USING CYBERLEARNING ENVIRONMENT TO IMPROVE STUDENT'S LEARNING AND ENGAGEMENT IN INTRODUCTORY COMPUTER PROGRAMMING COURSES

A Dissertation
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Mourya Reddy Narasareddy Gari

In Partial Fulfillment of the Requirements
for the Degree of
DOCTOR OF PHILOSOPHY

Major Program:
Software Engineering

May 2019

Fargo, North Dakota

North Dakota State University
Graduate School

Title

USING CYBERLEARNING ENVIRONMENT TO IMPROVE STUDENT'S LEARNING AND ENGAGEMENT IN INTRODUCTORY COMPUTER PROGRAMMING COURSES

By

Mourya Reddy Narasareddy Gari

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

DOCTOR OF PHILOSOPHY

SUPERVISORY COMMITTEE:

Dr. Gursimran Walia
Chair

Dr. Kendall Nygard

Dr. Oksana Myronovych

Dr. Limin Zhang

Approved:

June 6, 2019
Date

Dr. Kendall Nygard
Department Chair

ABSTRACT

All Computer Science majors are required to take introductory programming (CS1) as a fundamental course which has a high dropout rate. Researchers report that CS1 students lack motivation and need constant resource support. Motivated by these factors, we developed a cyberlearning environment embedded with learning engagement strategies such as Collaborative Learning, Social Interaction and Gamification. The purpose of research is to investigate the impact the cyberlearning environment had on student acquisition of programming concepts. I conducted a series of studies to empirically validate these learning engagement strategies in the context of student learning outcomes. The results of my dissertation have shown that Gamification and Social Interaction when combined or used individually had more positive impact on student learning when compared to that of other learning engagement strategies.

ACKNOWLEDGEMENTS

I would like to convey my earnestness to my primary advisor and mentor Dr. Gursimran Walia for his enthusiastic and encouraging support during the pursuit of my research. He consistently steered me in right direction whenever he thought I needed assistance, without his guidance I would not have been able to complete this dissertation. Not only did he guide me through the dissertation process, but also taught me the overall process of performing quality research. I cannot thank him enough for the supervision, encouragement and inspiration in developing the theory, conducting the experiment, publishing papers, and writing dissertation.

I also express my sincere regards to Dr. Kendall Nygard and Dr. Oksana Myronovych from Computer Science Department and Dr. Limin Zhang from Accounting and Information Systems Department for being on my supervisory committee and providing their feedback.

Finally, I would like to thank my parents and my friends, and a special thanks to my husband, who supported me throughout the entire process, both by keeping me harmonious and helping me putting pieces together. I will be grateful forever for your support.

This work is supported in part by the National Science Foundation under grants DUE-1225742 and DUE-1525112. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDIX FIGURES

1. INTRODUCTION

This section introduces students to the problem statement, research goals and research framework of the dissertation.

1.1. Problem Statement

Programming is at the heart of computer science. Typically, all computer science programs start with an introductory programing course across. In recent years, the demand for programmers and student interest in programming have risen quickly, and introductory programming courses have become increasingly popular. Despite this widespread interest, introductory programming courses are still perceived as intricate and difficult by students [1]. Some of the students are exposed to programming for the first time and within a semester, students enrolled in an introductory programming course are expected to be proficient at transitioning from natural language to machine language, be able to solve problems using some programming language and gain practical program development skills. Achieving expertise in programming generally requires students to understand how programming is used in the problem-solving process and then the design and development of computer programs. Students are introduced to a lot of information in an introductory programming course and this places a high cognitive load on students as they are constructing mental models of programming constructs and develop algorithmic thinking. Combining the cognitive overload, unmanageable work and perceptions about programming, students feel that introductory programming courses are uninteresting and lacking engagement.

It is notable that every student reacts diversely to challenges and stressful circumstances they face during their learning [2]. A standout amongst the most outrageous and unfortunate responses of students in a distressing situation is dropping out of the course. The enormity of the issue becomes concrete when we look at the dropout rates in introductory programming (CS1) courses that numerous organizations report. Some of the researchers have revealed that the dropout rates in computer science are as high as 30 -40% with most of the students leaving after taking CS1 – introductory programming course [5]. Prior research has reported that CS1 students would benefit from more resource support and if they are able to practice and receive timely feedback, and are more actively engaged in their own knowledge construction [7].

To address high dropout rate and lack of motivation amongst CS1 students, CS1 instructors have attempted numerous pedagogical approaches, ranging from special development environments [8], different learning approaches [10], using collaborative learning and modern technologies [9]. It has been reported that using modern technologies that uses gamification can help improve intrinsic motivation of CS1 students. This dissertation hypothesizes that, utilizing gamification coupled with other learning strategies can enhance peer-to-peer interaction, student retention and improve collaboration as students learn programming concepts.

Gamification is generally defined as the use of game elements in a non-game context [21]. The term gamification originated in the digital media industry around 2010 and has been utilized by wide range of industries, for example, art, call center, commerce, education, entertainment, environment, design, government, health, life, marketing, market research, mobile, social good, web sites and work. Gamification is starting to being used to support STEM pedagogy and has resulted in improved motivation and reduced dropout rate [11, 12]. This dissertation is an attempt to use gamification and other learning engagement strategies to support CS1 pedagogy.

To address above issues, I have been part of NSF research initiative that has led to the development of SEP-CyLE Software Engineering and Programming Cyber Learning Environment, (available at https://stem-cyle.cis.fiu.edu/) [13]. SEP-CyLE is a cyber - learning environment that uses various learning and engagement strategies including collaborative learning, gamification and social interaction. The integration of gamification, social interaction and collaborative learning in SEP-CyLE are meant to help students explore content, assess their own competency on a wide range of software engineering and programming topics, gain tool proficiency, and interact with their peers and instructors. SEP-CyLE platform provides essential learning and engagement resources to students, hopefully motivating them to access course relevant information at their own time through the employment of vetted Learning Objects, video tutorials, and self-assessment tools.

1.2. Research Goals

To evaluate how the Learning Engagement strategies (LEs) in SEP-CyLE impacts student learning (in the context of improving student motivation and engagement), the goal of this research is to validate the effective usage of SEP-CyLE in introductory programming courses by manipulating different combination of Les with different student groups. To that end, the primary goal of this dissertation is defined as follows:

> Analyze SEP-CyLE by manipulating the usage of different Learning Engagement strategies and measuring their effects on students' acquisition of programming knowledge and skills in an introductory computer programming course.

The result of this dissertation will also provide feedback to the SEP-CyLE development team on how SEP-CyLE design can be improved or adapted for different student groups and classrooms. A part of this dissertation focus was to gain a comprehensive understanding of gamification in CS education. To that end, a systematic literature review was conducted to identify and classify various GEs in CS education and how they can influence student learning which in turn will improve SEP-CyLE framework and student experience. Formally stated, the secondary goal of the dissertation is defined as follows:

Analyze educational research literature to identify and evaluate gamification elements in terms of their positive and negative effects on student learning of software engineering and programming concepts.

1.3. Research Framework

To achieve the research goals listed in Section 1.2, this dissertation followed a sequence of research activities that had four major focal points and are discussed below:

I. Systematic Literature Review (SLR): The first step was to perform the SLR of software engineering and computer science literature. This resulted in the identification of most commonly used gamification elements in CS/SE education and its impact areas on student performance indicators.

II. Empirical validation of Collaborative learning: This step evaluated the collaborative learning feature of SEP-CyLE in CS1 classroom through a series of studies at NDSU.

III. Empirical validation of different combinations of LEs: Next, different combinations of LEs (Gamification, Collaboration Learning, Social Interaction) were manipulated to measure the most effective way of using SEP-CyLE in CS1 classrooms. This was done by conducting studies at NDSU and then analyzing SEP-CyLE usage at other collaborating institutions to generalize the effective usage of SEP-CyLE.

IV. Analyzing the knowledge deficiencies of students: While SEP-CyLE usage demonstrated significant student learning outcome gains, this dissertation also provides findings on the most common knowledge deficiencies of CS1 students that can guide the development of Learning Objects specifically for those hard-to-comprehend CS1 topics. This dissertation also provides feedback to CS1 instructors on ways to integrate the digital learning objects in CS1 curriculum to ensure that students are learning most relevant topics.

2. LITERATURE REVIEW

This chapter introduced readers to the learning engagement strategies utilized in SEP-CyLE. This section will briefly describe most commonly used learning engagement strategies in CS/SE education and provide evidence on benefits of using them in a cyber-learning environment.

2.1. Gamification

Gamification is a method of *incorporating game elements into non-game contexts in an attempt to extend user-engagement* [21, 22]. Common gamification elements in education includes points, levels, and achievements [21]. Gamification allows participants to replicate their behavior in a non-game context (e.g., completion of a course module) as they would do in a game setting, as noted by Deterding [21]:

> "Indeed, games are full of points, scores, tokens, and so on… the 'fun', the pleasure of these elements does not come from some extrinsic reward value of those elements, but chiefly from the experience of competence they give rise to."

Gamification has the potential to increase user engagement in learning, at the same time providing input or feedback on that learning. Since students typically report a scarcity of engagement when learning introductory computer science course, gamification has the potential to provide an approach to advance student motivation and engagement and also providing feedback on the student's level of competence of the learned material [19].

In educational context, there have been few endeavors to gamify learning activities with two primary purposes. The primary objective is to encourage desired learning behaviors, for example, following software engineering best practices, fostering the participation of students into learning communities, or advancing active participation in peer assessment [24 - 28]. Second goal is to improve engagement of students in learning, for example by the utilization of learning materials like tutorials or digital tools [29 - 30]. The effects on gamification in education show that it's a promising technique if carefully integrated into learning material. A large portion of the studies have shown gamification has an improvement in students understanding and their engagement. Specifically, it was found that leaderboards were most motivating. Badges and progress bars accounted for low interest components. Studies conjointly found a major increase within the performance of the students once utilizing these GEs [31].

Gamification, applied to introductory Computer science pedagogy, might offer students an interesting, effective, novel approach to learn introductory computer science skills and consequently increase attraction and retention for students. The usage of gamification in education is tied to its potential to increase student engagement, and learning [32-34]. This dissertation is focused at using specific GEs (reward points and leaderboard) and evaluating their impact on student learning in an introductory computer programming course.

2.2. Collaborative Learning

Collaborative interactions among students, has shown constructive outcomes across totally different levels of education, starting from young kids doing their school projects like craft works in groups to college students performing on development projects [47-48]. Collaborative learning can be particularly helpful to first year programming students [41]. Collaborative learning is a common academic practice [41 - 43], where students work together on single task and become responsible for each other's learning still as well as their own [44]. Students are interactively engaged within the subject material, perceptive to different problem-solving approaches, keeping one another centered on the task, and learn to verbalize problems [45].

The researchers have observed benefits to students learning when using collaborative learning. Most of the benefits [41, 43] contribute to students' sense of belonging and sense of security, which is most important for first year students who are at risk of leaving the college or discipline [46]. It is hypothesized that collaborative learning will help students at improving their algorithmic critical thinking and course success rates. Prior research shows that it can also have a positive impact on cognitive growth and skill acquisition and transfer. Low retention rates in introductory programming course may be improved with early team experiences since team experiences are coupled to will increase within the sense of belonging, a key retention factor.

2.3. Social Interaction

Interactivity - a central part of Internet-based learning, and Web-based learning and plays an important role in education. Draves [59] contends that it is the quality interaction, but not content, that will determine whether real learning takes place. The importance of social interaction has been underlined by Hiltz (1994) [58] stating that "the social process of developing shared understanding through interaction is the 'natural' way for people to learn'' (p. 22). Whereas Hooper and Hanafin studying the effects of a group composition identified that "achievement differences attributable to group composition correspond to differences in intra-group interaction".

Social interaction helps students to learn from and exchange ideas with other students. This helps in creating a positive environment, where the students would be making connections, consolidating new ideas with students. Hurst [60] found that social interaction helped to enhance the critical and problem-solving skills of their students.

2.4. Digital Learning Objects

A Learning Object (LO) is an assortment of content, practice questions, and assessment based on a single learning objective to support learning. David Wiley describes LOs as small chunks of knowledge that are self-contained and re-usable [50].

To facilitate reuse with a minimum effort, a learning object is "bundled" to incorporate a lesson and an assessment. Digital learning objects assist understanding of more abstract and complex concepts. Digital learning objects are small, modular, discrete units of learning designed for electronic delivery and use.

Regolith and Nelson (1997) suggest that when teachers first gain access to instructional materials, they often break the materials down into their constituent parts. They then reassemble these parts in ways that support their individual instructional goals. This suggests one reason why reusable instructional components, or learning objects, may provide instructional benefits: if instructors received instructional resources as individual components, this initial step of decomposition could be bypassed, potentially increasing the speed and efficiency of instructional development" [51].

In brief, a learning object is any entity, digital or non-digital, which will be used for learning, education or training [53]. In SEP – CyLE, learning objects are limited to the digital type (Digital Learning Object- DLO). DLOs are extremely appealing to SEP-CyLE as a result of they're a lot of smaller chunks of learning content than courses, modules or units; they typically require between 2 to 15 minutes for completion [52]; and also, they are self-contained interactive, reusable and may be aggregative.

The IEEE standards for LOs establish four elements for every LO:

(1) Objective - ought to address just one learning objective;

(2) Content - learning material focuses on it objective;

(3) Practice - Opportunity for learners to review facts and key concepts using: exercises and/or quiz-type self-tests; and

(4) Assessment- the LO ought to assess whether or not the learner has achieved the educational objective

The central to the design of SEP-CyLE are deployment of digital learning objects (prepared and vetted by experts) to promote personalized study and enhance learning. SEP-CyLE contains an array of LOs spanning computer programming and software engineering concepts.

3. SEP-CYLE – SOFTWARE ENGINEERING AND PROGRAMMING CYBER LEARNING ENVIRONMENT

This section provides an introduction of the SEP-CyLE, its main features and how it can be used in programming courses by instructors (or TA's) and their students. This section also describes the interfaces used by the instructor and students to illustrate the usability of the system.

SEP-CyLE is a collaborative effort between Alabama A&M University, Florida International University, Florida Gulf Coast University, Florida A&M University, Georgia Southern University, Miami University - Ohio, and North Dakota State University and Virginia Commonwealth University. SEP- CYLE was initially referred as Web-based Repository of software Testing Tools (WReSTT) was introduced around 2009 and enclosed seven tools, software testing tutorials and links to alternative materials [52, 54]. To support undergraduate computer science education, SEP -CyLE became a cyber learning environment rather than a repository for learning materials. This section first explains the features of the SEP-CyLE and followed by how the LEs are integrated into the SEP- CyLE environment. Later, the usability of the system from the perspective of both the instructor and the student and finally, how it is used in the classroom.

3.1. SEP-CyLE Features

Below figure shows a block diagram of the main features of SEP-CyLE. The features of SEP-CyLE include:

- **Authentication** - provides numerous users with completely different levels of access. These users embody students, instructors and administrators.
- **LESs** - the Learning and engagement strategies supported in SEP-CyLE including Collaborative learning, gamification and social interaction. Additional details on the integration of the LESs are described later within the text.

