Vol.4 No.2 (2021)

Journal of Applied Learning & Teaching

ISSN : **2591-801X**

Content Available at : **http://journals.sfu.ca/jalt/index.php/jalt/index**

Experiential study to integrate theory and lab for an introductory programming course in an Indian engineering classroom

Anupama Ghattu[A] [A] *Associate Professor, Teaching Learning Center, SRM University AP, Andhra Pradesh, India*

Balaguruprasad Narayanan[B] [B] *Associate Professor, Teaching Learning Center, SRM University AP, Andhra Pradesh, India*

Kavitha P.[C] [C] *Assistant Professor, BVRIT Hyderabad College of Engineering for Women, Telangana, India*

Keywords

Active learning;
coding;
hands on practice;
ICT;
logical thinking;
problem-solving;
programming.

Correspondence

anupamaghattu@gmail.com [A]

Article Info

Received 24 July 2021
Received in revised form 30 October 2021
Accepted 1 November 2021
Available online 1 November 2021

DOI: https://doi.org/10.37074/jalt.2021.4.2.16

Abstract

At present, programming is a crucial competence for employability that is a requirement for engineering graduates. In India, all engineering students are required to register for an introductory programming course at the Freshmen level. Over the years, this introductory programming course is taught in a traditional chalk and talk approach. Traditionally, theory and lab classes were conducted separately. This kind of traditional approach limited students' ability to think logically and develop problem-solving skills through programming. On that account, to encourage experiential learning which improves students' logical thinking and problem-solving skills, this action research study looked at the possibility of implementing an integrated theory lab approach to a freshman C Programming course. Based on the course credits, the students met eight hours each week as part of the integrated implementation.

A pretest posttest experimental research approach was used to examine the effects of the integrated pedagogical strategy. The classes for the students in the integrated approach were conducted directly in the laboratory to ensure meaningful learning and to enable more hands-on coding practice in the classroom simultaneously while the concepts were taught. Active and peer learning activities were designed to help students learn meaningfully. Due to the short intervention time and small sample size, there was no statistically significant effect to this integrated approach. However, the classroom activities increased hands-on practice time and regular formative assessments showed that students' programming competency including logical thinking and problem-solving skills were improved through this integrated approach.

1. Introduction

The current engineering education in India continues to follow the traditional teacher-centric approach as the major mode of instruction delivery. This mode of delivery mainly includes lectures (chalk-and-talk), large classrooms, separate theory, and lab sessions. It is not different for programming courses in the curriculum; these courses are also taught traditionally where the students become passive learners and cannot translate the programming concepts from the classroom lectures into effective programs in the laboratory. This approach also has challenges to impart active learning that can improve problem-solving and logical thinking skills in students. In the Western education system, it has been a practice to have integrated courses specifically for programming to promote experiential and hands-on learning; this gives learners the ability to think logically and solve problems. Given that, it is essential for the instructors to consider various pedagogical strategies to transform the teaching learning process for meaningful learning. However, this is one of the less experimented and researched arenas in Indian engineering education. Hence, the purpose of this research study is to discuss implementation of the instructional design strategy – integration of theory and lab sessions (hands-on approach) for an introductory programming class to enhance problem-solving and logical thinking skills in freshmen engineering students.

Learning the fundamentals of computer and introduction to programming (using C / Python language) is one of the foundational courses during the first year of engineering education irrespective of the department. As programming is a hands-on course, delivering it in the traditional lecture mode will not benefit the learners in the way intended. Moreover, learning to code efficiently will also help students think critically in an organized manner to achieve the desired results. Furthermore, the change in the learning needs and career opportunities of the current generation learners has tremendous impact on the teaching and learning process of the higher education system. On this account, it is essential for educators to rethink and find pertinent scientific innovative approaches to enhance student learning. Thus, to ensure experiential learning and cater to students' learning needs, a learner-centered approach must be adopted. One such approach is designing and developing integrated lecture/theory and lab sessions that will facilitate active learning, thereby improving problem-solving and logical thinking skills among students.

