A Hybrid Open/Closed Lab for CS 1

Timothy Urness
Drake University
2507 University Ave
Des Moines, IA 50311
timothy.urness@drake.edu

ABSTRACT

In this paper we introduce hybrid labs, an alternative to open or closed labs for CS 1, in which a set of written instructions, demonstration of techniques, and code examples are provided to students in lieu of a lecture. The hybrid lab also consists of several challenges which require students to write code or answer questions based off the concepts introduced in the document. Students are presented with the lab two days prior to a class period and are given an option of submitting solutions to the challenges on their own time (similar to an open lab) or attending the class in which an instructor is available to provide additional help as needed (similar to a closed lab). We compare a section of CS 1 that utilized a combination of hybrid labs and lectures against a section that utilized only lectures. We found no statistical significance between the abilities of the students of the two sections, but surveys show that students found the hybrid labs to be more engaging and preferred the hybrid labs over lectures as means of instruction. Furthermore, instructors found that the hybrid labs allowed for more tailored, individualized instruction for a variety of student abilities.

Keywords

CS 1; Open Labs; Closed Labs

1. INTRODUCTION

The primary goals of a CS 1 course typically involve introducing the fundamental components of algorithm design and development. Many of the objectives common to CS 1 courses are articulated by the ACM/IEEE curriculum guidelines and include if statements, loops, simple I/O, files, methods, and object oriented programming [1].

There are many factors, however, that influence *how* an institution teaches the introductory computing sequence and achieves the desired objectives. These factors include the size of the program, the number of students per instructor, the course load of the faculty, the availability of teaching assistants or adjunct instructors, physical resources, students' access to laptops, and a tradition or history of offering labs. A combination of these issues often dictate whether or not a formal, hands-on instructional element (a "lab") is included as part of a course.

In cases where a lab is a formal component of the CS 1 course, the lab is implemented as either a *closed* lab or an *open* lab. A closed lab consists of a required timeframe in which students are all present in a computer lab or classroom that is equipped with the necessary hardware. The students work through guided exercises individually or in groups and are supervised by an instructor or teaching assistant. An open lab is an unconstrained time in which the required resources (e.g. computers) are made available to students. In an open lab, attendance at specific times is optional, and the lab may or may not be monitored by tutors or teaching assistants.

In this paper, we describe an alternative to an open or closed lab for CS 1, which we refer to as a *hybrid* open/closed lab (or simply *hybrid* lab). We introduce and assess the advantages of a hybrid lab and describe an experiment in which two sections of CS 1 were given the same instructional resources; however, one section utilized a combination of lectures and a weekly hybrid lab while the other section was only instructed via lectures.

ITiCSE '17, July 3–5, 2017, Bologna, Italy.

ISBN 978-1-4503-4704-4/17/07…$15.00

DOI: http://dx.doi.org/10.1145/3059009.3059014

2. RELATED WORK

Although the implementation details of labs may vary greatly among institutions, the main idea behind the utilization of labs is that students learn more by doing than by listening to lectures [5]. The use of a lab changes the instructional approach of a "standard" classroom and allows instructors to have a different perspective on student progress and understanding [8].

The literature regarding the use of labs in CS 1 (open or closed) with an explicit assessment of the lab experience vs. a non-lab control group is rather sparse. References [10, 14] postulate that this is due to all sections of a course typically being taught the same for pragmatic reasons. In this case, newly-adopted approaches must be compared to previously taught sections in which multiple variables have changed from the initial offering. As such, there are few simultaneous non-lab baseline sections in which to compare experimental lab sections. In addition, cases in which open labs are directly compared to closed labs are limited. In a study from 1994, which compared open labs against closed labs, Thweat et. al reported that students in a closed laboratory performed better on a comprehensive CS 1 exam than students in open labs [13]. A more recent study has re-

ported that benefits of open labs in computer science include an improved performance on exams, but also resulted in a decreased performance on extended homework assignments [14].

While many studies have not found a significant correlation between labs and student performance, the use of closed labs have been found to have several positive effects. Closed labs have helped students better prepare for online tests [7] and have been noted to have a qualiative effect on student learning [6]. Soh et. al conducted a study on closed CS 1 labs and concluded that motivated students found value in lab activities, and student performance in labs correlates to performance on exams and homework assignments [12]. Furthermore, structured exploration activities in the form of a lab have been shown to help novice programmers overcome common misconceptions regarding computing [9].