- **Learning Content** - contains the digital learning objects (DLOs) on numerous software programming and testing topics and tutorials for many testing tools.
- **Administration** - provides the administrator with the power to get numerous reports (number of users, types of users, etc.), configure the system, and facilitate the course management activities.
- **Course Management** - generate student reports for instructors and individual students, assignment of student and instructor roles.

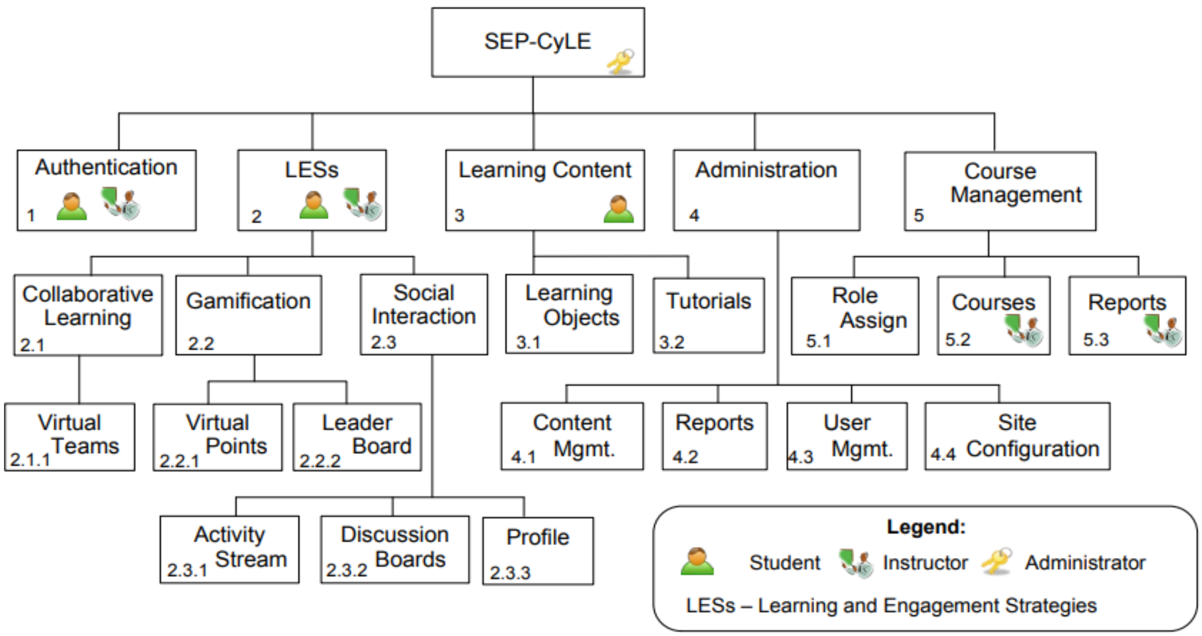


Figure 1: Block Diagram of SEP-CyLE

3.2. Integrating LEs into SEP-CyLE

Currently there 3 LESs integrated into the assorted features of SEP-CyLE, Collaborative learning, gamification and social interaction.

Collaborative learning is targeted around virtual groups and/or the complete category taking part in varied SEP- CyLE activities. Collaborative learning happens as follows:

(1) All members during a virtual team are needed to complete a DLO before every student's profile is refreshed to show the DLO has been finished;

Furthermore, (2) team member post to forum in SEP-CyLE benefiting other students within the class.

Gamification in SEP- CyLE is predicated on a virtual points system. These virtual points are awarded to every student upon successful completion of allotted individual and team activities, or different activities designed to motivate students to communicate with SEP-CyLE. The virtual points are displayed in a leader board, which shows how the other students have performed in the activities and who stands top in the class. Students could also be awarded virtual points for the subsequent activities: individual - finishing DLOs, viewing tutorials, posting to the class forum, change or update their user profile; and Collaborative - finishing DLOs and posting useful learning content that advantages the class.

Social interaction is based on the utilization of various social features amid the learning procedure, either exclusively or as a major aspect of a learning community. These features include: student profiles, message forums, rating of DLOs, rating of tutorials, rating of forum posts, activity streams (individual and class wide).

3.3. Usability of the System

Both the Instructor and student perspective will be discussed in this section. Let us first go through the features of the instructor portal and then will discuss about student portal.

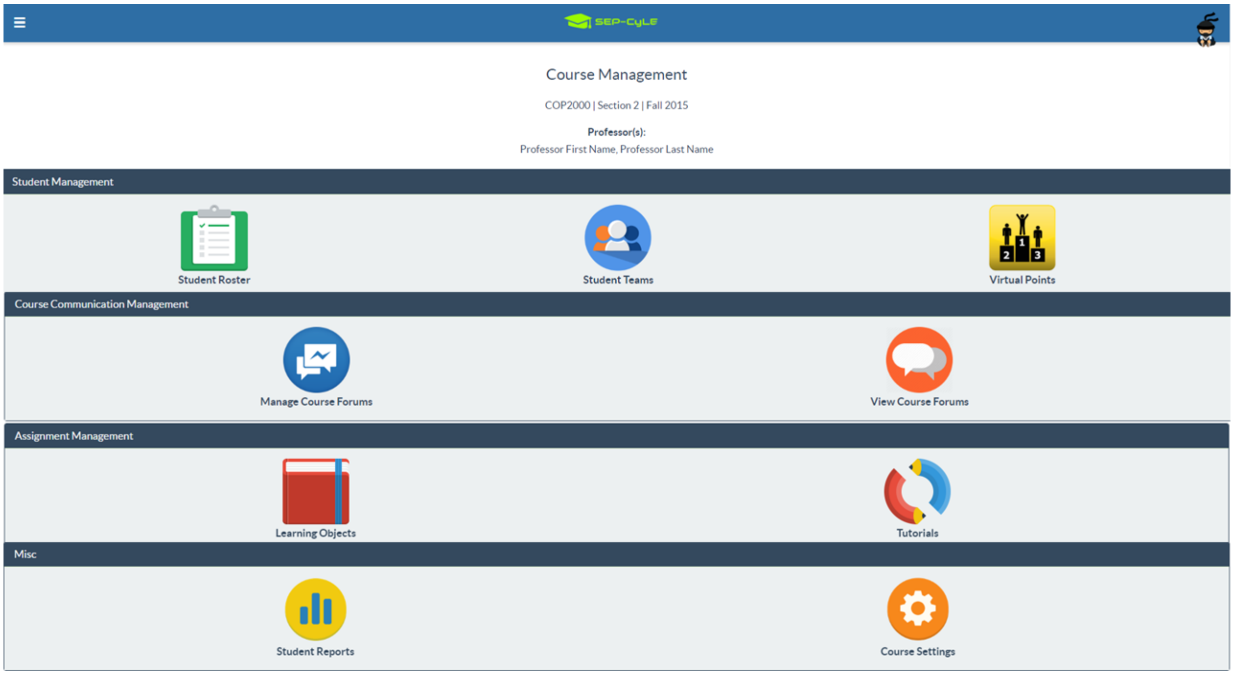


Figure 2: Instructor Home Page

3.3.1. Instructor Portal

Above figure shows how the instructor portal looks like. It has 4 sections:

1. Student Management
2. Course Communication Management
3. Assignment Management
4. Miscellaneous.

3.3.1.1. Student Management

The instructor will be able to upload the student roster by using the student roster feature. He could directly upload the csv file of the entire class or he could add single student at a time. Also, if he would like to remove a student he could remove that selected student by selecting the remove feature.

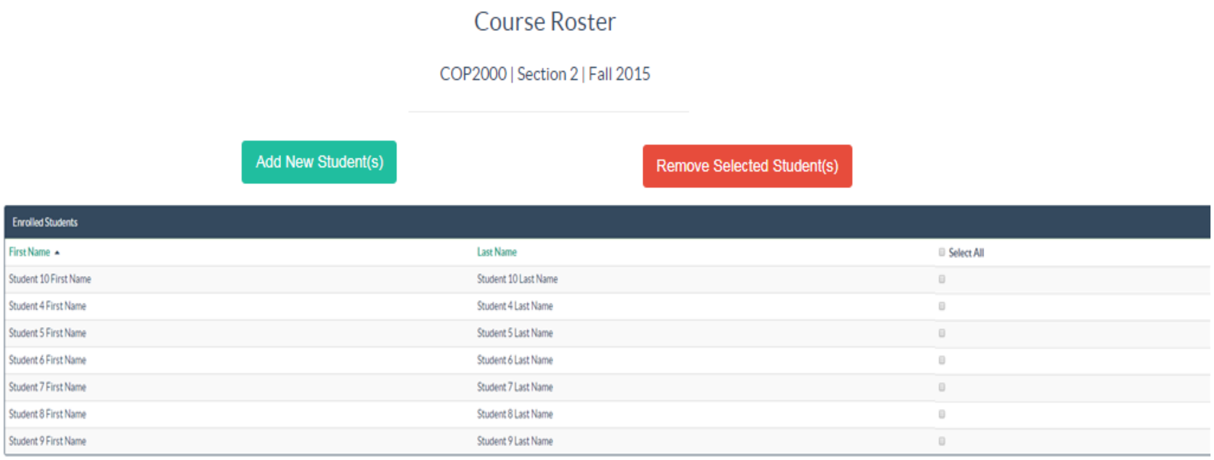


Figure 3: Student Management Page of Instructor

By using the student teams feature, the instructor will be able to assign students randomly to a virtual team and if the instructor wishes to edit or delete the team he could be able to do it with the help of edit or delete options.

Coming to the final feature of student management, which is assignment of virtual points. The instructor will be able to assign the number of virtual points the student would gain when they finish the quiz first, if the team completes a LO first the team would acquire few extra points and similarly if the student starts thread in discussion forum they might be able to score few virtual points which assigned by the instructor.

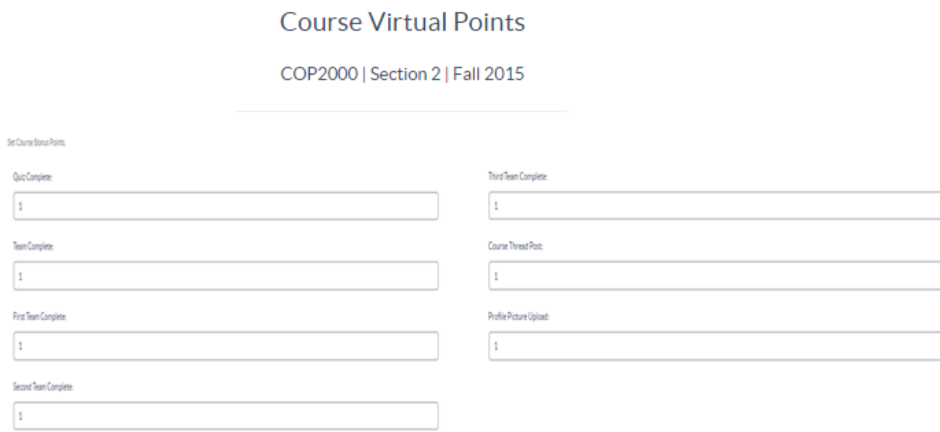


Figure 4: Assignment of Virtual Points Page

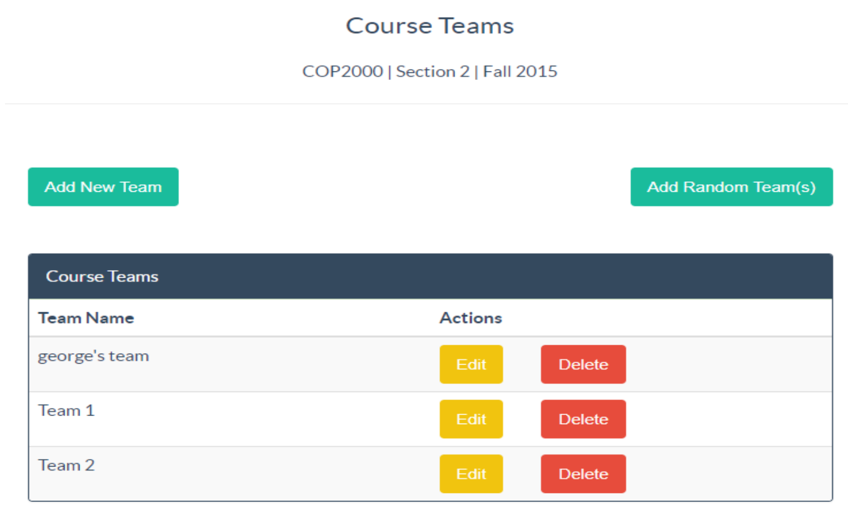


Figure 5: Assignment of Teams Page

3.3.1.2. Course Communication Management

If the instructor would like to add a new course forum or edit the existing course forum or view the existing course forums, then the instructor can use this space to perform any action regarding the course forums. Manage course forum would help the instructor to manage the forums and view forum would help to view the forums that are present for that course.

3.3.1.3. Assignment Management

This is an important section of SEP-CyLE. This where the instructors will assign the Learning objects and tutorials to students. Once the instructor clicks the learning object icon, he will be directed to a page where he could see the leaning objects that are been assigned and also, he would be provided an option to add new assignment.

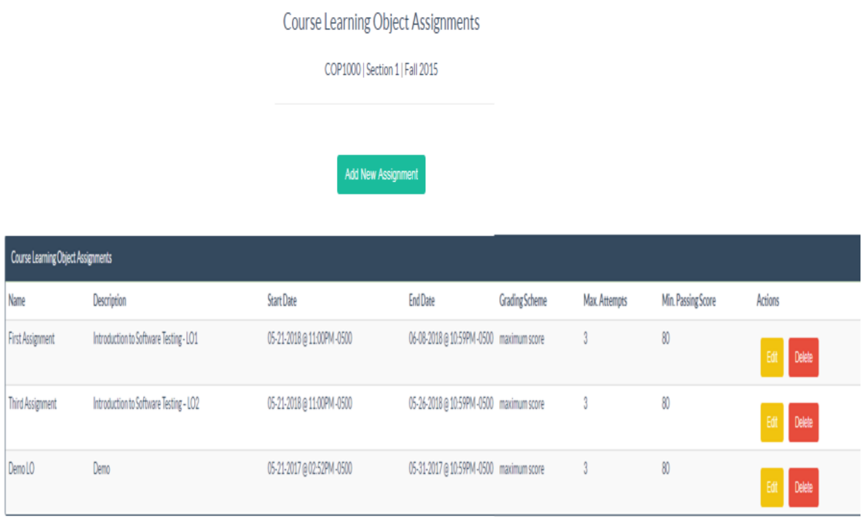


Figure 6: Assignment of LOs Page

The instructor could assign the start date and end date; how many attempts are allowed for students to pass the LO and what would be the minimum passing score.

3.3.1.4. Miscellaneous

This where the system generates student reports for instructors and individual students, assignment of student and instructor roles. The instructor could enable the required Learning Engagement strategies.