The major aim of this action research study is to instill problem-solving and logical thinking skills among students while learning programming. A programming course was taught to first year engineering students who had very little or no programming experience before. It is important for students to understand that the introductory programming course is not just about learning concepts and syntax but when learnt appropriately it will help develop logical thinking and problem-solving skills. However, this is only possible when students practice or learn lines of code alongside while learning the concepts for better conceptualization and improve critical thinking. In addition to hands-on practice to code, classroom group activities like Think-Share-Pair, fishbowl, group discussions, story board and peer evaluation were conducted to motivate and encourage students to experience meaningful learning. Hence, integration of class lectures and laboratories is one such approach to help students learn programming through problem-solving and logical thinking. This action research approach would provide faculty with insights on the implications and challenges of using hands-on learner centric methods to programming that can be modified and improved to implement integrated pedagogical strategy to foster student learning. Action research is said to have an immediate impact on society and education based on the practices that are planned, implemented and reflected during the research cycle (Masters, 1995).

Students in an introductory programming course are deeply focused on knowledge of a particular programming language rather than problem-solving skills. Moreover, the emphasis on mere knowledge leads to a lack in conceptualization and appropriate transfer of programming skills in the laboratory for meaningful learning. To facilitate students with conceptual understanding, this research followed an integrated theory and lab approach to improve student engagement in learning programming through logical thinking and problem solving. This investigation illustrates that integrating theory and lab sessions for an introductory programming class can enhance problem-solving and logical thinking skills in novice programmers.

Logical thinking and problem solving are among the major skills the current generation student needs to be equipped with. As programming/coding requires an individual to think logically to arrive at solutions, an introductory programming course is one of the avenues to instill and/or improve the said competencies in students. However, teaching programming in the traditional chalk and talk manner with separate lab and theory will not achieve the intended outcome. Furthermore, as writing lines of code is like solving a mathematical problem with all the necessary steps and procedures, this can only be achieved through hands-on practice (Ghattu, 2015) With ever changing learning needs of the students in this technology-oriented world, it is important for instructors to think of ways to engage students to foster meaningful learning. Hence an integrated lab and theory approach will facilitate learners to improve their logical thinking and problem-solving skills using programming.

An integrated lab and theory approach provides more hands-on practice time that allows students to learn coding at their own pace by trial-and-error methods. Albert Bandura's (2010) self-efficacy theory says that an individual's belief in their own ability determines how an individual can succeed in completing tasks and events in their lives. Furthermore, this approach motivates students to experiment, work out the solutions and improve their performance. Since this integrated approach was a new intervention in this institution, novice faculty were asked to observe and assist as it would help in their application of this strategy in the future. Naryanan (2015) suggests that continuous observation and practice lead to increased self-efficacy of faculty using instructional technology.

The integrated approach allows students to learn coding in an efficient way through hands-on practice and by improving their logical thinking and problem-solving skills. In the current day, it is crucial for engineering graduates to have exceptional problem-solving and logical thinking skills apart from their domain knowledge. Moreover, these are among the major competencies that are required for better career and higher education opportunities. These skills are acquired over a period with constant and consistent practice and efforts by the students. As these are priority learner needs, it is important that all the stakeholders, specifically the faculty, assist students in all aspects from day one to develop critical thinking, logical thinking, and problem-solving skills. Furthermore, more hands-on practice time will encourage students to program successfully and carrying it forward to subsequent programming courses (Canfield et al., 2012).

2. Literature review

According to Handur et al. (2016), learners lose focus and become passive in a lecture centric programming classroom. Even though there is less research on integrated classrooms, implementation of such instructional design strategy has advantages and challenges. This research emphasizes on how the integrated approach can enhance problem-solving and logical thinking skills in students. Canfield et al. (2012) suggest that a hands-on programming model "provides increased engagement and builds on incoming notions of programming in engineering that result in better learning". According to Handur et al. (2016, p. 163): "Programming courses must facilitate conceptual understanding in students. Conceptual understanding can be defined as ability to observe, interpret and summarize a concept".