Previous work on self-guided labs has proven to have positive results in the introductory computer science sequence. Students who have participated in self-guided labs in a learn-as-you go fashion (abductive learning) have been shown to have decreased failure rates [11].

There are many published works (e.g. [4]) that provide excellent exemplar studies for computer science courses that utilize labs in a variety of areas, and the hands-on approach of education has been widely accepted.

3. HYBRID LABS

3.1 Introduction

The hybrid lab is a set of written instructions, demonstration of techniques, and examples of code presented in a walkthrough format. The lab also consists of several challenges in which students must write original code or answer questions that are based off the concepts introduced in the written components of the lab. Students are presented with the lab two days prior to a class period and are given the option of completing the challenges in lieu of attending a class period (simiar to an open lab). Students must submit working solutions to the challenges in order to be excused from attending the class period. If the student does not have time or is unable to complete the lab, he or she is expected to attend the class session to get the necessary help from the instructor or spend time working through the material (similar to a closed lab). Participation can be strongly encouraged by using attendance at the class periods or completed lab submissions as a component of the final grade.

3.2 Theoretical Advantages of Hybrid Labs

The lecture-based instruction format is not ideal for individual learning. A significant disadvantage of a standard lecture-based instruction format is that material is presented at the same rate for all students in attendance. It is the responsibility of the instructor to present the material at an appropriate pace for most of the students. Unfortunately, lectures will likely be to too fast for some students and too slow for others. The motivating factor behind the hybrid labs is to allow students to practice the components of the course on their own and learn at an individual pace.

The hybrid lab will reward students who are proactive in their studies by allowing them to complete the lab on their own time and not attend a class period. Furthermore, this will reduce the number of students in the classroom on lab days, making the answering of individual questions more manageable for the instructor. In our study, we found approximately 50% of students would typically submit the challenges to the labs prior to class on Fridays.

As the lab on Friday must be attended by students who found the lab too challenging to complete on their own or didn't have sufficient time to complete it prior to class, the instructor has a smaller class size and can give more focused individual attention to the students who need it. Thus, the hybrid labs have multiple advantages: students are able to work at their own pace, and students who need additional help attend a lab session that is smaller and more intimate, in which the instructor can effectively answer individual questions as needed. This experience is in contrast to a regular lecture class period, in which the instructor must present the material at one pace to the entire class.

4. METHODOLOGY

4.1 Hybrid Labs in Practice

In the fall of 2016, we conducted an experiment to assess the effectiveness of the hybrid labs as compared to a completely lecture-based course. The CS 1 course at Drake University, a small private liberal arts college, is typically taught via 50 minute courses on Mondays, Wednesdays, and Fridays over a 15-week semester term. In the hybrid lab approach, the lab occurs on the Friday of each week of the term. The walkthrough lab is supplied to the students on Wednesday afternoon. The students have the choice of completing the lab on their own, submitting solutions to the challenges, and not attending the Friday session. Or, if the student does not have time or is unable to complete the lab, the student must attend the class session on Friday where the instructor is available for individual assistance.

We taught one section (the *lecture* section) as a standard, lecture-only course that met for 50 minutes on Mondays, Wednesdays, and Fridays. The experimental section (the *lab* section) met on Mondays and Wednesdays, and was presented the same material via lecture as the classroom-only section. The lab section, however, was given the hybrid lab on Wednesdays afternoons with the option of completing it prior to the class time on Friday. The lecture section was required to attend the class on Friday, and the material was presented in a traditional lecture-style with frequent breaks to allow for students to complete short in-class exercises on their own laptops. This was consistent with how the course was taught on non-lab days throughout the semester. Both sections were taught by the same instructor.

The CS 1 sections at Drake University are capped at 45 students. In recent years, this course has filled to capacity with several sections. In our case, the size of each of the sections makes a closed lab experience difficult to manage by a single instructor in a classroom that is limited on space.

Students use their own laptops which are not supplied by the university. The software tools used, Python using the Pycharm IDE, are reliable and platform independent which enables all of the students to participate in all of the required activities of the course without the need for a dedicated computer lab.

