance of effectively communicating the flipped pedagogy to the students.

Horton et al., compared a traditional CS1 offering with a flipped offering delivered the following year to a comparable student population and observed that learning as measured by final exam performance increased significantly [17]. They also observed that the overall time spent by the students in the traditional and flipped offering remained the same. They report that there were some differences in the traditional vs flipped offering in terms of the exam questions, instructors, level of difficulty of the assignments, therefore, any improvements in the learning could have been due to any of these factors.

One of the key challenges with flipping the programming classroom is the additional workload for preparing preparatory material, mostly videos and online tutorial explaining the programming concepts and principles. Baldwin has studied this challenge by flipping an introductory programming course for non-computing majors in an effort to see if freely available video lectures could support it [18]. Though his research does not attempt to compare the learning outcomes achievement between traditional and flipped classroom, the work concludes that open resources can support such a course, but just barely. This result underscores that for a successful flipped approach, additional effort is indeed needed for preparing focused preparatory material in the form of videos and other online content.

In order for the flipped model to be effective it is essential that students engage in pre-class work. Horton and Campbell studied the reward structure that will help motivate students to be more engaged during a one-month period of third-year database course [19]. Giving a small grade to the pre-class preparation exercises resulted in high student participation and nearly 85% of the students completed the exercise sets, and 76% reported spending more or much more time preparing during the flipped portion of the course than during the traditional portion.

An important factor contributing to the success of flipped classroom is that the students show up to class prepared. Lacher and Lewis tried to address this issue by having students take online multiple-choice quizzes based on the preparatory video content and have an element of the course grade come these quizzes [20]. However, this approach did not seem to have much impact on student preparedness and consequently on their grades. The authors argue that this result is because the multiple-choice quizzes test surface learning and do not support deep learning. They propose that when designing introductory programming courses using the flipped classroom model, instructors should design the out of class preparatory work by asking students to answer questions that test deeper understanding such as those involving filling in gaps in code segments or writing functions to do very simple calculations.

In the subsequent sections, we report our experiment with using traditional and flipped classroom and analyse the results.

III. THE STUDY

The main objective of this study is to compare traditional and flipped classroom pedagogy and answer the following question:

Will “flipping” the class impact student learning in terms of final exam score, competency acquisition and feedback levels.

In order to answer the first two aspects of this question, we use the measure of student performance in the final exam and also the level of competency acquisition. To answer the third aspect of this question, we use a survey instrument.

This study was conducted in the first year course, Introduction to Software Foundations (ISSF) over two cohorts from academic years, August-December 2013 and 2014. In 2013 the cohort size was 273 and in 2014 it was 280, and all were freshmen year students.

In 2013, the course was delivered using the traditional pedagogy and in 2014, it was delivered using the flipped classroom pedagogy. In both deliveries, the course content and teaching team were the same. The total number of assessments was six for both 2013 and 2014, comprising of quizzes, lab tests and a final exam. In 2013 and 2014, the competencies covered in the first assessment (Quiz 1) were the same, and this was also true for the competencies covered in last assessment (Final Exam). Additionally, the first and last assessments occurred in the same weeks (week 4 for Quiz 1 and week 15 for Final exam). Hence we use these two assessments as measurement points in 2013 and 2014 for comparison.

A. Course Details

ISSF is a foundation course of the BSc (Information Systems) program delivered in the first semester of the first academic year. The semester extends over a 15 week period with 12 weeks of 3 hour class sessions per week. The course focuses on the fundamental building blocks of a software application. Students learn programming fundamentals through the use of object-oriented programming concepts. As a part of the course, students are required to design, code, and test software applications using the Java programming language.

Each year, about 280 students take the ISSF course and are divided into seven sections of about 40 students each. Each section is managed by two instructors and two teaching assistants (TA). The instructors take the overall responsibility for designing and delivering the course, designing and delivering labs, and supporting student project work, and the TAs assist the students with labs.

The entire teaching team is present in all classes, allowing efficient support during class exercises and lab sessions as well as consultations outside of class time.

For the ISSF course, 36 core competencies have been defined. Given the technical nature of the course, these competencies mainly address the program-learning outcome, architecture analysis and design skills, and implementation skills. Figure 3 shows an excerpt of the competencies for ISSF course.