Figure 7: Enabling LEs by Instructor

3.3.2. Student Portal

Once the student logins into SEP-CyLE, they could see the courses that they have been enrolled for. On clicking the course, they will be navigated to their course dashboard where they could see the LOs and tutorials that are been assigned, leader board, their team and various course activities that being performed by their classmates.

3.3.2.1. Learning object and tutorial assignment

The student could view the LOs and tutorials that been assigned to the student, once they open the LO, they could see the information related to it followed by the practice quiz and real quiz (for which the instructor had assigned the number of attempts). Tutorial assignment is like LO assignment, only that they need to complete the tutorial, but they would not have quiz attached to it.

3.3.2.2. Course team

If the instructor has assigned the student to any team, they could see it in this section. They could see their team members also and to how many groups they have been assigned, this where the collaborative learning LE comes into picture. Each team would gain few virtual points based on their team member's individual completion of the task.

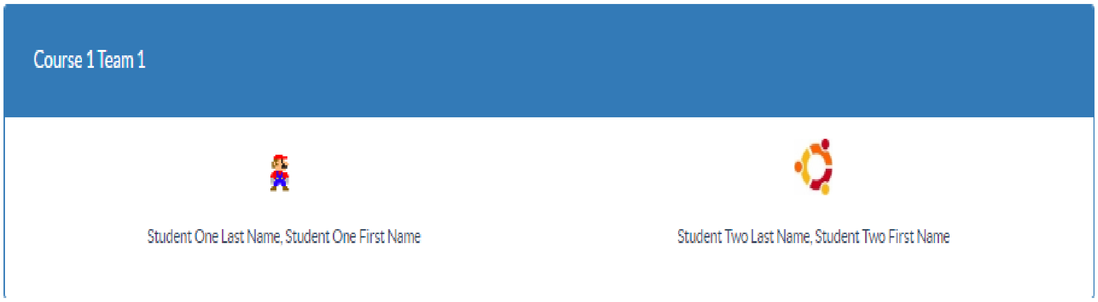


Figure 8: Student Team View

3.3.2.3. Course Leaderboard

Student could see which student is performing better and who is on the top, along with their position in the course.

3.3.2.4. My Course Activity

This provides the students their recent activity in the course, they could also view their report. By viewing their report, they would see how the rest of the class is performing and how he/she is performing for that assignment.

3.3.2.5. Course Activity

This provides the information about what are the other students activities and what are they performing in the course. How many points did the other students have scored and they have many comments in the course forums.

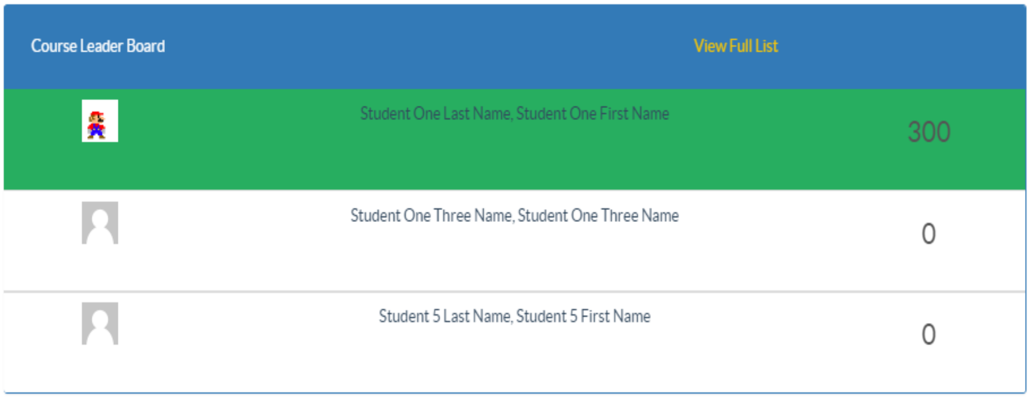


Figure 9: Course Leaderboard View

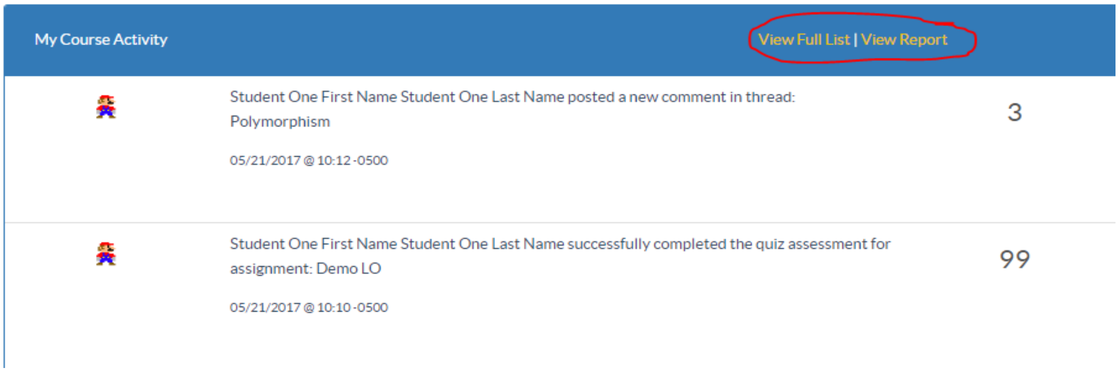


Figure 10: Course Activity View Page

4. RESEARCH WORK

This chapter describes the framework of research. Section 4.1 describes the overview of the studies that were conducted during the dissertation. Section 4.2 describes the Systematic Literature Review conducted to identify and analyze the gamification elements that are used in Computer science and software engineering education which have empirical evidence of positive impact on student learning, Section 4.3 describes the LO's that are used in the studies, Section 4.4 describes the empirical studies conducted to evaluate the different learning engagement strategies to improve student learning. Section 4.5. explains the study that evaluates the impact of different combination of LEs at North Dakota State University. I conducted a Multi institutional study that evaluated the impact of different combinations of LEs, which is described in Section 4.6. Section 4.7 explains the study that was conducted to analyze the knowledge deficiencies of students. Section 4.8 describes the study that evaluated the impact of LOs on Student Learning.

4.1. Overview of Studies Conducted

Below Figure 12 presents an overview of my research work that I completed. This includes the Systematic literature review that was conducted and the control group studies that I have conducted to measure the effect of the learning engagement strategies on student learning.

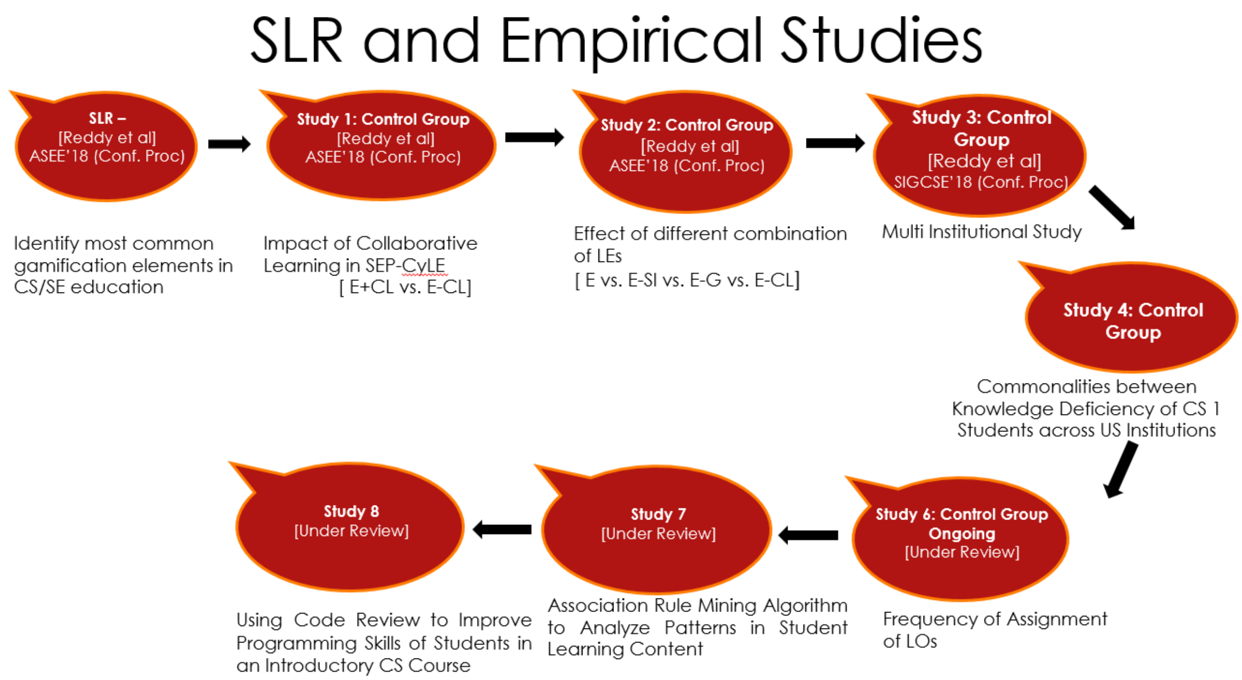


Figure 11: Overview of Research Work Complted

4.2. Systematic Literature Review

This systematic review relies on rules established by Kitchenham and Charters [55] in guidelines for performing Systematic Literature Reviews in Software Engineering. A Systematic literature review could be a means that of assessing and decoding all the accessible analysis applied to a selected research question, area of interest. The subsequent steps were enforced in accordance with the rules for a systematic literature review established by Kitchenham and Charters [55]. The main goal of the survey literature review is formally stated as [56]

> Analyze literature for understanding and evaluating gamification elements with respect to their effects on student learning in the context of software engineering and computer science courses.

First, we formulated the research questions to achieve the main goal. Then selected the source and developed a search string, following that we have included few criteria's for including and excluding the papers that are resulted from the study. Then, we discussed about the study execution, data extraction and finally jotted the results from the search.

Let us start from the first step, which is formulation of the research questions. They are listed as follows:

RQ1: What are the most commonly used gamification elements in CS and SE courses?

RQ2: What is the evidence for the benefits of these gamification elements on student learning?

RQ3: How can answers to RQ1 and RQ2 be incorporated into the design of cyber learning environments?

The second step is source selection and search string development; the databases that are searched are ACM Digital Library, IEEE Xplore, ProQuest, and Web of Science. In addition, the following conference and journal proceedings were also reviewed to ensure that all the relevant results were included: SIGCSE, CSEE&T, ICER, ITICSE, TOCE, Computers and Education.

Coming to the search string that we have used, we combined some relevant keywords and synonyms which developed the following search string

(game or gamification) OR (elements) AND (effect or impact) AND (education or learning or student or course or computer science or software engineering or lecture).

The results included the papers that are from most recent times which are dated 2010 and above. Also, no more than 500 results considered because they were more relevant to the research.

Inclusion and exclusion criteria: All the papers that have been included have answered RQ1 and RQ2; papers that had empirical evidence; papers that in English and finally, that can be applied to CS/SE courses. Whereas the exclusion criteria included the papers that are not in English; papers that have not focused on student learning; papers that don't have empirical results; and papers that talk about designing games. Next step in systematic literature review is study execution. We have drilled our papers count to 16. This happened as follows first we found 5450 papers at first, then Application of the inclusion, exclusion criteria got the paper count to 876. This was further reduced to 154 papers based on review of the abstracts. Each of the remaining papers was read in its entirety. After reading each of the selected papers, only 16 remained and the list of the selected papers is provided in Table 2.

Table 1: List of Included Publications

ID	Title	Reference
P1	The Study of Gamification Application Architecture for Programming Language Course	[63]
P2	Applying gamification in the context of knowledge management	[64]
P3	Game2Learn: Improving the motivation of CS1 students	[65]
P4	Gamification in Educational Software Development	[66]
P5	Improving Participation and Learning with Gamification	[67]
P6	A Playful Game Changer: Fostering Student Retention in Online Education with Social Gamification	[68]
P7	On the Role of Gamification and Localization in an Open Online Learning Environment: Javala Experiences	[69]
P8	Does Gamification Work? — A Literature Review of Empirical Studies on Gamification	[62]
P9	TrAcademic: Experiences With Gamified Practical Sessions for a CS1 Course	[70]
P10	How (not) to Introduce Badges to Online Exercises	[71]
P11	Motivating Skill-Based Promotion with Badges	[72]
P12	Increasing Students' Awareness of Their Behavior in Online Learning Environments with Visualizations and Achievement Badges	[73]
P13	The Effect of Virtual Achievements on Student Engagement	[74]
P14	Gamification for Engaging Computer Science Students in Learning Activities: A Case Study	[75]
P15	A Gamified Mobile Application for Engaging New Students at University Orientation	[76]
P16	Teaching Software Engineering Through Game Design	[77]

Coming to the data extraction part, since we have gotten the important data out of all data consistently, we have developed a data extraction form. We extracted the following data from each paper to ensure that same information was extracted from each paper and to avoid any researcher bias.

Table 2: Data Extraction Template

Field	Description
Study ID	Unique identifier for the paper (same as the reference number)
Bibliographic data	Author, year, title, source
Study Type	Journal/conference
Study aims	The aims or goals of the primary study
Study type	The type of research performed (e.g. case study, controlled experiment)
Gamification elements	The gamification element(s) identified by the study
Focus area	CS or SE course
Evidence/Results	Evidence regarding the usefulness of gamification elements for student learning
Concepts	The key concepts or major ideas in the primary studies
Higher-order interpretations	Limitations, guidelines or any additional information

Finally, the results have given the information about commonly used GEs, evidence of benefit of these elements and how could we implement these in SEP-CyLE and improve the design of SEP-CyLE. Gamification is usually outlined as the use of gamification elements in non-game context, that helps in increasing the student's engagement. As there are several GEs utilized in previous works, this focuses on the list of GEs that are utilized in within the areas of CS and SE education. Supported our literature review, we tend to know eleven most commonly used GEs within the educational contexts. we've got provided brief descriptions of every GE below.

Points (Pt): These are the rewards that are assigned to the students for the completion of a specified task. The point system is utilized as measure of success or progress or an accomplishment

Badges (B): These are depicted as a token of accomplishment. These are also rewards that are assigned to students after they complete a selected task or after they reach a goal.

Leaderboards (Lb): These produce a competitive atmosphere among the students. A leaderboard is that the board that shows the positioning of the students based on the points that they have earned. The show is often created on an individual basis for every score or for overall scores that are earned by the students.

Avatars (A): These are the characters that fill in as a virtual illustration of the player. Because the player progresses within the game, they will get additional things, which permits the avatar to evolve over the time.

Storylines (S): Storylines confer with the employment of a narrative or a subject within the game that the user is taking part in. Storyline provides extra external motivation as a result of students become endowed within the narrative and need to assist progress it. The theme provides additional extraneous motivation by relating the learning modules to one thing that students realize or discover as more charming.

Visualizations (V): This is often special and extraordinary GE, wherever the student's position are depicted in the form of dot and this dot offers them a mental image that if they progress within the same pace it'd offer an inspiration of what would the tip results of the student be.