The main challenge for students in an introductory programming class is that they need to learn different competences at the same time. These include the programming language, logical thinking and problem-solving skills. In the traditional approach, learners focus mainly on the syntax and semantics. Such an approach fails to translate to the appropriate programming skills in the lab. This in turn limits their ability to identify and analyse a given problem statement. According to Malik et al. (2017), learning programming in a traditional approach is a challenging task for beginners, as they have to develop and build their problem-solving skills along with trying to learn the syntax and semantics of the specific programming language being taught. Moreover, research has shown that for novice programmers, the programming language in itself determines their ability to learn the programming concepts and successfully translate it to write lines of code using appropriate syntax (Stefik et al., 2011).

Learning to program is the need of the hour as our lives are driven by technology. To code effectively one must be able to break a given problem into small sequential tasks to arrive at the solution, this requires exceptional logical thinking and problem-solving skills. 'Learning by doing' helps students develop these skills in an introductory programming class. The traditional process of teaching programming not only makes students passive learners but, it can also instil fear of coding. When taught in this approach, the students mainly focus on learning the syntax and semantics of the specific programming language rather than the actual process of how-to code. Moreover, this takes away the purpose of the introductory programming course where students are intended to learn problem-solving through programming (Miller, 2019).

The availability and flexibility of the integrated lab class will allow faculty to explore various avenues and new strategies to improve student learning. Although there are some studies that determine this problem in various ways, this research is specifically designed to study how an integrated lab and theory classes approach can make students active learners, thereby improving their logical thinking and problem-solving skills. Increasing the time spent practicing the lines of code simultaneously while learning the concepts of programming makes students active learners and ensures meaningful learning. In addition to the integrated approach the in-class group activities enables students to work collaboratively to conceptualize the topics and encourage self-directed learning (Silvia, 2019). This in turn helps students think logically to solve problems through programming.

Another advantage of this learner-centric approach was 'peer instruction' during the lab practice sessions, where students assisted their peers to get to the solutions using coding; this is known as the 'scaffolding' technique developed by Jerome Bruner, where a more competent peer helps another student whenever necessary and continues to aid until it becomes unnecessary (Wood et al., 1976; Titterton et al., 2010). Students tend to learn better when programming classes are shifted to increased collaborative and hands-on practice hours in the lab. In an introductory programming class, active learning happens through hands-on practice and trial & error approaches when lecture hours are replaced with lab hours per week; in addition to active student involvement, this approach also facilitates in developing logical thinking and problem-solving skills necessary for programming. Moreover, increased guided practice time also allows the faculty to cater to individual student learning needs. According to Berland et al. (2013), practicing programming on alternative programming environments which allows beginners to quickly learn complex programming concepts and to execute programs with minimal errors, thereby encouraging learners to code efficiently. Titterton et al. (2010, p. 79) suggest: "Instructors also benefit from a deeper window into student progress and understanding".

Due to the global digital revolution, problem-solving has become one of the essential 21st century skills for all learners. In this regard, it is necessary for faculty members to adapt pedagogical strategies to encourage and facilitate learners to develop and build on their problem-solving skills; "Problem solving skills refer to the capability of tackling issues and problems in different domains, such as personal, social and work" (Kožuh et al., 2018, p. 3). Problem-solving skills are an individual's ability to reason and think critically which also improve logical thinking skills of the learners (Seyhan, 2015). Albrecht (1984) mentioned that, to think logically is to arrange things, ideas, and facts in a sequence

that make sense and that leads to solving the problem. Logical thinking is the basis to all problem-solving in a step-by-step procedure.

3. Implementation

The purpose of this study was to investigate if there was any change in student attitudes towards programming, and to enhance students' logical thinking & problem-solving skills. The questionnaire also included a part on student demographic information. In the spring semester, there were six sections across various engineering disciplines which had the introductory programming course as part of their first-year curriculum. Among those, two of the Computer Science Engineering department sections were chosen for this study: one experimental group and one control group. Based on the four-year degree curriculum, the computer science department students had advanced programming courses throughout the program, and these were chosen for this study. The two sections were taught by two different instructors with similar course syllabi, course structures, course objectives, course outcomes, course requirements, including the learner expectations. Students enrolled for this program were admitted based on a standardized test and are college freshmen who have similar learning experiences. In total, 120 students were admitted into the Bachelor of Technology (B.Tech) - Computer Science Engineering program who were divided into two sections of 60 students each; one section (A) was the experimental group and the other section (B) was the control group. The primary objective of this exploratory study was to compare between the experimental group and the control group in an introductory programming class. Based on the curriculum, the introductory programming was scheduled for five theory and three lab hours per week; all the sections were allotted those hours. However, for the experimental group, this schedule was converted to six lab and two theory sessions per week. This implies that each student in the experimental group had twice the practice time in a lab as compared to the control group. The lab practice time was used to understand, analyze and solve a given problem statement by breaking it down into smaller chunks and following appropriate strategies to reach a solution. The pre-test and post-test were administered to both control and experimental groups at the beginning and the end of the integrated instruction method, respectively.