4.2 Survey Questions

The goals and objectives for CS 1, inspired by the 2013 ACM/IEEE curriculum guidelines [1], include mastery of data structures, if-statements, loops, files, methods, and

Table 1: Survey Questions Measuring Attitudes Related to Computer Science

#	Question
1	Errors generated by computers are random, and when they happen there's not much I can do to understand why.
2	I find the challenge of solving computer science problems motivating.
3	If I want to apply a method used for solving one computer science problem to another problem, the problems must involve very similar situations.
4	Tools and techniques from computer science can be useful in the study of other disciplines (e.g., biology, art, business).
5	I enjoy solving computer science problems.
6	Learning computer science is just about learning how to program in different languages.
7	A significant problem in learning computer science is being able to memorize all the information I need to know.
8	The subject of computer science has little relation to what I experience in the real world.
9	There is usually only one correct approach to solving a computer science problem.
10	I worry that mistakes I make when writing a program may damage my computer.
11	I am interested in learning more about computer science.
12	I prefer a classroom lecture format for learning.
13	I prefer a hands-on lab format for learning.

object oriented programming. We were particularly interested in measuring students' attitudes and abilities related to these objectives prior to the start of the semester so that exposure to programming and computer science concepts could be controlled for in the assessment of the effect of the hybrid labs vs. the lecture-only instructional methodology. These survey questions are designed to assess the exposure and abilities students have acquired before the course. The surveys are repeated after the course to assess the effectiveness of the instructional techniques.

Student attitudes play an important role in shaping how students learn from their experiences [2]. To assess the attitudes and perceptions of computer science, we utilized survey questions from the CAS (Computing Attitudes Survey) [3] on the first day of the course, and again on the last day of the course. The questions ask students to rate their attitudes to various statements on computer science on a Likert-like scale: Strongly Disagree (0), Disagree (1), Neutral (2), Agree (3), and Strongly Agree (4).

In addition to questions designed to measure attitudes related to computer science (questions 1 through 11), we also asked questions related to a preferred learning style (lecture vs. hands-on lab) on questions 12 and 13. The questions presented to the students are listed in Table 1.

In the pre-semester survey, we asked two different kinds of questions related to student's abilities: the first had the students rate on a scale from 0 to 5 how confident they felt they could answer questions related to the objectives for the course (Table 2). The second method asked them to answer basic programming questions as part of the survey (Table 3).

Table 2: Survey Questions for Self-Reported Abilities Related to Computer Science Content

#	Question
14	Write a computer program that uses a variety of data types (e.g. int, float, string).
15	Write a computer program that effectively uses if statements and if-else statements.
16	Write a computer program that uses loops and nested loops.
17	Write a computer program that uses a method or function, passes parameters to the function, and returns a value.
18	Write a computer program that uses a file to read or write information.
19	Write a computer program that creates a class, including accessor and mutator methods, and creates an object of this new class type.
20	Write a computer program that helps solve a problem that you are working on for another course or research area.

Table 3: Coding Questions that Appeared on the Pre-Semester Survey and the Final Exam

#	Question
21	Write the code that will prompt a user for an integer and a float value. Add the two numbers together and print out the result.
22	Write the code that will print out "within range" if the value of the variable score is between the numbers 60 and 100 (including both 60 and 100).
23	Use a loop to print out the integers 1 thru 10 followed by the square of the number.
24	Write a function called double that returns 2 times the input parameter.
25	Assume that there exists a file named temps.txt that consists of a sequence of temperatures measured over the past several weeks. Open the file and determine the maximum temperature in the file. Print this value in a separate file called result.txt.
26	Write the code to define a class called Student. The class should inherit from a class called Person. The Student class should contain a data attribute called gpa. Write the Student class to include an initializer method, accessor method, mutator method.

The self-confidence survey was repeated on the last day of the course, and the programming questions were a part of the final exam.

5. RESULTS

5.1 Assessment of Student Attitudes

The lecture section consisted of 44 participating students whereas the lab section consisted of 41 participating students. In addition to being the entry-level course in the computer science major, the CS 1 course at Drake University satisfies a category in the general-education curriculum. As a result, a majority of the students entered the course

Table 4: Survey Results Measuring Attitudes Related to Computer Science

#	Lecture Pre-Term	Lab Pre-Term	Lecture Post-Term	Lab Post-Term
1	0.95	0.70	0.50	0.44
2	2.73	3.20	2.98	3.47*
3	1.75	1.87	1.64	1.63
4	3.34	3.58	3.41	3.44
5	2.34	2.85	3.00	3.37*
6	1.74	1.78	1.72	1.81
7	2.00	1.60	1.91	1.38
8	1.16	1.18	0.89	0.91
9	0.72	0.58	0.77	0.44
10	1.35	0.95	0.64	0.54
11	3.34	3.56	3.25	3.56*
12	2.32	2.23	2.47	2.35
13	2.89	2.55	2.98	2.84

as novice programmers without any computer science background.