Progress bars (Pb): This shows the students regarding their progress in reaching a goal.

Punishment (Pu): This GE is employed to award if students commit a blunder or a mistake.

Levels (Le): The game is partitioned into completely different levels. At the initial stages there would be less effort needed to complete it, as the level progresses it becomes tougher for the player to complete.

Challenges (C): This might offer the user a challenge that they ought to complete it. On finishing the challenge, the user would be rewarded. Every level will have one or additional range of challenges.

Feedback (F): Feedback helps in avoiding students obtaining lost or confused. The frequency of these GEs appeared is shown in the below figure.

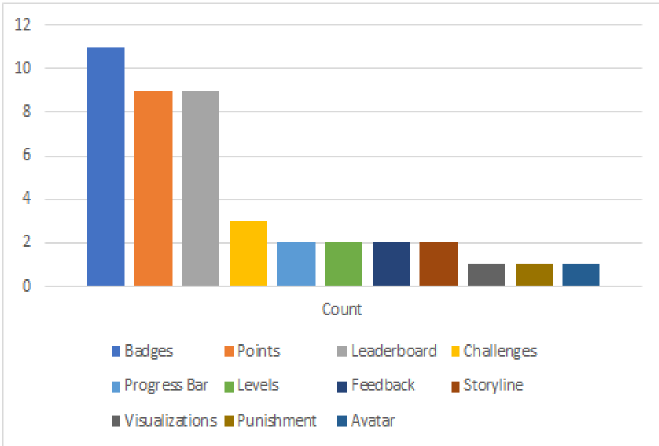


Figure 12: Most Used GEs in Computer Science Education

Whereas the empirical evidence for the benefit of using GEs which answers the second research question is given below [56]:

Table 3: Empirical Evidence of Usefulness of Game Elements in CS Education

ID	Student Learning Focus	GEs	Results of the Study
P1	User participation, student engagement, and student achievements.	Pt, L	There was a positive impact on learning effectiveness. When the gamification elements were omitted from the application, there was decline in student achievements, decrease in user participation and user engagement.
P2	Participation of software development teams in knowledge construction processes.	Pt, B	The results evidenced an improvement in three areas: participation, collaboration and contribution). It was observed that each team member took an interest in each of the activities. The results showed 100% achievement in participation.
P3	Teaching introductory programming through game design to improve engagement, motivation, and learning.	Pt, Pu	The student interviews and observations provided strong evidence that Game2Learn could be successful at enhancing student engagement and motivation. The results were positive only when students understood the game design concepts. Conversely, the performance was poor when students attempted tasks without reading the instructions.
P4	Students use of tools when gamification elements are included in the tools	Pt, Lb	The results indicate an increase in the points earned by the group where the students can compare themselves with others.
P5	Effect of gamification on students in terms of grades, engagement, and motivation	Pt, B, Lb, L, C	There was increase in the lecture downloads from 1.5 to 3 times. Compared to non-gamified years, the number of posts per student grew significantly 4 to 6 times on the first gamified year and 6 to 10 times in second gamified year. They also observed higher minimum grades during gamified years.
P6	Do social gamification elements amplify possible positive effects	Pt, B, Lb, A	Using game and social conditions resulted in higher average retention periods. Students in the game and social conditions group had higher test scores than students in control group.
P7	Usage patterns for online learning systems when using gamification elements	Pt, B, Lb	When the gamification is used, users spent more time and completed more exercises. The total time that the student spent was significantly smaller when gamification was turned off.
P8	The contexts of gamification, dependent variables, and independent variables outcomes from gamification.	Pt, B, Lb, S, V, L, C, F	The major finding of the paper is that most of the gamification elements acted as motivational affordances, but reward points, leaderboards and badges were the most influential game design elements.
P9	TrAcademic was used to gamify the practical sessions in introductory CS course.	Pt, Lb	There was a 500% increase in the attendance. TA's strongly agreed that the practical sessions were helpful to students.

The answer for the third research question is that the incorporation of the GEs into SEP-CyLE. Where SEP-CyLE already incorporates top 2 gamification elements which are leaderboard and points. From the results badges are the undeniable decision for expansion, as they have shown positive impact on students learning in most of the cases. While it's clear that sure game elements like badges, progress bar (and within the context that they have to be used) will facilitate to encourage students as such, incorporation of any gamification component to SEP-CyLE would want to be through empirical observation tested. Which has been tested by conducting case studies which are discussed in the coming sections

4.3. Learning Objects that are Used in the Studies

Initially SEP-CyLE had only the learning objects (Los) that focused on testing concepts. When we wanted to conducted study to see the impact of learning engagement strategies, the instructors of the course suggested that the existing Learning Objects were not enough and not so relevant to their syllabus. So, based on the suggestions from the instructors I had developed few learning objects and we have used the same LOs in the case studies.

At the beginning of developing the LOs gathering the information was not tough. There was a lot of valuable information via textbooks, papers and internet. So, collecting the information was never an issue. But, it was cutting it right down to be simply browse on a webpage for the students utilizing them and to not overload students. This writing was done to create the LO informative enough and to provide a full understanding of the subject. Students that are accessing the location will have any level of understanding or exposure of programming concepts. This had to be unbroken in mind, so a novice to somebody with some information will learn something from the LO. The last thing was making the LO to be esthetically pleasing for the user. This was consummated by making page breaks and adding visuals inside the pages. The first LO that was developed was on Debugging. In the year 2016, I attended a workshop conducted by SEP-CyLE team called LESSEP 2016, which was the first workshop on Using Learning and Engagement Strategies in Software Engineering and Programming Courses, where I learnt how to create LOs.

There has been a tremendous support and valuable inputs from my advisor Dr. Gursimran Walia and the instructors of the course Alex Radermacher and Dr. Otto Borchert and helped me throughout this journey. The template of how an LO looks like is presented in Appendix A. The List of LO's that I have developed are listed below:

Arrays: This LO provides the basic information on Arrays like declaring and initializing an Array like declaring and initializing an Array.

Advanced Debugging LO: This provides a little in-depth detail like Step into expression, Adding Watches and Conditional Breakpoints to the code.

Debugging LO: This Learning Object provides an idea on different types of errors and gain an understanding on the process in which the students can go about debugging them.

Designing Graphical User Interfaces in NetBeans: This LO provides a summary of NetBeans GUI Builder and creating simple GUI programs inside NetBeans.

Fixing Programming Errors in NetBeans: This LO is focused on identifying and fixing diverse kinds of programming errors when using NetBeans IDE.

Pair Programming LO: This LO provides the details about pair programming, how it is conducted and its Pro's and Con's.

SOLID Principles LO: This LO provides the details about the design principles that help in making the software more understandable, flexible and maintainable.

4.4. Study 1: Impact of Collaborative LEs

The study was designed to analyze the impact of SEP-CyLE on undergraduate student's acquisition of programming concepts in an introductory programming course (CS1) at North Dakota State University (NDSU). The study was conducted across the two different sections of CS! Taught by same instructor. The experimental group had all the features of SEP-CyLE which include social networking, gamification and team collaboration. Whereas the control group had all the features enabled except team collaboration. Which is, the control group excluded the student teams and team based virtual points. This has done to isolate the particular impact of team collaboration of SEP-CyLE on student performance

4.4.1. Study 1: Study Goal

The main goal of the study is to investigate the impact collaborative learning engagement strategy of SEP-CyLE on students learning of software programming concepts and their performance in an introductory programming course.

4.4.2. Study 1: Research Questions and Variables

This section describes the Research Questions (RQ's) and the variables used in the study.

Table 4: Study 1 - Variables

Independent Variables	**Description**
Team collaboration points	These are the virtual points that are received by the students on completion of a group task
Dependent Variables	**Description**
Virtual Points	These are the reward points that are earned by the for performing activities within SEP-CyLE
Total grade	This is the course grade received by students at the end of semester, based on their performance throughout the course
Number of LO's the students attempted	This is the count of number of LOs that each student attempted
Number of LO's completed	This is the number of LO's that each student passed with at least 80% of the questions correctly answered.

RQ: Does the inclusion or exclusion of collaborative learning in SEP-CyLE impact the student's acquisition of programming concepts and performance in introductory programming course.

4.4.3. Study 1: Participating Subjects

The case study was conducted in an introductory programming course which had two sections. One section had 46 students and the other had 47 students (93 students in total). 35 out of 93 students participated in the study.

4.4.4. Study 1: Study Procedure

The study procedure had three major steps:

Step 1 -- PreTest: This was conducted at the beginning of the semester to assess the baseline knowledge of students on understanding of programming concepts.

Step 2 -- Introducing SEP-CyLE: Later to pretest, the students were trained on how to use SEP-CyLE. This includes introduction of SEP-CyLE features (viewing tutorials, virtual points, taking quizzes, posting and viewing the comment in the course forum).

Students earn virtual points for successful completion of LO and can earn team based virtual points by completing a task as a team.

Step 3-- Posttest: At the end of the semester, the students are asked to participate in the posttest. This instrument uses the same set of questions as of pretest and this helps us to assess improvement in student's knowledge related to software programming concepts. We also interviewed the course instructor to gain the insights into the usability of SEP-CyLE and to improve future usage.

4.4.5. Study 1: Data Capture

Virtual points earned by each student at the end of semester, student activity on SEP-CyLE for e.g., time spent on SEP-CyLE for different activities, LOs attempted, LOs passed. These helped in finding the correlations between the usefulness of SEP-CyLE features and student course performance, i.e., course grades.

The main source of data is the pre and post test results, which helped in evaluating the improvement in students understanding of programming concepts. Further, to get more understanding on their performance on specific programming concepts, the pre/posttest was divided into 8 specific programming concepts, which is provided in the below table.

Table 5: Categories of Questions

Categories	**Question Numbers**
Memory Management	1, 2,12 and 15
Basic Programming Concepts	3,4,8,9 and 22
Arrays	5,11
Basic Operators	6 and 21
Methods	7
Pointers	10
OOPs Concepts	13, 14 and 16
Software Testing Concepts	17, 18, 19 and 20

4.4.6. Study 1: Summary of Results and Analysis

This section describes the results that have been found in this study.

4.4.6.1. Pre-Test vs. Post Test Results

To analyze the impact of SEP-CyLE on student learning, average score of pretest was compared against the average score of posttest for control group and experiment group respectively. There has been a higher increase in the posttest scores for the control group (this is the section that did not included team based virtual points) when compared to the posttest scores of experiment group. A paired t-test has been performed and the results indicated that there is statistical significance ($p< 0.001$) increase for both the groups and this is shown in Figure 14.

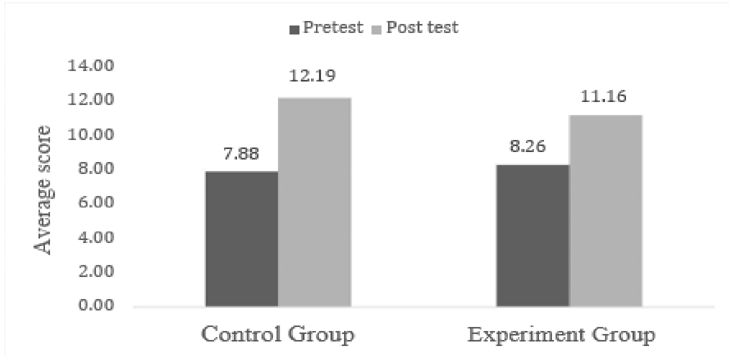


Figure 13: Average Pre/Post Test Value of Control Group & Experiment Group

As we described earlier, that we wanted to see the impact of SEP-CyLE on specific programming concepts and hence divided the pre/ posttest questions into eight categories. Pre- and post - test for each of these eight categories is compared and the results are shown in Figure 15 below.

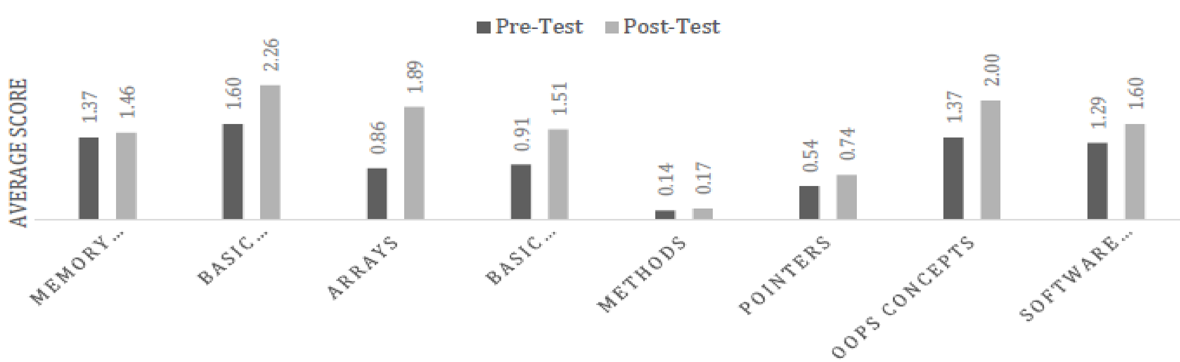


Figure 14:Average Pre/Post Test Value for all Programming Concept Categories

All categories have shown significant improvement ($p<0.001$) except for the Methods category (p- value is 0.023). Based on the instructor feedback, once SEP-CyLE is populated with increased Learning Objects and mainly for the concepts that the students continue to be lacking during posttest, their insight into programming concepts and utilization of SEP-CyLE is expected to increase in future.

4.4.6.2. Study 1: Correlation Results Between SEP-CyLE Activity and Course Performance

We conducted correlation tests which is shown in Table 7, to analyze the relationship between the SEP-CyLE metrics (time spent on SEP-CyLE, number of virtual points earned, number of LOs completed, and number of LOs attempted) against their course performance (i.e., their end of semester grades). Based on the amount of time spent on SEP-CyLE, number of LOs attempted (or completed), and virtual points earned, student engagement has been measured.

Table 6: Correlation Results

	Control Group		Experiment Group	
	p-value	Pearson Correlation	p-value	Pearson Correlation
# of LO's attempted vs. Course performance	.067	.469	.949	.016
# of LO's Completed vs. Course performance	.071	.463	.827	.055
Virtual points earned vs. Course performance	.157	.371	.164	.333
Time spent on SEP-CyLE vs. Course performance	.825	.060	.129	.372

Based on the results the students spent significantly more time on SEP-CyLE when collaborative learning was disabled which is control group section when in comparison to the experiment group. This was incompatible to our expectations. Interestingly, more time spent on SEP-CyLE did not had any impact on virtual points (i.e., both groups fared equally well on virtual point portions).

Based on the instructor's feedback, this could be because of the team had been formed randomly, which constrained the engagement and that might have resulted with less team-based collaboration within SEP-CyLE.

The results from this study indicate effectiveness of usage of SEP-CyLE with gamification features to teach programming concepts and methodologies in introductory programming course. An interesting result that was found in this study is that the collaboration features did not result in a significant difference in the performance of the students exposed to the team collaboration features. In addition to that, the collaborative tasks in SEP-CyLE (i.e., team virtual points) did not require interaction among team members as long as each member of team completed an LO individually.