In this integrated approach:

(a) Each student had two lab sessions per week. That is each week there were two lab sessions for the class of 60.

(b) As there were 60 students per lab session, there was one instructor and two assisting faculty to assess and help the students with the coding practices.

(c) More time was allotted for students to practice coding. These programming sessions were followed by a brief lecture on the topics and then the students practiced and tried the concepts for better understanding.

(d) In addition to the lab practice, activities like Think-Pair-Share, FishBowl, Group Discussions, Story Board and Minute Papers were designed to help students learn meaningfully.

(e) Quick formative assessments using Kahoot and Edmodo were conducted to assess and reflect student learning.

In this integrated approach, students were able to learn and test their programming skills at the same time; and this created a competitive attitude that enhanced peer learning among the students. This allowed students to learn from their mistakes and from each other. Furthermore, the classroom activities helped students conceptualize the content and connect their learnings. Minute papers were used often to review student understanding of the concepts taught in the session and to reiterate them if necessary. All the group activities helped students focus, connect and involve themselves to promote critical thinking skills. The story board was a group activity which was designed to improve creativity and critical thinking skills; for this activity, the students were also introduced to peer evaluation through rubrics. In addition to the class activities, there was a simple programming language test that was conducted as part of the asessment.

There was one final exam as part of the summative assessment for this course. Two term exams, team work, quizzes, practice programs, coding, story boards and a test were part of the formative assessments to assess students' conceptualization, coding skills, problem solving skills and logical thinking skills. Feedback was provided to students for continues improvement improve and progress. Below are some sample class assessments to understand student learning and help cater to their requirements.

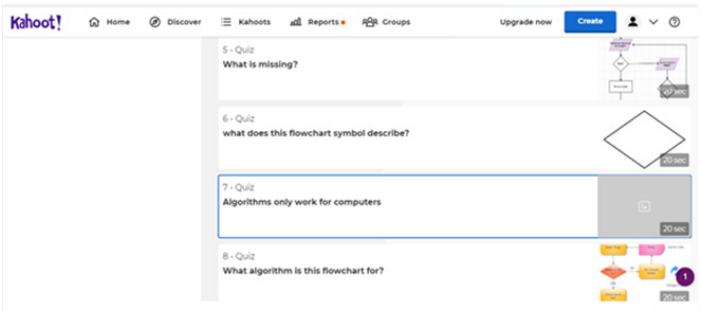


Figure 1: Snapshot of Kahoot assessment conducted during the session.

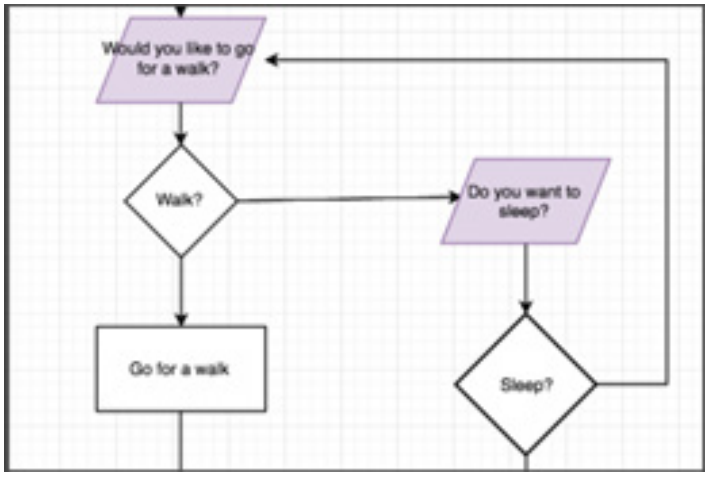


Figure 2: Snapshot of flowchart 1 in Kahoot assessment.