2. Software and IT architecture, design and development skills (Level 1 Learning Outcome)

2.1 …

2.2 Architecture analysis and Design skills (Level 2 Learning Outcome)

- Apply object oriented concepts and principles such as classes versus objects, single responsibility principle and data encapsulation (C1-Course Specific Competency)

2.3 Implementation skills

- Apply acquired knowledge and experience to write an algorithm for solving efficiently a problem (C2)
- Apply efficiently the Java API or user-defined API to look for information about a particular class or method or constant so as to use it appropriately in the code (C3)
- Write effectively a Java class (with instance attributes - primitive or reference type -, default and specific constructors, getters and setters, instance methods with business logic, and toString() method) that compiles and runs with a test class, producing a given output trace (C4)
- Explain the difference between parameters (primitive or reference types located in the method definition) and arguments (primitive or reference types located in the method call) when invoking a method. Notion of method signature (C5)

Fig. 3. Excerpt of Competencies in the ISSF Course

B. Traditional vs Flipped Pedagogy

Figure 4 shows the detailed flow of activities along with the time frame for both the approaches. In the traditional pedagogy, the three hour class time is divided into two hours of lecture and one hour of in-class exercises.

The in-class exercises help students to apply the concepts and principles covered during the lecture. In addition, students are given take-home lab exercises. On a need to basis, students are given personalized tuition by the student TA, to help them tackle the take home labs.

Flipped classroom pedagogy consists of three steps.

Step 1-Before Class: Students prepare before coming to class. Each week they are expected to watch tutorial videos prepared by the teaching team which are uploaded on YouTube. Videos are organised by learning objects, one video per learning object, and each video is limited to 10 to 15 min.

The students access the links to the videos through the Learning Management System, eLearn. After watching the videos, the students are expected to complete the self-check quizzes which are posted on eLearn. Each video is associated with one quiz comprising 3 to 10 MCQs. The students can take these quizzes any number of times thus supporting formative assessment.

Step 2-In-Class: The first thirty minutes are dedicated to analysing the answers to the self-check quizzes and clarifying misconceptions. Following this, the teaching team distribute to students a set of exercises, arranged in order of complexity from easy to complex.

This allows students to tackle the exercises at their own pace. It is an advantage for students with no programming background but also for students with prior experience as they can move faster and can reach the challenging one that are towards the end of the exercise set. The teaching team comprising 2 instructors and 2 TAs for a class of 40 go around and provide feedback on a one-to-one basis. Students who complete early are also encouraged to help their peers. The teaching team encourages this behaviour and the seating plan arrangement is designed to facilitate this interaction. About 30 minutes before the end of the class, one instructor provides feedback to the whole class on the common mistakes observed during the session, the pitfalls to avoid and the best practices to adopt. This is a reinforcement to the feedback provided individually while the students worked on the exercises.