4.5. Study 2: Impact of Different Combination of LEs

The study was designed to analyze the impact of various learning commitment methodologies (LEs) (e.g., Social Interaction – SI; Gamification – G; and Collaborative Learning – CL) of SEP-CyLE on student learning and understanding of programming concepts and testing techniques in an introductory programming course at NDSU. This study has been conducted across four different sections (taught by two instructors), where each section had a different experimental condition. Section 1 had everything enabled (E); Section 2 had everything except social interaction (E-SI); Section 3 had everything except collaborative learning (E-CL); Section 4 had everything except gamification (E-G).

At the end, we have also conducted user survey to get feedback from students on SEP-CyLE and how it helped them improve their success on assignments or understanding of course concepts.

4.5.1. Study 2: Study Goal

This study has two goals. The main goal of the study is to investigate the impact of combinations of various learning engagement strategies on students' acquisition of programming concepts and methodologies and the second goal is to evaluate the overall satisfaction of students with the features of SEP-CyLE and its usability in the programming course.

4.5.2. Study 2: Research Question and Variables

This case study investigated the following research question:

RQ: Does the various combinations of learning engagement strategies within SEP-CyLE impact the student's acquisition of software programming and testing conceptual knowledge, tools and techniques in an introductory computer programming course.

The independent and dependent variables of this study are provided below:

Table 7: Dependent and Independent Variables

Dependent Variables	***Description***
End of semester grades	This is the course grade received by students at the end of semester, based on their performance throughout the course
Independent Variables	***Description***
Virtual Points	These are the reward points that are earned by the for performing activities within SEP-CyLE
Total time spent on SEP-CyLE	This is amount of time spent by each student in the whole semester
Number of LO's the students attempted	This is the count of number of LOs that each student attempted
Number of LO's passed	This is the number of LO's that each student passed with at least 80% of the questions correctly answered.

4.5.3. Study 2: Participating Subjects

As described in the beginning, this case study was conducted across four sections of an introductory programming course (CS1), which is taught by two instructors (each instructor taught two sections using the same lecture material, assignments, exams, and schedule). A total of 145 students were enrolled across all four sections of the course. 68 out of 145 students elected to participate in the study.

4.5.4. Study 2: Study Procedure

This study included five main steps:

Step 1 – Pretest: Before introducing SEP-CyLE to students, at the very beginning of the semester a pretest was conducted to assess students' baseline knowledge of programming concepts. This pretest consisted of 21 questions.

Step 2 – Introducing SEP-CyLE: The students were then introduced and trained on how to use SEP- CyLE. They were taught how to browse video tutorials and learning objects and track their performance, how to change their profile pictures and interact with their peers in the cyber learning environment, and how to post to the discussion forums. Each section was trained separately depending upon the LEs that were enabled for them.

Step 3 – Assigning LO's to Students: Every week students were assigned a LO to be completed as part of their course. The LO's that are used in this case study are listed (Table 9). These LO's had been previously created by the SEP-CyLE development team or were developed by researchers specifically for use in this course.

Step 4 – Posttest: At the end of the semester, the students were retested on the knowledge of their proficiency on software programming and testing concepts, methodologies and tools using the posttest instrument which has the same set of questions as that of the pretest instrument. This was done for all four sections to understand the influence of different LEs on improvement in their conceptual knowledge.

Step 5 – Survey: At the end of the study, students had been asked to fill out a survey to evaluate the usability and effectiveness of SEP-CyLE in an introductory programming course. We furthermore collected feedback on enhancing SEP-CyLE for future courses.

Table 8: List of LO's Used in the Study

LO Name	Description	Week #
Introduction to NetBeans	An introduction to the NetBeans IDE and some of its basic functionality	1
Advanced NetBeans	A look at some of NetBeans advanced features and functionality	2
Using Subversion with NetBeans	A coverage of basic SVN commands including conflict resolution	3
Introduction to Equivalence Classes	An introduction to the concept of equivalence classes and how they relate to unit testing.	4
Equivalence Classes	An explanation of creating equivalence classes for a Java method	5
Equivalence Classes for Methods with Multiple Parameters.	An explanation of the process for creating equivalence classes for methods with more than one parameter	6
Using Debuggers	An introduction to using the debugger in NetBeans	7
White Box Testing	An overview of white box testing	8
Advanced Debugging	An overview of advanced techniques for using NetBeans debugger.	9
LA Array LO	A learning object related to arrays.	10
SOLID Design Principles	A learning object related to good software design principles	11

4.5.5. Study 2: Data Capture

The main data source of this study are the student responses to the pre- and posttest, which was described in the above section. This pre/posttest consists of 21 questions and the student would get one point for each question answered correctly. Another data source is the survey responses from the students. The students responded to the questions using a 5-point Likert scale, where 1 =

Strongly Disagree, 2 = Disagree, 3 = Neither Agree nor Disagree, 4 = Agree and 5 = Strongly Agree. This Likert scale data has been treated as interval scales.

Furthermore, SEP-CyLE data was collected SEP-CyLE data for each student in terms of number of LO's attempted, number of LO's passed, time spent on SEP-CyLE, and number of virtual reward points earned. In addition to that, student's end-of-semester grade to correlate against SEP-CyLE metrics.

4.5.6. Study 2: Summary of Results and Analysis

In this section, initially the pre- and posttest results were analyzed and later that the correlation analysis and survey results are described.

4.5.6.1. Pre-Test vs. Post Test Results

A comparison between the average pre- and post-test scores of each section is performed. This is to evaluate the impact of different combinations of LEs on-student's acquisition of software programming concepts.

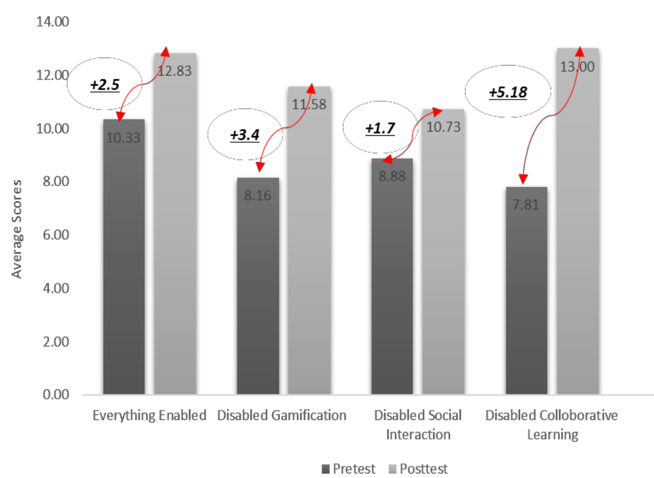


Figure 15: Average Pretest vs Posttest Scores Across all Sections

Clearly, from the above results (Figure 16), there is an increase from pretest to posttest across all the sections. However, there was a highest increase for the section that had collaborative learning disabled.

This result is similar and consistent with the previous study result (Study 1 result). To put in simple words, Gamification and Social interaction played the larger roles in affecting and influencing student's engagement and mastering in an introductory computer programming course.

4.5.6.2. Study 2: Correlation Results between SEP-CyLE Activity and Course Performance

Second set of results is the correlation results. This correlation evaluated the relationship between different variables to evaluate the impact of SEP-CyLE usage metrics on the student's final grade in different sections.

Table 9: Correlation between SEP-CyLE Usage Metrics vs. Course Grade

	E	E-G [without Gamification]	E-SI [without Social Interaction]	E-CL [without Collaboration]
# of LO's attempted vs. Course Grade	Coeff. = 0.762 P-value = 0.07	Coeff. = 0.14 P-value = 0.55	Coeff. = 0.306 P-value = 0.13	Coeff. = 0.013 P-value = 0.5
# of LO's Passed vs. Course Grade	Coeff. = 0.699 P-value = 0.12	Coeff. = 0.145 P-value = 0.55	Coeff. = 0.303 P-value = 0.13	Coeff. = 0.013 P-value = 0.96
Time Spent vs. Course Grade	Coeff. = 0.78 P-value = 0.06	Coeff. = 0.418 P-value = 0.4	Coeff. = 0.32 P-value = 0.113	Coeff. = -0.24 P-value = 0.36
Virtual points earned vs. Course Grade	Coeff. = 0.792 P-value = 0.06	Coeff. = 0.5 P-value = 0.8	Coeff. = 0.326 P-value = 0.104	Coeff. = -0.09 P-value = 0.73

Table 10 provides the correlation results and the strength between the variables. The main result is that, when everything is enables the utilization of SEP- CyLE had a strong positive correlation (significant at p-value <0.1) with the course performance. The results indicated that promise of using SEP-CyLE with gamification and social interaction LEs provided positive results when compared to that of the other combinations. Student comprehension of testing ideas, their insight into programming and testing techniques and their expertise of tool usage showed a significant improvement when a combination of gamification and social interaction learning strategies are enabled in SEP-CyLE. Also, collaborative learning strategy did not have much influence on the students which was an unexpected result.

4.5.6.3. Study 2: Survey Results

With regards to student feedback, there was positive feedback of SEP-CyLE when it comes to overall satisfaction with the website, its simplicity and clearness of data/information as shown in Figure. Students evaluated their reactions on a 5-point Likert scale [1= Strongly Disagree; 2= Disagree; 3= Neither Agree nor Disagree; 4= Agree; 5= Strongly Agree].

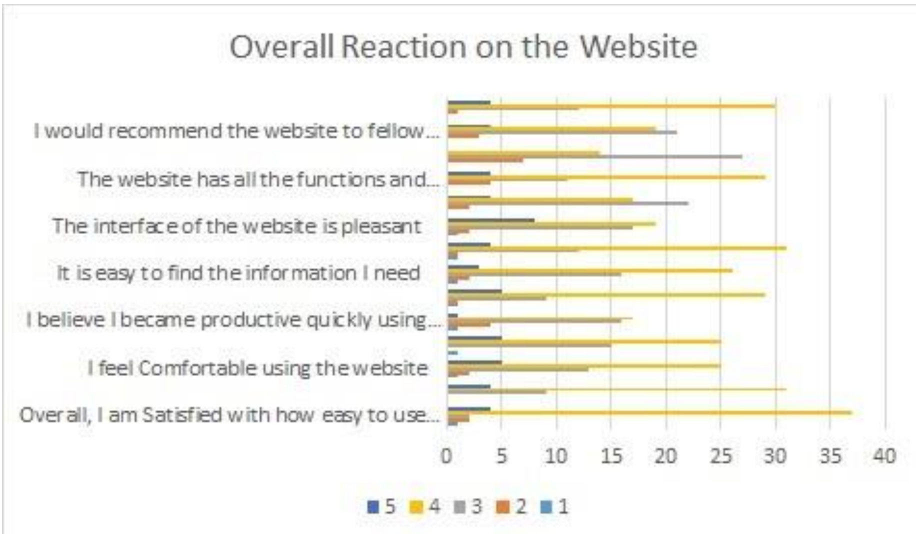


Figure 16: Survey Results of Overall Reaction on the Website

4.6. Study 3: Impact of Combinations of LEs Across Different Institutions

In addition to the above study, we wanted to see the impact of various combinations of LEs across different institutions during different terms of year. So, this study is aimed to evaluate the impact of student learning and understanding of programming concepts of various combinations of LEs i.e.., Gamification (G), Social Interaction (SI), Collaborative Learning (CL) within SEP-CyLE at North Dakota State University (NDSU) and Virginia Commonwealth University (VCU). All the combinations that have been used across both the institutions over a period of time are shown in the below table 11.

4.6.1. Study 3: Study Goal

The primary objective of this study is to evaluate the impact of various combinations of LEs of SEP-CyLE on student understanding and learning of programming concepts and testing concepts in an introductory programming course.

4.6.2. Study 3: Research Questions and Variables

The main objective of the study can be formally stated as a research question as follows:

RQ: Investigate the impact of different combinations of learning engagement strategies (i.e., CL, G, and SI) used in SEP-CyLE on student's understanding and learning of programming concepts in an introductory programming course at multiple institutions.

The independent and dependent variables of the study are provided in the below table 11.

Table 10: Study 3 Independent and Dependent Variables

Independent Variables	***Description***
Virtual Points (VP)	These are the reward points earned by the student for the completion of tasks in SEP-CyLE
# of LOs assigned	These are the # of LOs that were assigned by instructors at each site and for each section.
# of LOs attempted (LOA)	These are the # of LOs attempted by student from the list of assigned LO's
# of LOs Completed (LOC)	# of LOs student passed with at least 80% of the questions correctly answered.
Dependent Variables	***Description***
Pre/Post-test Score	Each test consisted of 21 questions and the students would earn one point for each question correctly answered
Higher-level Programming Concepts	The pre-/ posttest is divided into 8 programming concepts and we measured students' understanding of 8 high-level programming concepts (i.e., memory management, conditional statements, basic operators, methods, arrays, loops, OOP, software testing).
Course Grade (CG)	Final grade achieved by the student at the end of semester
Assignment Grade (AG)	Average of the assignment scores (in terms of percentage out of 100) for each of the subjects during the semester.
Exam Grade (EG)	Average of exam grades scored by the students in a semester.

4.6.3. Study 3: Participating Subjects

A total of 209 students participated in this study from both the universities. These subjects are divided as sections of CS1 at two universities. Each section utilized SEP-CyLE with different combination of LESs (3 or 2 or 1 or none). A summary of the intervention groups and subjects are provided in Table 12.

Table 11: Participation Subjects

# of LESs	Groups	Site	Year/Semester	# of subjects
3	G+SI+C G+SI+C	NDSU	Fall'2017 and Spring 18	14
		VCU	Fall 2017	36
2	G+SI	NDSU	Fall 2017	16
	G+CL	NDSU	Fall 2017	26
	SI + CL	NDSU	Fall 2017	19
1	G	VCU	Spring 2018	29
	SI	VCU	Spring 2018	32
	CL	NA	NA	NA
0	None - SEP-CyLE with no LESs enabled	VCU	Fall 2017	29

4.6.4. Study 3: Study Procedure

Step 1 -- Pretest: At the beginning of the semester, a pretest was conducted to establish a baseline regarding the student's understanding of software programming concepts. The test consisted of 21 questions that assessed knowledge of students on various programming concepts like arrays, pointers, software testing, memory management, etc.,

Step 2 -- Introducing SEP-CyLE to students: The students were shown how to use SEP-CyLE including how to peruse video tutorials and learning content, how can they track their progress, how to change their profile pictures, interact with their peers in the cyber learning environment, and how to post to the discussion forums. Each section of university site was trained separately depending upon the LEs that have been enabled to them.

Step 3 -- post-test: Toward the end of the semester, the students were re-tested on the knowledge of their proficiency on software programming and testing concepts. The post-test contained the same set of questions as of the pretest. This was done across all the sites to understand the impact of different combinations of LEs on improvement in their conceptual knowledge. We additionally assessed the student's course grade, assignment grades, and exam grades to comprehend the effect SEP-CyLE had on their course performance.