The results of the pre-semester and post-semester survey, which correspond to the questions listed in Table 1, are presented in Table 4. Note that the post-semester responses for the lab section on questions 2 (*I find the challenge of solving computer science problems motivating*), 5 (*I enjoy solving computer science problems*), and 11 (*I am interested in learning more about computer science*), are statistically significant with $p < 0.05$, as indicated (*) in Table 4.

A majority of the differences between the pre-term survey and the post-term survey indicate a favorable influence of the course on the attitudes and beliefs students have regarding computer science. However, the results also indicate potential areas of improvement for our future courses, such as emphasizing the variety application areas for computer science and the ability to implement solutions in a number of different ways. We noticed a slight decrease in agreement from the lab section in question 4 (*Tools and techniques from computer science can be useful in the study of other disciplines*) and a slight increase in agreement for the lecture section in question 9 (*There is usually only one correct approach to solving a computer science problem.*). Further conclusions are discussed in the final section of the paper.

5.2 Assessment of Self-Reported Abilities

On the first day of the course, we asked students to rate their confidence levels on their ability to answer questions related to the primary goals for the course (if-statements, loops, files, methods, object oriented programming) as listed in Table 2. This allowed us to identify several students in the lab section who had previous programming experience. We classified the experienced students with self-reported average scores above 2.0 (on a scale from 0 to 5) on the pre-term survey, which consisted of 7 students in the lab section. All other students were considered novice programmers.

Table 5 indicates the average pre- and post- semester self-evaluation scores for the entire class, including the students with a self-reported knowledge of some of the material. Table 6 displays the average scores of only the novice programmers from both sections, which are demonstrably similar amongst both sections. We can use the results of the novice programmers as a standard baseline for students in both sections to help determine any difference that the hybrid labs make in comparison to the lecture instructional format. All of the post-term scores are statistically significant with $p < 0.01$, indicating that the course had a significant impact on students' confidence regarding the course objectives.

Table 5: Survey Results of Self-Reported Abilities of Entire Class

#	Lecture Pre-Term	Lab Pre-Term	Lecture Post-Term	Lab Post-Term
14	0.57	1.23	4.70	4.72
15	1.05	1.78	4.74	4.70
16	0.43	1.08	4.38	4.56
17	0.45	1.28	4.34	4.43
18	0.50	1.05	3.95	4.07
19	0.18	0.55	4.27	4.37
20	0.39	0.93	3.78	3.70

Table 6: Survey Results of Self-Reported Abilities of Novice Students

#	Lecture Pre-Term	Lab Pre-Term	Lecture Post-Term	Lab Post-Term
14	.057	0.57	4.70	4.73
15	1.05	1.07	4.74	4.76
16	0.43	0.41	4.38	4.58
17	0.45	0.57	4.34	4.39
18	0.50	0.47	3.95	4.03
19	0.18	0.07	4.27	4.39
20	0.39	0.33	3.78	3.67

5.3 Assessment of Abilities

In addition to students' self-assessment of abilities, the pre-term survey also contained questions asking them to demonstrate abilities related to the course objectives (Table 3). The average score for all of the questions in the pre-term survey was below 0.2. This indicates that while some students had a self-reported confidence in the material, many students were not able to (or chose not to) demonstrate it on the pre-semester survey. Each of the questions was repeated on the final exam to measure any difference in specific programming skills between the sections of the course. Table 7 displays the individual scores (out of 5) for each of the programming questions which were a component of the final exam.

The average final exam scores as well as the average individual question scores were very similar between the two sections, regardless of programming experience. The average final exam score for the lecture section was 84.90% whereas the average final exam score for the lab section was 86.64%.

Table 7: Exam Results of Students for Coding Questions

#	LecturePost-Term	LabPost-Term
21	4.83	4.90
22	4.72	4.50
23	4.45	4.63
24	4.00	4.28
25	3.94	4.07
26	4.49	4.65

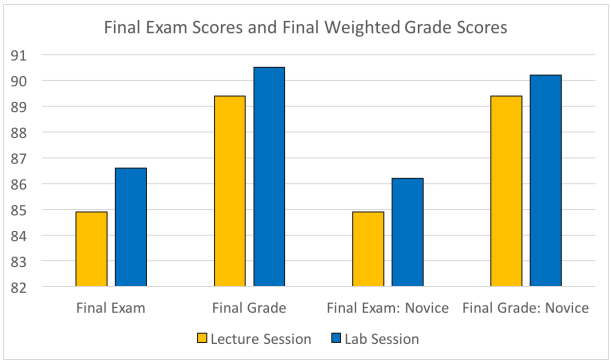

Figure 1: Final exam scores and semester grades for both the lecture and lab sections. The left half of the figure represents the scores for all students in each section. The right half of the figure represents the scores for students who had no prior programming experience.