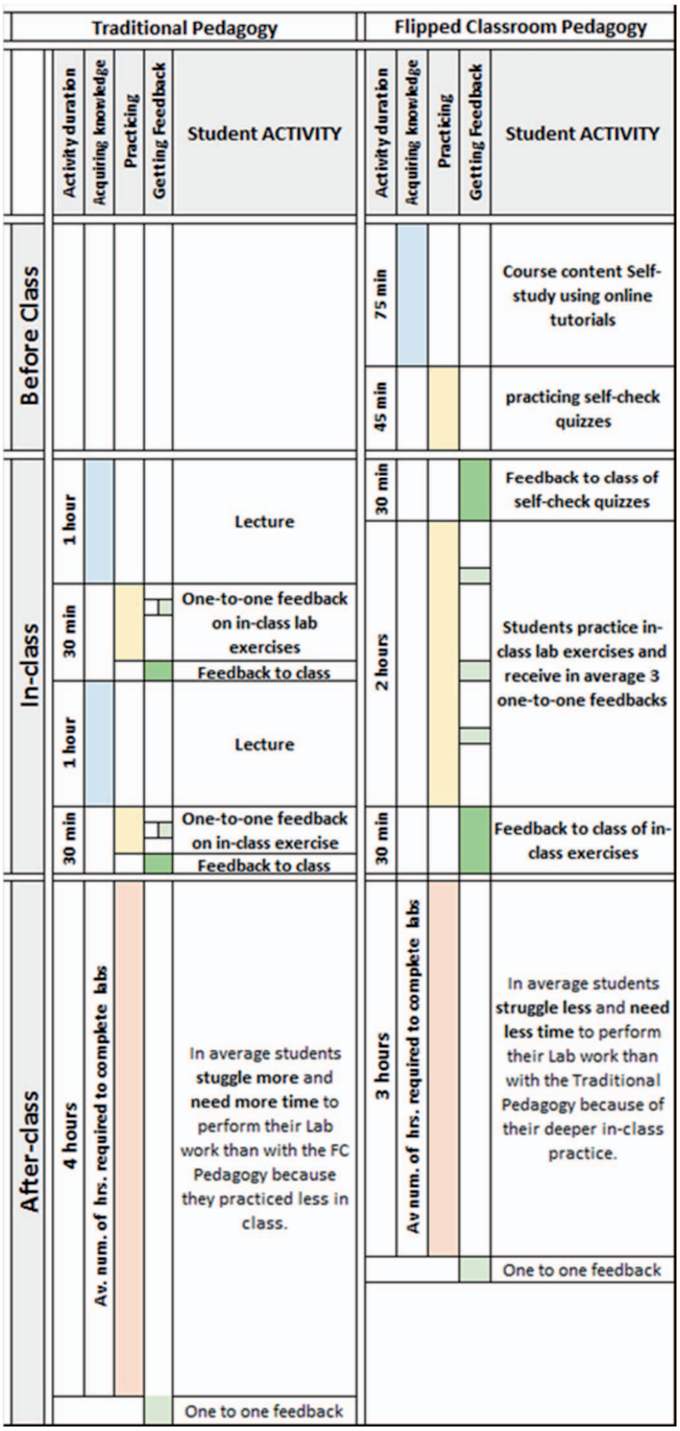

Fig. 4. Implementation of the Traditional and Flipped Classroom Pedagogies

Step 3-After-Class: The students are given a set of additional lab exercises that they are required to work on. These are usually more advanced compared to the ones done in the class, and are designed to ensure students develop “learning to learn” skills. On a need to basis, students are given personalized tuition by the student TA, to help them tackle the take home labs.

IV. RESULTS AND DISCUSSION

The cohorts in 2013 (traditional) and 2014 (flipped) were of comparable size and with similar percentage of genders. Table 2 shows the details of the cohorts

TABLE I. COHORT FOR TRADITIONAL AND FLIPPED

	Total	**% Males**	**% Females**
Traditional (2013)	273	61%	39%
Flipped (2014)	280	67%	33%

To study the impact of flipped classroom on student learning we use three components namely, final exam scores, competency acquisition and feedback levels, and compare to the previous traditional run (non-flipped).

1) Final Exam Scores

At the end of the term all students take an exam in Week 15. It is a 3 hour closed book exam covering all topics in the course.

The total number of points for the exam is 70 marks and this contributes to 35% of total marks for ISSF course. 50% of the questions carrying 35 marks were identical in 2013 and 2014, and the rest of the questions were comparable in terms of type of questions, level of difficulty, and topic and competency coverage.

From Table II, based on the exam results, it can be observed that there has been a substantial increase of 17% in the average mark scored by the cohort using the flipped pedagogy. There was also an increase in the minimum and maximum scores for the flipped pedagogy. Most notably the flipped classroom resulted in 20% reduction in the percentage number of failures compared to the traditional.

TABLE II. COMPARISON OF EXAM RESULTS

	Avg.	**Min Score**	**Max Score**	**Std. Dev**	**No Of Failures**
Traditional (2013)	43	5	67.5	12.8	78
Flipped (2014)	50.4	18	68	10.3	26

2) Competency Acquistion

In the final exam a large part of the competencies that are supposed to be acquired for the complete course are tested. 70%

of the course competencies are covered in the final exam that is 25 out of the total 36 competencies. Therefore, in order to compare the competency acquisition, we only report the results from the final exam. Table III shows an excerpt of competencies that were within the scope of the final exam.

Each exam question is mapped to the corresponding competencies tested. Each question has a total mark and threshold mark. If a student gets above the threshold, it is deemed that the competencies related to that question have been acquired. The questions were designed such that in some instances one question addressed more than one competency, and in other instances one question addressed only one competency.

Due to space constraint, we only show the detailed comparison of competency acquisition for the traditional and flipped using the above four competencies namely C8, C13, C16 and C9 (see Table III, IV and V).