4.6.5. Study 2: Data Capture

The primary data source of data are the responses from students for pre-posttest data. We also assessed the relationship between the student's course grade, assignment grades, and exam grades to comprehend the effect SEP-CyLE had on their course performance.

4.6.6. Study 3: Summary of Results and Analysis

This section provides the details of the results that are analyzed and how these learning engagement strategies impacted student learning and understanding of programming concepts.

4.6.6.1. Pretest vs. Posttest Results

First, we calculated the average pre- and posttest results and Table 13 provides the brief details about the findings.

Table 12: Average Pre- and Posttest Scores Across All the Sections

Average Score	3 LESs	G+SI	CL+SI	CL+G	SI	G	None
# of subjects	50	16	19	26	32	29	29
Pre-test Scores	8.6	7.81	8.16	8.88	7.03	8.21	8.52
Post-test	11.56	13	11.57895	10.73077	10.1875	11.13793	10.21
Improvement	+2.96	+5.19	+3.42	+1.85	+3.16	+2.93	+1.69
Significance (p-value)	<0.001	<0.001	<0.001	0.012	<0.001	<0.001	0.036

These results have indicated that all the students have shown improvement in the posttest scores indicating that SEP-CyLE had a positive impact on student learning. When observed in detail, the sections that had enabled social interaction and gamification together had more impact on student learning when compared to that of the other sections with different combinations.

CL did not appear to add substantially to knowledge gains for students in this study. This is apparent all through the study groups. SI or G, when used independently had had comparative or higher gains compared contrasted with when CL was added to SI or G. Also, adding CL to "G+SI" did not positively affect students' knowledge gains.

When we compared the average assessment scores of students of different sections across both the universities, the results have also shown that the combination of Gamification and Social Interaction had a larger positive effect on student learning when compared to that of the other combination. This could be seen in Table 14 below.

Table 13: Average Assessment Scores of Students

	3 LESs	G+SI	CL+SI	CL+G	SI	G
# of LO's Assigned	11	11	11	11	11	11
# of LO's Attempted	4.22	6.13	1	0.85	8.56	8.17
# of LO's Completed	3.38	4.88	0.32	0.5	5.25	5.34
Assignment Scores	69.69	81.26	71.55105	61.85269	80.46	80.15
Exam Scores	78.01	88.04375	86.70947	78.93346	74.81	82.40
Class Grade	86.72	87.15875	83.62947	79.01885	79.51	84.85

Also, the students in G+SI, G, SI groups tried and subsequently finished (passed with at least 80% of questions correctly answered) the most significant number of LOs This may have been a result of the accompanying reasons:

To begin with, for each and every LO completed by students, students are awarded certain virtual points (VPs). VP is a gamification element and students just get VPs when "G-gamification" is empowered. Likewise, there are VPs related with SI (social Interaction) exercises. Only groups that have SI empowered will get VPs for those exercises (e.g., profile pic, posting/replying to discussion thread).

Next, students most abundant in VPs also show up in the “leaderboard”, so students are motivated (perhaps extrinsically) to be more connected on SEP-CyLE and get greater perceivability with their peers.

4.6.6.2. Study 3: Correlation Results between SEP-CyLE Activity and Course Performance

The correlation results have provided us the relationship between the SEP-CyLE metrics and students’ performance at different levels during the semester. All these results are discussed in detail in below:

Regarding CL+G treatment group, VPs are positively correlated to assignment scores. The outcomes demonstrated a significant but weak correlation ($r2=0.2$ and $p= 0.03$) between VPs and

AS. For G treatment group; VPs, LOA and LOC were all positively and significantly correlated to CG and AS.

VPs - weak correlations with CG (r2=0.16 and p= 0.04); and with AS (r2=0.26 and p= 0.04)

LOA - weak correlation with CG (r2=0.17 and p= 0.02); moderate correlation with AS (r2=0.40 and p= <0.001)

LOC - weak correlations with CG (r2=0.28 and p= 0.003) and with ES (r2=0.17 and p= 0.02); moderate correlation (r2=0.40 and p= 0.001) with AS

For SI treatment group; VPs, LOA and LOC were all positively correlated to CG

VPs - weak correlation (r2=0.14 and p= 0.03) with CG;

LOA - weak correlation (r2=0.11 and p= 0.06) with CG;

LOC - weak correlation (r2=0.17 and p= 0.02) with CG;

This shows us that SEP-CyLE had a positive impact on student learning and helped students to improve their conceptual knowledge of programming concepts and testing topics.

In brief, the results have indicated that SEP-CyLE enabled with any learning engagement strategies helped students to improve their conceptual knowledge of computer programming. Leaderboards had a positive impact indicating that it is a good motivator for students to be more engaged on using SEP-CyLE. At the same time, SEP-CyLE with some LEs enabled had a better impact on students learning versus SEP-CyLE with no LEs.

4.7. Study 4: Analyzing Knowledge Deficiencies of Students

The aim of this study is to analyze the knowledge deficiencies of students enrolled in an introductory programming course. For CL+G treatment group, VPs were positively correlated to assignment scores. The results showed a significant but weak correlation ($r2=0.2$ and $p= 0.03$) between VPs and AS.

This study has been conducted across 5 different universities across the United States. The group of universities includes urban and rural universities, located in 5 different U.S. states, mix of undergraduate institutions and PhD granting institutions, as well as minority serving institutions. At each site, students were pre-tested on core CS1 principles at the beginning of the course, then were introduced to various learning contents with help of the SEP-CyLE cyber learning environment, and post-tested at the end of the course using the same instrument that was used during the pre-test. More details of the study are provided in the following sections.

4.7.1. Study 4: Study Goal:

The main criteria of the study is to evaluate the knowledge deficiencies of students enrolled in an introductory programming courses at different U.S. Institutions. To achieve this objective, coordinated studies were planned and executed across five different U.S. Universities that pre-tested incoming CS1 students and then re-tested graduating CS1 students on their understanding of computer programming concepts. To allow the comparison across institutions, the same instrument was utilized to assess the student's comprehension of programming concepts during the pre and post-test.

4.7.2. Study 4: Research Questions

This study investigated the following four research questions:

RQ1: What are the knowledge deficiencies of CS1 students at each collaborating institution?

RQ2: Can knowledge deficiencies be generalized across different student groups in CS1?

RQ3: How can the results from 1 and 2 be used to guide the expected learning outcomes for CS1 students?

4.7.3. Study 4: Participating Subjects

There was a total of 505 subjects across all the sites (128 subjects at Virginia Commonwealth University, 84 subjects at North Dakota State University, 141 subjects at Miami University, 40 subjects at Georgia Southern University and 112 subjects at Florida International University).

4.7.4. Study 4: Study Procedure

The students were pretested at the beginning of the semester and they were introduced to SEP-CyLE. The instructors assigned the list LO's based on the topics that was taught in that week. At the end of the semester, the students were again tested with the help of posttest instrument. This instrument had the same set of questions as that of the pretest.

4.7.5. Study 4: Data Capture

The main data source of this study was the responses from students for the pre- and post-test. These instruments consisted of 21 questions each and these are categorized into eight categories of programming concepts (for example, arrays, memory management, and software testing, etc.). The instrument was developed in consultation with professors, CS1 instructors and STEM researchers (also co-authors) at participating institutions and is available at https://stem-cyle.cis.fiu.edu/docs/publications/Fall17-Pre-Posttest.pdf.

The eight categories into which the pre-/posttest is divided below:

1. Memory Management
2. For Loops.
3. Conditional Statements.

4. Arrays.
5. Basic Operators.
6. Methods.
7. OOPs Concepts.
8. Software Testing Concepts.

4.7.6. Study 4: Evaluation Criteria

The student's pre and post tests are assessed to get an idea of their programming knowledge at the beginning and their knowledge gains/ deficiencies at the end of the semester. This is illustrated with the comparison of question Q1's responses by 111 students as shown in Table 2. Table 15 displays the frequency distributions of the different answer options selected by the students in pretest vs. posttest.

Table 14: Contingency Table: Answer Frequencies Computed for Q1 (N = 111): Pretest Vs Posttest

N=111		Posttest					Total
		A	B	C	D	E	
Pretest	A	**59**	**0**	**6**	**1**	**0**	66
	B	**9**	2	1	0	0	12
	C	**17**	3	4	0	0	24
	D	**8**	0	0	1	0	9
	E	**0**	0	0	0	0	0
Total		93	5	11	2	0	111

Correctly answered - Both pre/post tests
Correctly answered - Pretest only
Correctly answered - Posttest only
Incorrectly answered - Both pre/post tests

We labeled the answers in color based four categories based on the following ways: 1) when the questions are answered correctly in both pre and posttest. 2) Questions correctly answered only in pretest, this means that students had prior knowledge on those concepts. From the table we could see that 59 out of 111 students chose correct option ‘a’ for Q1 in both pre and posttests instruments. Based on these answers, we measured as knowledge deficiency if the student was not able to correctly in the posttest, irrespective of what they answered in the pretest. The student has knowledge gain, if the student answered correctly that question in their posttest.

4.7.7. Study 4: Summary of Results and Analysis

This section provides an overview of the knowledge deficiency study results both in pre- and in post-tests. In this analysis, the categorization of the questions was done using the percentages of incorrectly answered questions rather than looking at the percentages of correctly answered question. We did this since our goal was to identify deficiencies in knowledge both at the start and again at the end of the CS1 course.

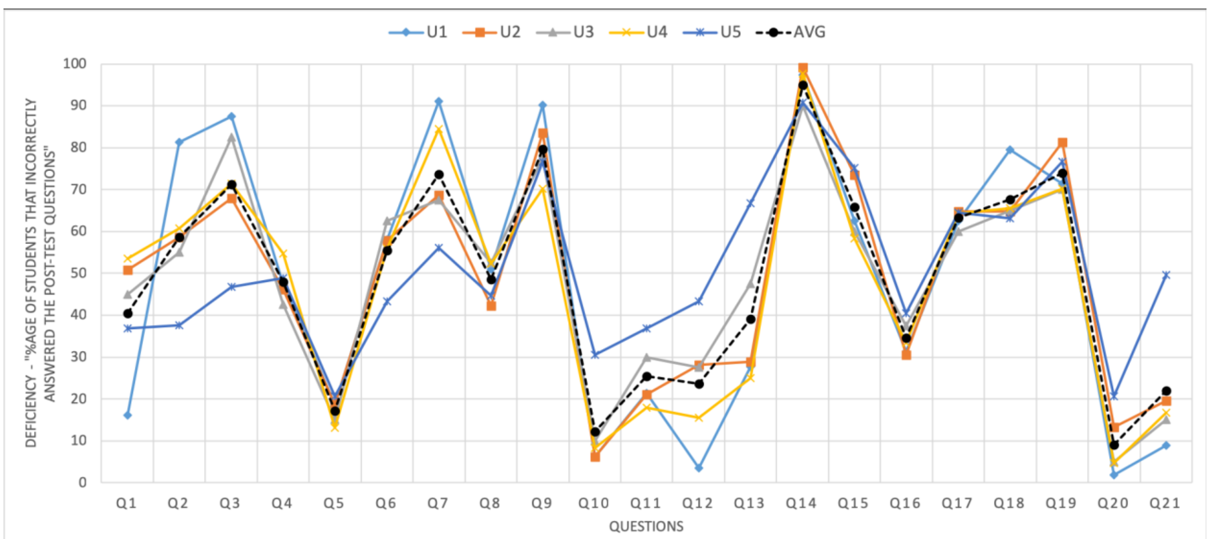


Figure 17: Knowledge Deficiencies Across all the Institutions

RQ1: What are the knowledge deficiencies of CS1 students at each collaborating institution?

From figure 18, it is evident that performance of students in questions 7, 9,14, 15, 17, 18, and 19 depict knowledge deficiencies in the related topics. These topics include Basic operators, For Loops, Methods and software testing concepts. It is visibly evident that all the students across different universities have knowledge deficiencies in these areas and they need assistance in these topics.

RQ2: Can knowledge deficiencies be generalized across different student groups in CS1?

On comparing pre and posttest results, from figure 19, we computed knowledge deficiencies and gains across all the universities.

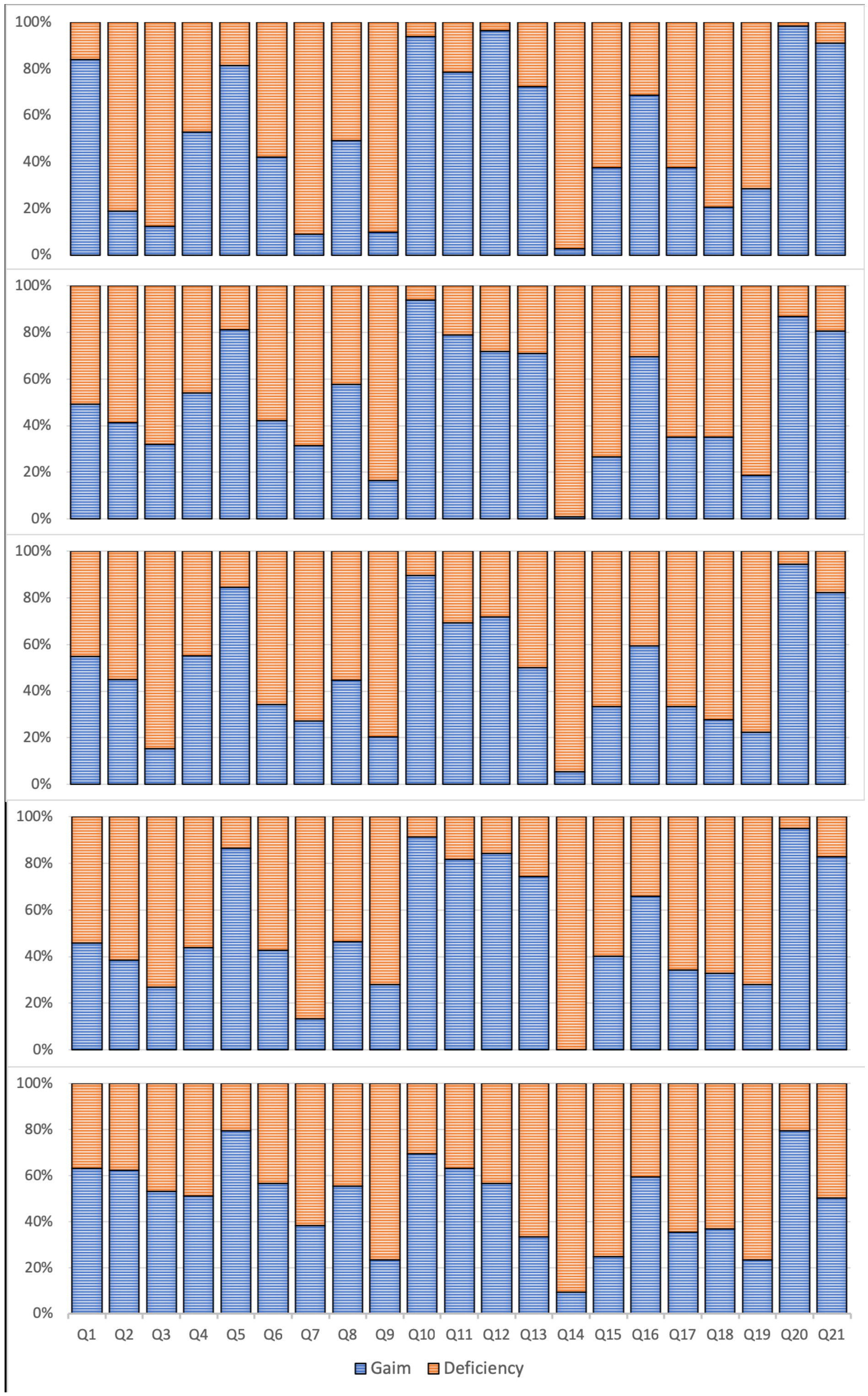


Figure 18: Knowledge Gain Vs. Knowledge Deficiency Across all Universities

RQ3: How can the results from 1 and 2 be used to guide the expected learning outcomes for CS1 students?