If we consider only those students with no self-reported previous programming experience, the final differences are also not significant (84.9% for the lecture section and 86.2% for the lab section). The scores for the final grade, which were calculated as a weighted average amongst programming assignments, two midterm exams, and a final exam, were also similar (89.4% for the lecture section and 90.5% for the lab section) as shown in Figure 1. While the lab section outperformed the lecture section by one to two percent on each of the midterm and final exams, no statistically significant differences were found when comparing the two sections and controlling for gender, year of study, or programming experience.

6. CONCLUSIONS

In the self-reported and demonstrated metrics related to skills developed during the course, we did not detect any statistically significant differences between to two sections. This indicates that both the lecture format and the hybrid-lab format are justifiable, acceptable methods for instruction. However, there are several results from the study that are noteworthy.

Result #1: Students appreciate the hybrid lab format.

We can determine to what extent students' perceptions changed by comparing the responses to questions from the pre-semester survey to the post-semester survey. In the case of questions 12 and 13, the extent students agree with the statements "I prefer a classroom lecture format for learning" and "I prefer a hands-on lab format for learning" can indicate the appreciation for the different formats. In our study, both the lecture section and the lab section responses increased for both statements. The lecture section responses for the classroom lecture format increased from 2.32 to 2.47 (increase of 0.15), and for the lab format from 2.89 to 2.98 (increase of 0.09) as shown in Figure 2. The lab section responses for the classroom lecture format increased from 2.23 to 2.35 (increase of 0.12) whereas the responses for the lab format jumped from 2.55 to 2.83 (increase of .28). The

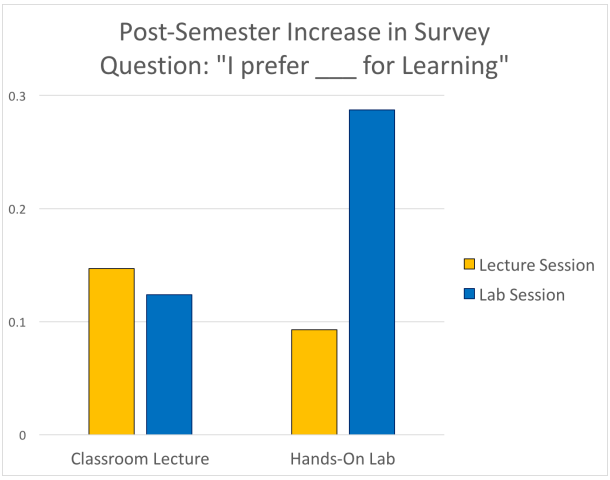

Figure 2: The pre-semester to post-semester increase in the average agreement to the survey question "I prefer (blank) for learning". The relative increase for the lab session's change of preference to the hands-on lab was much greater than all others.

large relative change indicates that students' appreciation for hands-on learning greatly increased over the semester.

Result #2: Students are more engaged with the hybrid lab format.

We obtained statistically significant differences between the sections on three of the post-semester survey questions measuring attitudes towards computer science: "I find the challenge of solving computer science problems motivating.", "I enjoy solving computer science problems.", and "I am interested in learning more about computer science." Each of these statements were more strongly agreed with by the lab section. In particular, the notion of "I am interested in learning more about computer science" drops slightly for the lecture group from the pre-semester to post-semester survey (3.34 to 3.25) whereas the interest remains consistent for the lab section (3.51 to 3.51) between the pre-semester and post-semester surveys.

These questions are related to students' attitudes and engagement regarding the problem solving aspects of the course. Indeed, the lab section was given more freedom to explore and solve problems (and make mistakes) as part of the hybrid lab that the lecture section didn't directly experience. One of the main advantages of the hybrid lab format is the unstructured nature of the lab, which allows for more hands-on exploration and engagement with the material.

Result #3: The labs allowed for a more tailored, individualized instruction for a variety of student abilities.