TABLE III. EXCERPT OF COMPETENCIES TESTED IN THE EXAM

#	Competency Definition
C8	Define and use effectively Java primitive variables with some commonly used Java operators (pre/post increment/decrement operators, notion of precedence and associativity, etc.)
C13	Apply repetition constructs in Java (while, do-while, for loop) to solve a repetition problem
C16	Apply conditional constructs in Java (if, if-else, if-else-if, nested-if, switch) to control the path of execution of statements
C9	Draw a memory state diagram to deduce an output trace while using several classes, ArrayLists of objects and methods invocations, etc.

In both the traditional and flipped models, Q4 and Q6 were similar (see Table IV). The threshold mark required to confirm competency acquisition was 3 points out of a max of 5.

TABLE IV. EXCERPT OF EXAM QUESTIONS AND COMPETENCIES

Question #	Competency Tested	Question Description
Q4	C8, C13, C16	Trace an output given some Java code
Q6	C9	Draw a memory state diagram

TABLE V. COMPARISON OF COMPETENCY ACQUISITION

	% of Students scoring above the threshold	
	Traditional	*Flipped*
C8,C13,C16 (Q4)	53%	75%
C9 (Q6)	23%	48%

As can be observed from Table V, there was an increase of 22% for competencies C8, C13 and C16 and an increase of 25% for C9 in the flipped classroom compared to the traditional. Across all competencies tested in the final exam (24 competencies), the increase in acquisition ranged from 4% to 52%. These results confirm that the flipped approach enhances student competency acquisition.

3) Feedback Levels

In order to measure the feedback levels we compare the traditional and flipped classroom in terms of the time allocated for the students to practice in the class and for the teaching team to provide feedback in the class (Table VI).

TABLE VI. COMPARISON OF FEEDBACK LEVELS

	Traditional	Flipped Classroom	Impact on Feedback Levels
Average number of exercises done in class	2	6	Students get to practice on more exercises in the class
In-class practice session	40 min	120 min	3 times more practice time
One-to-one feedback sessions	1	3	3 times more one-to-one feedback
Feedback to the whole class	20 min	60 min	3 times more feedback to entire class (misconceptions, common mistakes, best practices etc.)

V. THREATS TO VALIDITY

Though both the traditional and flipped approaches followed the same curriculum, weekly lesson plans, in-class exercises, take-home exercises, and the same teaching team, the student cohorts were different. Therefore, one might argue, having different cohorts could have affected the results of the study. To partially neutralize this issue, we conducted a Quiz at the beginning of the course in Week 4 for both the cohorts.

TABLE VII. QUIZ 1 RESULTS

Cohort	Average	% Of Failures
2013	12.1	4%
2014	11.4	14%

As seen from Table VII, the 2014 cohort (flipped classroom) actually performed worse that the 2013 cohort (Traditional) and there were more failures in 2014. This confirms that in fact, the 2014 cohort was slightly weaker than the 2013 cohort. However, the final exam scores show that the 2014 cohort using the flipped classroom did much better than the 2013 cohort.

VI. CONCLUSIONS

In this paper, we study the comparison of a traditional and flipped offering of an introductory programming course. Our research sought to compare the impact of flipped classroom on student learning in terms of final exam scores, competency acquisition and feedback levels. From the analysis of the data conducted in this study we can conclude that flipped classroom results in enhanced student performance in the final exam and also contributes to improving competency acquisition. One concern that the teaching team had was whether the flipped model required greater student time investments. Through informal discussions with the students, we understood that in comparison to the traditional, for flipped model, though the students were spending about 2 additional hours for pre-class preparations, the post class exercise time was reduced by 1 hour. So overall the students spent an additional 1 hour more for each week in the flipped classroom, but this did not seem to have any negative impact on student attitude and motivation.

Since this study has found positive support for flipping the introductory programming course, we recommend future work to further explore application of this pedagogy for other courses in the curriculum.

ACKNOWLEDGMENT

The authors would like to acknowledge the efforts of the teaching team involved in the design and delivery of the ISSF course.

REFERENCES

[1] B Villalobos, J., Gonzalez, O., Jimenez, C and Rueda, F. 2011. Curricula design model for designing and evaluating systems and computing engineering programs, *41st ASEE/IEEE Frontiers in Education Conference (FIE 2011)* 2011, pp. S4E-1-S4E-7.