Based on the results from Figure 19, we gathered the questions into six general categories of topics covered (Figure 20), as we can see that there are particular questions under various classifications that show knowledge deficiency after the completion of course. For example, the students across all the universities show a significant knowledge deficiency in three questions of five under methods and OOPS concepts.

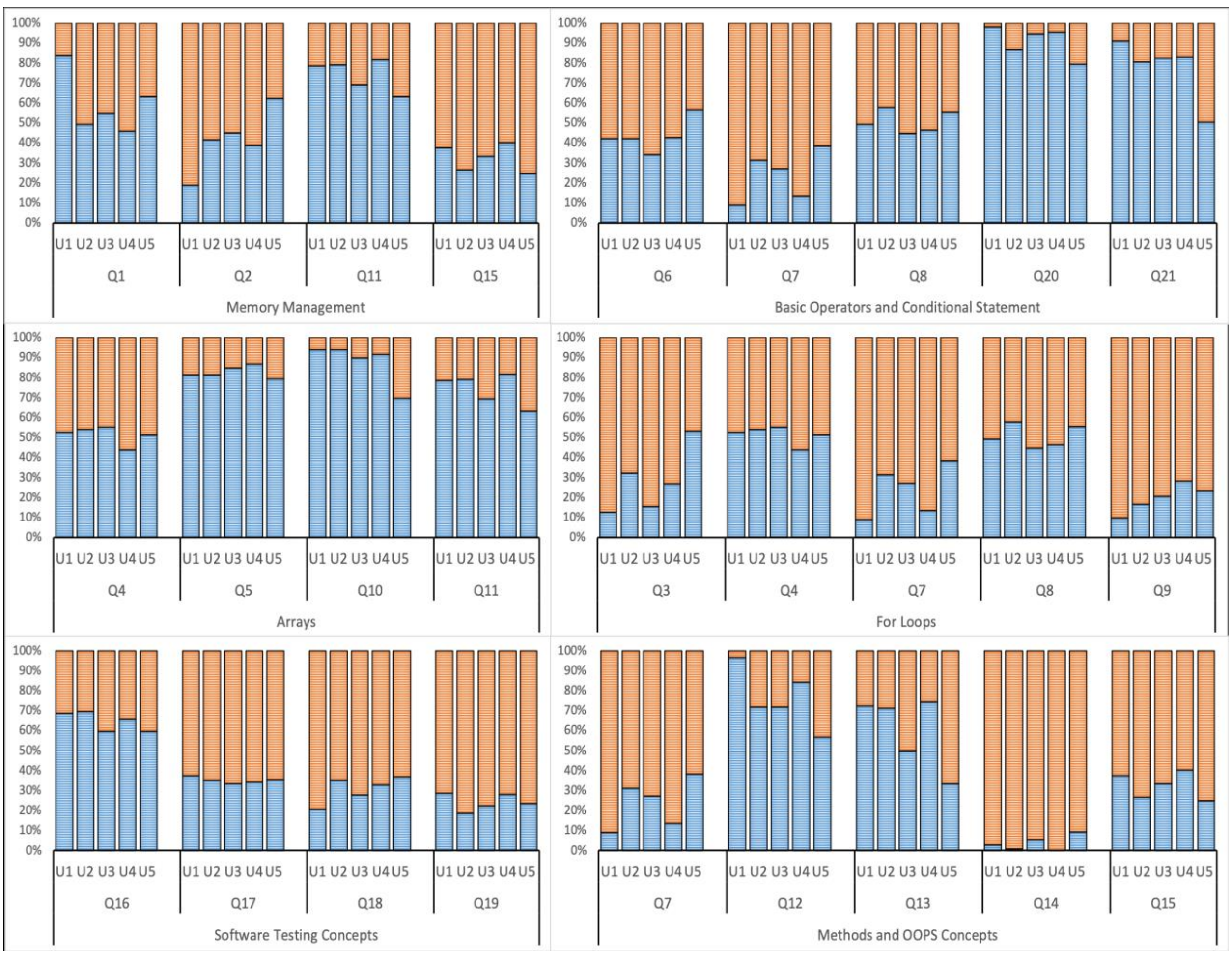


Figure 19: Knowledge Gain Vs. Knowledge Deficiency Across 5 Universities in Pre-/Post Tests Grouped by Concepts

In Software Testing concepts, there is not much variation between the height of corresponding bars indicating that this area of learning needs more attention. With only a few exceptions, when we compare each university's performance differ from pretest to post-test, that change in each KD area was similar for each and every university. Furthermore, the areas that offered students problems in one university are very similar to those at the others. That is, there is some internal consistency to this data although the data sizes may be too small to draw statistical conclusions.

Encouraged by these results, we will keep on developing extra learning objects that attention on the particular areas of each programming concept that have revealed to be a knowledge deficiency. To achieve this, each LO should be separated into subtopics. This specific would enable a student to focus an individual concept in an inside before moving further into another one. For instance, including more Learning Objects about programming testing ideas that focus on single concepts, for example, choosing test data for a conditional statement as a stand-alone LO, would enhance student performance on software testing and may have a side effect of improving student performance in other area related to understanding program output.

4.8. Study 5: Impact of LOs on Student Learning

The primary objective of this study is to analyze and evaluate how the repository and availability of digital learning objects affected student learning and comprehension of programming concepts.

This study was conducted across two different sections of introductory computer science course taught by different instructors.

4.8.1. Study 5: Study Goal

The basic objective is to analyze the outcomes of student learning when the digital learning objects within SEP-CyLE are assigned to students. To assess the results, we used pre/post test instruments for this study, which is same across both the sections. The study also looked into the satisfaction of students with SEP-CyLE and how the Digital Learning Objects have helped students in learning via survey (which was conducted at the end of the semester).

4.8.2. Study 5: Research Questions

The study determined the following research question:

Investigate the effect of availability of digital learning objects included in SEP-CyLE on students' acquisition of software programming and testing conceptual knowledge, tools and techniques in an introductory computer programming course.

4.8.3. Study 5: Participating Subjects

The case study was conducted across two different sections of an introductory programming course at North Dakota State University, taught by two different instructors. A total of 102 students are enrolled across the two sections of the course. Of those students, 26 elected to participate in the study.

4.8.4. Study 5: Study Design

The case study followed the similar study design as of the previous studies. The students are asked to take pretest at the beginning of the semester (consisted of 21 multiple choice questions). Following that week, students are introduced to the features of SEP-CyLE and how to use SEP-CyLE. Then students are assigned the LO's based on the group i.e., if it is control group, then the LO's are assigned at the beginning of the semester, experiment group students are assigned

LO's based on the content that was taught in the respective week. A total of 11 LO's are assigned to students in both the sections. During final week, posttest is conducted (which is same as pretest).

4.8.5. Study 5: Data Collection and Evaluation Criteria

The key data source is the students' responses to the pre- and post-tests (as described in chapter 3). In addition, the students completed a survey where they responded to questions using a 5-point Likert scale, (chapter 3) (data about survey). We also collected SEP-CyLE data for all students in terms of the number of LOs attempted, number of LOs passed, and time spent on SEP-CyLE, and the number of virtual points the student earned. Furthermore, the students' performance will be analyzed in terms of their assignment scores, exam scores and end of semester grades. With the help of these data variables, we analyzed the relationship between them.

The independent and dependent variables of the study are listed below:

4.8.5.1. Independent Variables

Number of LO's attempted: This is computed as the Number of LOs that students attempted from the assigned ones in Sep-CyLE.

Number of LO's passed: This is computed as the number of LOs completed by students with at least 80% questions correctly answered.

Total time spent on SEP-CyLE: Sum of the total amount of time spent on all the assigned LOs.

Virtual Points earned: For each LO or activity completed, students earn virtual points.

4.8.5.2. Dependent Variable

Average assignment scores: Average of all the assignment scores of students.

Average exam scores: Average of all the exam scores of students.

End-of-semester course grades: These are the grades that are assigned to students at the end of the semester

4.8.6. Summary and Analysis of Results

This section talks about the impact of the digital learning objects within SEP-CyLE on student learning. To understand the impact, I conducted paired t-test. The results from the t-test showed that there is a significant increase ($p<0.01$) in the post test scores of students in both the sections, shown in Figure 21.

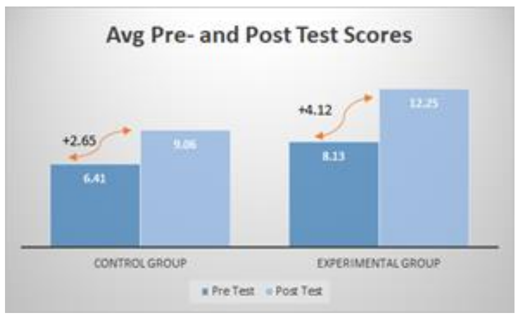


Figure 20: Pre and Post Test Scores of Control Group and Experiment Group

The highest increase in the post test scores is seen in the experimental section for which the LO's have been assigned on weekly basis.

In addition to that, to better understand the impact of LOs on individual concept acquisition of students, we divided pre-/post test questions into different categories as listed below:

- Memory Management: 1, 2,11 and 15
- Basic Programming Concepts: 3,4,7,8,9 and 21
- Arrays: 4, 5, 10, 11
- Basic Operators: 6 and 20
- Methods: 7
- Pointers: 10
- OOPs Concepts: 12, 13, 14 and 15

- Software Testing Concepts: 16, 17, 18, 19 and 20

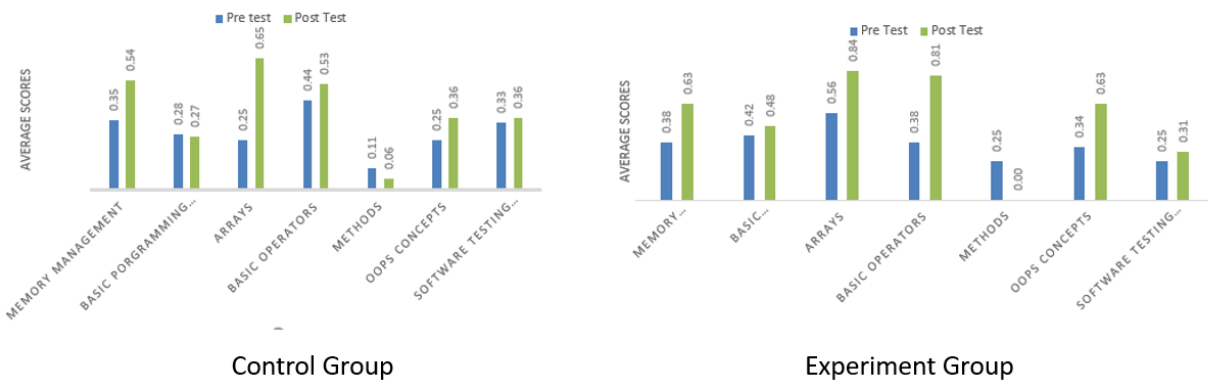


Figure 21: Mean Pre/Post Test Scores for Different Categories of Programming Concepts

To analyze these results, we measured the average pre-post test scores across both the sections. these results have shown that in both the sections, as shown in figure 22.

When we closely observe this figure 22, we can see that student’s performance has seen improvement in most of the concepts. But then for methods concepts there has been no improvement in their posttest scores. This could be because the students are struggling with this methods concepts. The study helped us understand that the students belonging to both the sections, irrespective of their environments, are having trouble in understanding methods concepts (also can be considered as knowledge deficiency).

To get better understanding of the impact of SEP-CyLE on student learning, we looked into the different variables of SEP-CyLE and their relationship with the performance of students (Table 15). Table 15 gives Pearson correlation coefficient ‘r’ (strength of correlation) and p-value ‘p’ (statistical significance) between these independent and dependent variables. The experiment group has more grounded positive connection (significant at p-value <0.1). Correspondingly, giving additional resource support can certainly contribute to the overall success of students in an introductory computer programming course. In view of these outcomes, SEP-CyLE LO’s positively wedged students’ performance.

Table 15: Relationship between SEP-CyLE Usage Metrics vs. Student Performance

	Control Group			Experiment Group		
Vs.	Average assignment scores	Average exam scores	End-of-semester course grades	Average assignment scores	Average exam scores	End-of-semester course grades
# of LOs attempted	r= 0.412 p =0.101	r = 0.231 p =0.371	r= 0.392 p =0.119	r = 0.685 p =0.06	r = 0.283, p =0.497	r = 0.645 p =0.08
#Los passed	r = 0.412 p =0.101	r = 0.231 p =0.371	r.= 0.392 p =0.119	r = 0.829 p =0.01	r = 0.516 p =0.190	r = 0.872 p =0.005
Time Spent	r = 0.275 p =0.285	r = -0.192 p =0.459	r = 0.157 p = 0.547	r = 0.667 p =0.07	r = 0.477 p=0.233	r = 0.717 p =0.05

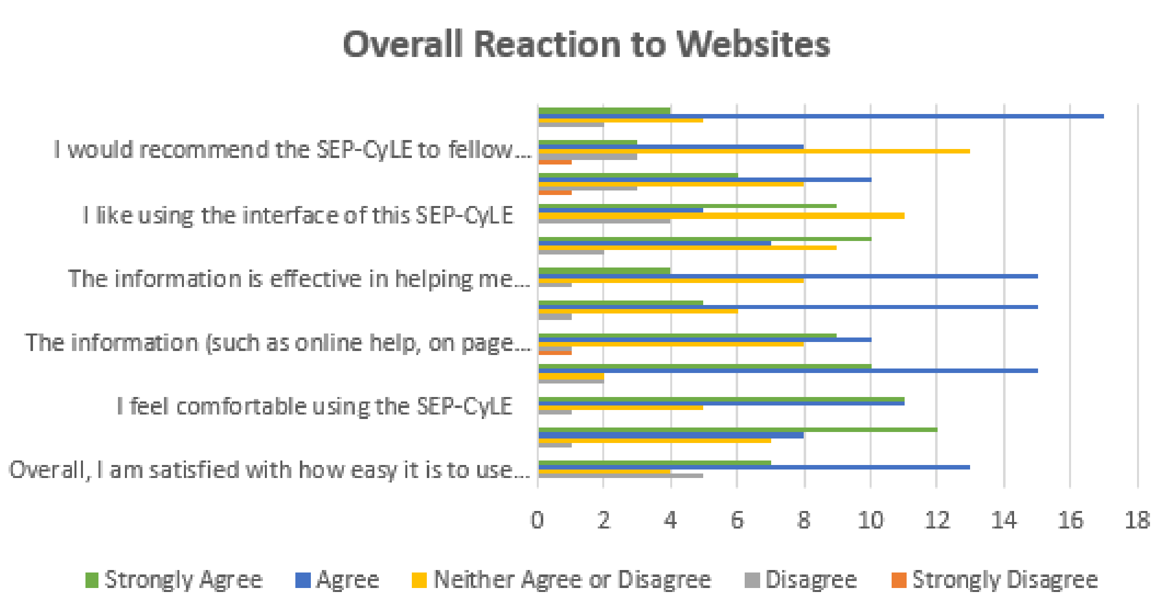


Figure 22: Survey Results of Overall Reaction on the Website

4.8.7. Study 5: Survey Results

In respect to student feedback, there was a positive feedback on SEP-CyLE (How students felt about the LO's, usability of SEP-CyLE and overall satisfaction on SEP-CyLE). Students recorded their survey responses on a Likert scale [1= Strongly Disagree; 2= Disagree; 3= Neither Agree nor Disagree; 4= Agree; 5= Strongly Agree]. These survey responses are shown in Figures 23 and 24. The majority of the students emphatically concur that SEP-CyLE is pleasant.