The most significant drawback with the lecture format of delivering information to a group of students is that not all students learn at the same pace. Undoubtedly, the speed in which the content of the course is being covered will be too fast for some and too slow for others. This dilemma

can be articulated best by a selection of student comments from the end-of-semester course evaluations from the lecture section:

> *"sometimes information was rushed."*
>
> *"class was a bit slow..."*
>
> *"I wish I was in the Friday lab section so I could practice more rather than sit in a lecture."*

The hybrid lab allowed for a tailored, individually-paced instruction. For students who were able to complete the lab challenges at home, they were not forced to sit in a lecture while others caught up. For the students who needed more time or more help, the lab provided an opportunity to get more focused attention from the instructor. The following selected comments from the lab section end-of-semester course evaluation articulates this advantage:

> *"I liked the hands-on, in-class examples and Friday labs. I think they really helped me try to learn it on my own with hands on work."*
>
> *"I really enjoyed how the labs could be done outside of class. The detailed instructions basically counted as a lecture, but I was able to work at my own pace rather than getting distracted while waiting for others."*
>
> *"The combination of lecture and hands-on learning helped to understand both concepts and applications of the material."*

Result #4: The lecture section students learned the material and had a better rapport with the instructor.

As the collected data demonstrates, the lecture section learned the same material to a comparable degree as the students from the lab section. Anecdotally, we noticed that the class dynamics and interactions amongst students and the instructor were more lively during the lecture section than during the lab section. While the lab students have expressed feeling more engaged with the material, the class dynamics were different. An explanation for this result is that the lecture students had more time collectively with the professor and each other in the lecture format. The attendance on Fridays during the lab section was smaller due to students completing the lab at home. As such, the group dynamics were not as consistent, and the lecture section seemed to benefit from a better sense of comradery.

In conclusion, while we found no statistical significance between the abilities of the students of the two sections, the surveys show that students found the hybrid labs to be more engaging and preferred the hybrid labs over lectures as means of instruction. Furthermore, instructors found that the hybrid labs allowed for a more tailored, individual instruction for a variety of student abilities. In the future, we plan to offer all sections of CS 1 using hybrid labs and will consider extending the use of hybrid labs to additional courses.

7. ACKNOWLEDGMENTS

Thank you to the students that took part of the study as well as Kevin Moenkhaus for his assistance in the data analysis.

8. REFERENCES

[1] ACM/IEEE-CS Joint Task Force on Computing Curricula. Computer science curricula 2013. Technical report, ACM Press and IEEE Computer Society Press, 2013.

[2] B. Dorn and A. Elliott Tew. Becoming experts: Measuring attitude development in introductory computer science. In *Proceedings of the 44th ACM Technical Symposium on Computer Science Education (SIGCSE '13)*, pages 183–188, 2013.

[3] B. Dorn and A. E. Tew. Empirical validation and application of the computing attitudes survey. *Computer Science Education*, 25(1):1–36, 2015.

[4] W. Du, K. Jayaraman, and N. B. Gaubatz. Enhancing security education with hands-on laboratory exercises. In *Proceedings of the 5th Annual Symposium on Information Assurance (ASIA'10)*, pages 56–61, 2010.

[5] D. A. Kolb. *Experiential learning: Experience as the source of learning and development*. FT press, 2014.

[6] A. N. Kumar. The effect of closed labs in computer science i: An assessment. *Journal of Computing Sciences in Colleges*, 18(5):40–48, May 2003.

[7] A. N. Kumar. Closed labs in computer science i revisited in the context of online testing. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education (SIGCSE '10)*, pages 539–543, 2010.

[8] D. Laurillard. *Rethinking university teaching: A conversational framework for the effective use of learning technologies*. Routledge, 2013.

[9] R. Lischner. Explorations: Structured labs for first-time programmers. In *Proceedings of the Thirty-second SIGCSE Technical Symposium on Computer Science Education (SIGCSE '01)*, pages 154–158, 2001.

[10] R. P. Mihail and K. Roy. Closed labs in programming courses: A review. In *Proceedings of the International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS)*, pages 104–109, 2016.

[11] A. Radenski. Digital support for abductive learning in introductory computing courses. In *Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education (SIGCSE '07)*, pages 14–18, 2007.

[12] L.-K. Soh, A. Samal, S. Person, G. Nugent, and J. Lang. Analyzing relationships between closed labs and course activities in cs1. In *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education (ITICSE '05)*, pages 183–187, 2005.

[13] M. Thweatt. Csi closed lab vs. open lab experiment. In *Proceedings of the Twenty-fifth SIGCSE Symposium on Computer Science Education (SIGCSE '94)*, pages 80–82, 1994.

[14] N. Titterton, C. Lewis, and M. Clancy. Benefits of lab-centric instruction. *Computer Science Education (Ed. Y. Ben-David Kolikant)*, 20(2):79–102, 2010.