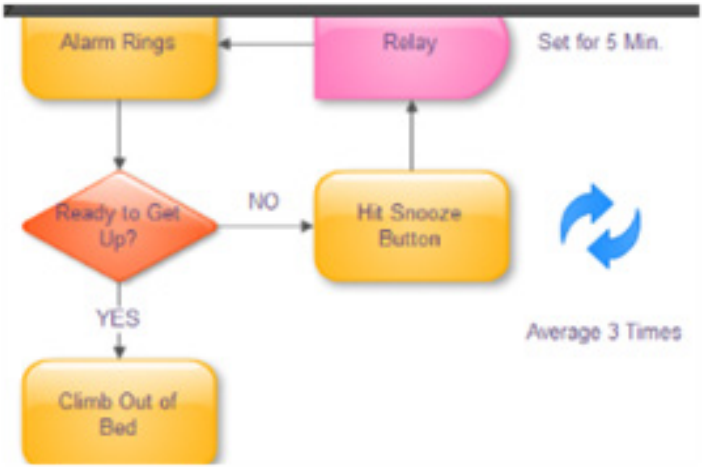
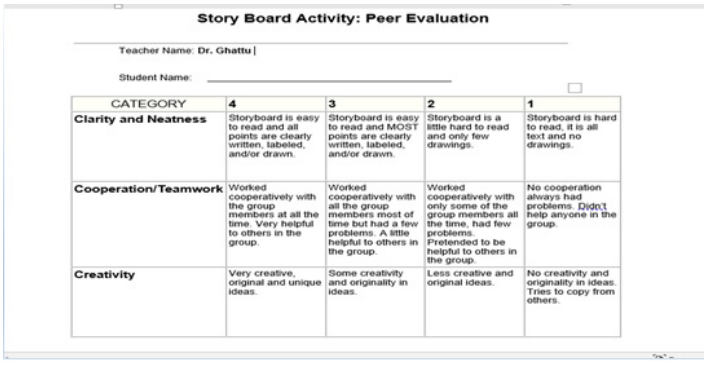


Figure 3: Snapshot of flowchart 2 in Kahoot assessment.

What is the value of n at the end of the program execution? #include <stdio.h> void display(); int n = 5; // global variable int main() { ++n; // variable n is not declared in the main() function display(); return 0; } void display() { ++n; // variable n is not declared in the display() function printf("n = %d", n); }

Figure 4: Snapshot of predicting output in the practice session.



Story Board Activity: Peer Evaluation

Teacher Name: Dr. Ghattu |

Student Name: _____

CATEGORY	4	3	2	1
Clarity and Neatness	Storyboard is easy to read and all points are clearly written, labeled, and/or drawn.	Storyboard is easy to read and MOST points are clearly written, labeled, and/or drawn.	Storyboard is a little hard to read and only few drawings.	Storyboard is hard to read, it is all text and no drawings.
Cooperation/Teamwork	Worked cooperatively with the group members at all the time. Very helpful to others in the group.	Worked cooperatively with all the group members most of time but had a few problems. A little helpful to others in the group.	Worked cooperatively with only some of the group members all the time, had few problems. Pretended to be helpful to others in the group.	No cooperation always had problems. Didn't help anyone in the group.
Creativity	Very creative, original and unique ideas.	Some creativity and originality in ideas.	Less creative and original ideas.	No creativity and originality in ideas. Tries to copy from others.

Figure 5: Peer evaluation rubrics.

4. Analysis and results

The pre-test and post-test survey method was conducted between the experimental group and control group to study and compare student problem-solving skills. As hands-on programming practice can help develop problem-solving and logical thinking skills, a ten-question aptitude survey was administered to compare problem-solving between the control group and the experimental group. The performance of students improved from pre-test to post-test in the experimental group as compared to the control group. There was improvement in the average test scores between the groups from pre-test to post-test as shown in Figure 6.