[2] Lister, R., et al. 2012. Toward a shared understanding of competency in programming: An invitation to the BABELnot project. *Proceedings of the Fourteenth Australasian Computing Education Conference (ACE2012)*, 2012, Melbourne, Australia, pp53-60.

[3] Tovar, E and Soto, O. 2009. Are new coming computer engineering students well prepared to begin future studies programs based on competences in the European Higher Education Area?, 2009, Proceedings of the *39th IEEE Frontiers in Education Conference*. Imagining and Engineering Future CSET Education, pp. 1-6

[4] Baumgartner, I and Venky Shankararaman. 2013. Actively linking learning outcomes and competencies to course design and delivery: experiences from an undergraduate Information Systems program in Singapore. *Proceedings of IEEE Global Engineering Education Conference (EDUCON 2013)*, 2013, Berlin, Germany, pp 238-246.

[5] Ducrot, J and Venky Shankararaman. 2014. Measuring student performance and providing feedback using competency framework. *IEEE 6th International Conference on Engineering Education*, 2014, Kuala Lumpur, Malaysia.

[6] Venky Shankararaman and Ducrot, J. 2015. Leveraging Competency Framework to Improve Teaching and Learning: A Methodological Approach. *Journal of Education and Information Technologies*. Springer, Vol. 20, No 1, March 2015, pp1-29.

[7] Bekki, J.M., Dalrymple, O. and Butler, C.S. 2012. A mastery-based learning approach for undergraduate engineering programs. *Proceedings of the IEEE Frontiers in Education Conference (FIE)*, 2012, pp. 1-6.

[8] Sitthisak, O., Gilbert, L. and Davis, H.C. 2008. Deriving E-Assessment from a Competency Model. *IEEE International Conference on Advanced Learning Technologies* (ICALT 2008), 2008, pp. 327-329.

[9] Ilahi, M., Belcadhi, L.C. and Braham, R. 2013. Competence web-based assessment for lifelong learning. *Proceedings of the First International Conference on Technological Ecosystem for Enhancing Multiculturality* (TEEM '13), 2013, Francisco José García-Peñalvo (Ed.). ACM, New York, NY, USA, pp541-547.

[10] Sharp, J.H. 2014. Journey toward a flipped C# programming class: An Experience Report. *Proceedings of the Information Systems Educators Conference*, Baltimore, Maryland USA.

[11] Bishop, J.L., and Verleger, M.A. 2013. The flipped classroom: A survey of the research. *ASEE National Conference Proceedings*, Atlanta, GA, 2013.

[12] Bransford, J.D., Brown, A.L., and Cocking, R.R. 2000. How People Learn: Brain, Mind, Experience, and School, Washington, D.C.: *National Academy Press*.

[13] Stone, B. B. 2012. Flip your classroom to increase active learning and student engagement. *28th Annual Distance Teaching & Learning Conference*, University of Wisconsin, Madison, USA, pp1-5.

[14] Rutherford, R.H and Rutherford, J.K. 2013. Flipping the Classroom - Is It For You?. *Proceedings of Special Interest Group for Information Technology Education Conference, SIGITE'13*, October 10–12, 2013, Orlando, Florida, USA, pp20-22.

[15] Yarbro, J., Arfstrom, K.M., McKnight, K. and McKnight, P. 2013. Extension of Review of Flipped Learning. Last Accessed on 10th Jan 2016, http://www.flippedlearning.org/review.

[16] Maher, M.L., Latulipe, C., Lipford, H. and Rorrer, A. 2015. Flipped Classroom Strategies for CS Education. *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, ACM, pp 218-223.

[17] Horton, D., Craig, M., Campbell, J., Gries,P. and Zingaro, D. 2014. Comparing Outcomes in Inverted and Traditional CS1. *Proceedings of the 2014 conference on Innovation & technology in computer science education*, ITiCSE'14, June 21–25, 2014, Uppsala, Sweden, pp261-266.

[18] Baldwin, D. 2015. Can We "Flip" Non-Major Programming Courses Yet?. *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, ACM, pp 563-568.

[19] Horton, D and Campbell, J. 2014. Impact of Reward Structures in an Inverted Course. *Proceedings of the 2014 conference on Innovation & technology in computer science education*, ITiCSE'14, June 21–25, 2014, Uppsala, Sweden.

[20] Lacher, L.L and Lewis, M.C. 2015. The Effectiveness of Video Quizzes in a Flipped Class. *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, ACM, pp 224-228.