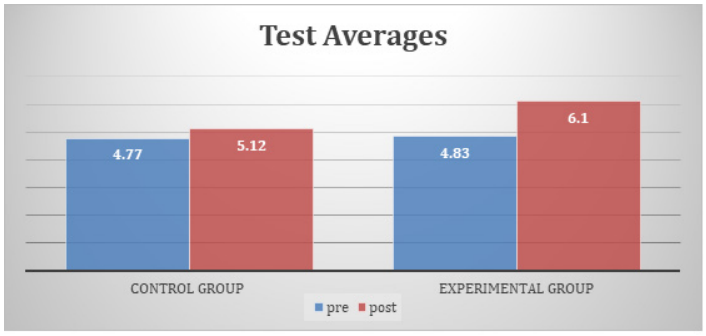


Figure 6: Average scores between experimental and control groups.

This observation shows that there was not much change in the average scores in the control group from pre to post. However, even though the average test scores have not improved exponentially within the experimental group and between both the groups, Figure 5 illustrates: (a) the score fell to -1 for one student in each group; (b) there was no change in scores for 38 students in the control group and five students in the experimental group; (c) 31 students in the experimental group and 20 students in the control group improved by one point; (d) 23 students in the experimental group and one student in the control group improved by two points from pretest to posttest.

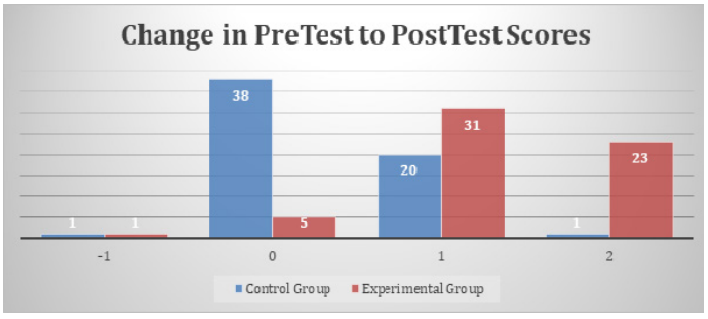


Figure 7: Change of scores related to number of students.

One-way Analysis of variance was performed to examine the test scores of the experimental and control group. Based on the statistical analysis, the test scores with the groups (MS = 1.05, SS = 123.92) and between the groups (MS = 3.68, SS = 3.68) with p = .70, indicated that there was no significant

difference in the test scores between the control and experimental groups. However, all the other interventions used for the experimental group in this study demonstrated that there was improvement in programming and problem-solving approaches of students in the experimental group This indicates that the integrated instructional approach in an introductory programming class improved students' logical thinking and problem-solving skills. Due to the small sample size, this study does not show any significant improvement in the scores between the experimental group and control group. Since this intervention was mainly focused on learning programming through problem-solving, and assessments were part of the lab sessions, there was little to no evaluation on how students would articulate their concepts in a pen and paper testing. Even though there were quizzes and other assignments to assess students' programming skills, the pen and paper testing component would help improve their conceptual knowledge to better prepare them for their end-semester examinations.

5. Conclusion

Introductory programming courses are amongst the mandatory fundamental courses for engineering students during their freshmen year. The main objective of this course is to teach problem-solving through programming, but the traditional teaching approach fails to achieve the outcome. Hence, the integrated theory and lab approach was one of the pedagogical approaches developed to encourage and improve problem-solving and logical thinking skills in students through programming. The results show that this approach has improved student competencies. Students using the integrated approach were able to write lines of code with minimal to no errors. Along with the activities and quizzes, the addition of a written evaluation component to the intervention would help the students with deeper conceptualization. Programming is a skill that the learner (the Computer Science and Engineering (CSE) student) must possess and that can be performed in various programming languages. Coding is the fundamental step in the programming process which requires enormous amounts of logical thinking, flow and problem-solving to address a given scenario. Moreover, one of the best ways to strengthen logical thinking and problem-solving is through constant hands-on practice. This specific integrated approach along with active learning was chosen to demonstrate to the students that hands-on practice can improve their coding skills which are very much required for continuous education, employment and higher education. Wieman (2014) has shown that college students learn better through active learning methods than the traditional lecture approach.

Finally, an integrated lab approach may possibly improve programming, logical thinking and problem-solving skills among students. However, implementing the integrated lab and theory approach is time-consuming as it emphasizes on additional hands-on practice time. In addition to this, there were other challenges concerning infrastructure, manpower and syllabus completion as per university requirements. These challenges can be addressed in a further study to provide better solutions. Furthermore, this research can also be extended to a bigger group of learners to better replicate the results. Also, a foundational course can be designed as a prerequisite to bridge the knowledge gap. Practicing programming on other coding platforms can be one of the ways to instil coding skills in students. In conclusion, the integrated instruction method improves students' problem solving, logical thinking skills and it ensures student learning. It also allows the instructors to cater to individual student needs based on their performance.

6. References

Albrecht, K. (1984). *Brain building: Easy games to develop your problem solving skills. Prentice Hall.*

Bandura, A. (2010). Self-efficacy. *The Corsini Encyclopedia of Psychology*, pp. 1-3.

Berland, M., Martin, T., Benton, T., Petrick Smith, C., & Davis, D. (2013). Using learning analytics to understand the learning pathways of novice programmers. *Journal of the Learning Sciences, 22*(4), 564-599.

Canfield, S. L., & Abdelrahman, M. (2012). Enhancing the programming experience for first-year engineering students through hands-on integrated computer experiences. *Journal of STEM Education: Innovations and Research, 13*(4), 43-54.

Ghattu, A. (2015). *An exploratory study on the impact of mobile wireless technologies in a teacher education classroom*. Doctoral dissertation, Indiana State University.

Handur, V., Kalwad, P. D., Patil, M. S., Garagad, V. G., Yeligar, N., Pattar, P., Mehta, D., Baligar, P., & Joshi, G. H. (2016, December). Integrating class and laboratory with hands-on programming: Its benefits and challenges. In *2016 IEEE 4th international conference on MOOCs, Innovation and Technology in Education (MITE),* pp. 163-168. IEEE.

Kožuh, I., Krajnc, R., Hadjileontiadis, L. J., & Debevc, M. (2018). Assessment of problem solving ability in novice programmers. *PloS One, 13*(9), e0201919.

Malik, S.I., & Coldwell-Neilson, J. (2017). Impact of a new teaching and learning approach in an introductory programming course. *Journal of Educational Computing Research, 55*(6), 789-819.

Masters, J. (1995). The history of action research. In I. Hughes (Ed.) *Action research electronic reader.* The University of Sydney, http://www.behs.cchs.usyd.edu.au/arow/Reader/rmasters.htm

Miller, M. (2019, June). Board 98: Lessons learned from an integrated class-lab approach to a mechanics of materials course. In *2019 ASEE Annual Conference & Exposition*, n.p.

Narayanan, B. (2015). *University faculty comfort in using handheld/mobile technology to communicate with students.* Doctoral dissertation, Indiana State University.

Seyhan, H. G. (2015). The effects of problem solving applications on the development of science process skills, logical thinking skills and perception on problem solving ability in the science laboratory. In *Asia-Pacific Forum on Science Learning & Teaching, 16*(2).

Silvia, D., O'Shea, B., & Danielak, B. (2019). A learner-centered approach to teaching computational modeling, data analysis, and programming. In *International conference on computational science*, pp. 374-388. Springer.

Stefik, A., Siebert, A., Stefik, M., & Slattery, K. (2011). An empirical comparison to the accuracy rates of novices using the Quorum, Perl, and Randomo programming languages. In *Proceedings of the 3rd ACM SIGPLAN workshop on evaluation and usability of programming languages and tools*, pp. 3-8. ACM.

Titterton, N., Lewis, C. M., & Clancy, M. J. (2010). Experiences with lab-centric instruction. *Computer Science Education, 20*(2), 79-102.

Wieman, C. (2014). Stop lecturing me. *Scientific American, 311*(2), 70-71.

Wood, D., Bruner, J. S., & Ross, G. (1976). The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry, 17*(2), 89-100.