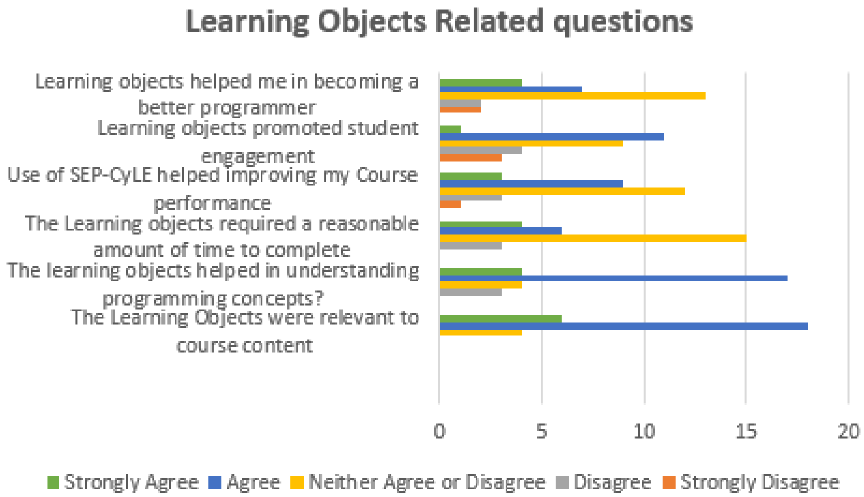


Figure 23: Survey Results on Learning Object Related Questions

Figure 24 results indicate that students are satisfied with the Learning Objects that are assigned to them during the semester. They are positive about that the LO's helped them in understanding the programming concepts.

We also wanted to look into if students are interested in using Learning Objects as their learning material. And the survey results indicate that more than half of the students completed more than ten Learning Objects during thru their course (Shown in Figure 25).

Figure 24: Number of LOs Completed

We likewise requested students to describe how they are motivated to use SEP-CyLE and complete all the allotted assignments. Also, asked them if Gamification has impacted their behavior towards learning. Figure 26 delineates the overview reactions with respect to gamification features of SEP-CyLE. What's more, there was a most astounding positive reaction that the virtual points have helped students to take up the task and finish it.

This study has provided evident results that SEP-CyLE would provide promising results on students with continuous resource support and learning engagement strategies included give better outcomes in instructing programming and testing concepts for introductory computer programming courses.

From this study, it was learnt that, a digital LO ought to be assigned after a topic is introduced in the class however before in class tests or exams to enhance student performance. Overall survey results have shown that students like to work on LOs in groups utilizing SEP-CyLE. LO enabled student to assess and improve their comprehension before endeavoring assignments.

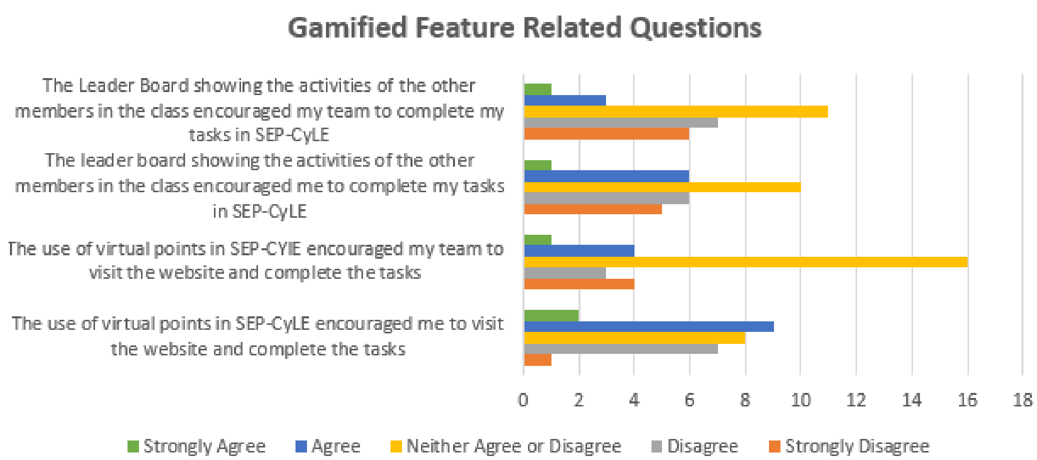


Figure 25: Survey Results on Gamified Feature Related Questions

5. DISCUSSION

All the studies have provided an evidence that learning engagement strategies in a cyber-learning environment (SEP-CyLE) improved student engagement and learning. Also, it was found that using SEP- CyLE in classrooms help motivate students towards their own learning, especially in an introductory programming course.

The systematic literature review results revealed that Gamification had positive impact on student learning. Some of the papers revealed that it had constructive outcome on women participation in STEM. It was also found that if the learning environment is constructed in the local language with all the gamification features, then it will have a more positive impact on learners. In addition to these positive outcomes, gamification can also have negative impact on student learning, so care needs to be taken when adding gamification in educational learning environments. For example, one of the papers reported that one participating student was not happy seeing his/her badge in his dashboard. Another paper reported that neither badges nor heat maps influenced the behavior of majority of the students.

Study 1 revealed that introducing team collaboration points in combination with gamification features did not result in noteworthy difference in the performance of the students exposed the collaboration features. Albeit surprising, this could have been to some extent because of the way that virtual groups on SEP-CyLE were arbitrarily chosen and were not same as groups that cooperated on course task and activities which may have affected their motivation. Also, the collaboration points in SEP-CyLE did not require communication among the group as long as individual team members finished an LO independently.

Study motivated the design and implementation of follow-up studies to investigate the impact of learning engagement strategies on student learning across different student groups and institutions. A variety of learning engagement strategies combination were analyzed across multiple studies at North Dakota State Universities and across different universities. The results from these studies showed that combining gamification with social interaction yields better results (in terms of student learning and engagement), when compared to other LESs combinations. We also found that SEP-CyLE in any manner helps students improve their conceptual knowledge of computer programming. Gamification and Social-Interaction learning engagement strategies are more advantageous (when used separately or in combination) when compared against Collaborative Learning. Digital LOs helped students' progress in their assignments and also helped at improving their course grades.

Based on the results from the studies, SEP- CyLE could be extended in other domains and settings as well. For example, using SEP-CyLE and its digital learning objects can also help identify the order the content should be introduced in CS1/CS2 programming course. This would help discover hard topics along with their prerequisites, with the goal that the instructors can make sure to teach the prerequisites when students have difficulty in understanding the topics. Learning objects that are present in SEP-CyLE would be accessible for this because, all these learning objects are individual and digestive chunks of information that can be assigned to students.

Motivated by the results of the knowledge deficiency study, we are planning to include more digital learning objects in SEP-CyLE that will help students at understanding additional programming concepts. Also, if needed, we would categorize and breakdown the complexity of each LO into smaller modules. This way, it will help students gain better understanding of each concept and assess their knowledge at each step. In future, we are planning to conduct in depth analysis, which includes to learn the reason why student selected particular option as answer. In a way this will analyze the pattern of mistakes of students and can provide them better feedback to improve their knowledge and skills.

6. CONCLUSION

This section discusses the major contribution of the work described in this dissertation to Computer science and Software Engineering education research and practice. This section also enlists the publications that will be the output of this dissertation work.

6.1. Contribution to Research and Practice

The main goal of this thesis is to enrich computer science education by deepening our understanding of learning engagement strategies: Gamification, Collaborative Learning and Social Interaction.

Matocha et al. [61] asserted that creative teaching methods should be employed in computer science education, particularly for non-majors and students in the start of their studies, due to the fact that they typically find computer science to be brimming with theoretical, technical, or even tedious concepts. Learning engagement strategies provide opportunities to tap the motivational power in education.

This dissertation presents an overview of some learning engagement strategies and techniques that can be used to elicit motivation, supported by the results of the studies (described in Chapter 3). We believe that if these LEs are successfully used, they can support learning by enhancing students' study practices and motivation to learn. The results from this work provide insight into how to enhance students' motivation and understanding of programming concepts in an introductory programming course. These insights promise the use of SEP-CyLE had a positive impact on student learning and could be recommended to other institutions.

6.2. List of Publications

This section describes the publications that will result from the work done for this dissertation.

1. MR Narasareddygari, Walia, G., and Radermacher, A. "Gamification in Computer Science Education – A Systematic Literature Review" 125th Annual ASEE Conference, June 24 - 27, 2018, Salt Lake City, Utah.d
2. MR Narasareddygari, Walia, G., and Radermacher, A. "Using Gamification and Cyber Learning Environment to Improve Student's Learning in an Introductory Computer Programing Course —an empirical case study" 125th Annual ASEE Conference, June 24 - 27, 2018, Salt Lake City, Utah.
3. MR Narasareddygari, Walia, G., Borchert, O., and Radermacher, A. "Evaluating Learning Engagement Strategies in a Cyber Learning Environment during Introductory Computer Programming Courses-- An Empirical investigation" 125th Annual ASEE Conference, June 24 - 27, 2018, Salt Lake City, Utah
4. Narasareddygari, MR., Walia, G., Duke, D., Ramasamy, V., Kiper.J., Davis, D., Allen, A., and Alomari, H., "Evaluating the Impact of Combination of Engagement Strategies in SEP-CyLE on Improve Student Learning of Programming Concepts" SIGCSE 2019.
5. Borchert, O., Byram, A., Duke, D., Radermacher, A., Narasareddygari, MR., Walia, G., " Lessons Learned using a Cyber Learning Environment in CS1 Classrooms" ASEE 2019.

6.3. Publications under Review

1. Narasareddygari, MR., Walia, G., Duke, D., Ramasamy, V., Kiper.J., Davis, D., Allen, A., and Alomari, H., " Evaluating Common Knowledge Deficiencies of Students in Introductory Computer Programming Courses: A Multi-Site Study" Computers and Education Journal.
2. Narasareddygari, MR., Walia., Radermacher, A., Potvin, G., and Kotala, P., "Using SEP-CyLE to Support CS1 Pedagogy: An Empirical Investigation" *FIE 2019*.

3. Brown, T., Singh, M., Kaur, R., Walia, G., “Using Peer Code Review to Support Pedagogy in an Introductory Computer Programming Course”, FIE 2019.
4. Kaur, R., Singh, M., Brown, T., Narasareddygari, M., Using Association Rule Mining Algorithm to Improve the Order of Content Delivery in CS1 Course”, FIE 2019.
5. Duke, D., Narasareddygari, MR., Thirunarayanan, M.O., and Byram, A., " Collateral Benefits of Multi-Institution Collaborations in Computer Science Education".

REFERENCES

1. Narasareddygari. M.R., Walia. G., Borchert. O., and Radermacher. A. Evaluating Learning Engagement Strategies in a Cyber Learning Environment during Introductory Computer Programming Courses-- An Empirical investigation. *In the proceedings of 125th Annual ASEE Conference*, June, 2018, Salt Lake City, Utah

2. Winn.S. Student motivation: a socio-economic perspective, *Studies in Higher Education*, 27. 445-457. 10.1080/0307507022000011552.

3. Robins. A., Rountree. J., Rountree. N., Learning and Teaching Programming: A Review and Discussion, *Computer Science Education*, 13(2), Swets & Zeitlinger, pp. 137-172, 2003.

4. Lin.H.T., and Kuo. T.H, Teaching programming technique with edutainment robot construction *In the proceedings of 2nd International Conference on Education Technology and Computer*, Shanghai, 2010, pp. V3-226-V3-229. DOI: 10.1109/ICETC.2010.5529557

5. Beaubouef. T., and Mason. J., Why the high attrition rate for computer science students: some thoughts and observations. *In the proceedings of SIGCSE* Bull. 37, 2 (June 2005), 103-106. DOI= 10.1145/1083431.1083474

6. Kinnunen. P., and Malmi. L., 2006. Why students drop out CS1 course?. *In Proceedings of the second international workshop on Computing education research (ICER '06).*ACM, New York, NY, USA, 97-108. DOI= 10.1145/1151588.1151604.

7. Kölling. M., *Teaching Java with BlueJ – A Sequence of Assignments* University of Southern Denmark Technical Report No 1, September 2002, ISSN No. 1601-4219.

9. Pirker.J., Schiefer.M.R., and Gütl.C., Motivational active learning: engaging university students in computer science education. *In Proceedings of the 2014 conference on Innovation & technology in computer science education (ITiCSE '14).* ACM, New York, NY, USA, 297-302. DOI: 10.1145/2591708.2591750

10. DISESSA, A. A. 2000. *Changing Minds*. MIT Press, Cambridge, MA..

11. Kurkovsky.S., Engaging students through mobile game development. *In Proceedings of the 40th ACM technical symposium on Computer science education (SIGCSE '09).* ACM, New York, NY, USA, 44-48. DOI: 10.1145/1508865.1508881

12. C.S.Gardiner, Improving the student success and retention of underachiever students in introductory computer science, *Journal of Computing Sciences in Colleges*, v.26 n.6, p.16-22.

13. R. Chang-lau and P. J. Clarke. Software engineering and programming cyberlearning environment (SEP-CyLE), 2018. https://stem-cyle.cis.fiu.edu/instances (retrieved Jan. 2018).

14. Papert, S. 1980. *Mindstorms: Children, computers, and powerful ideas*. Basic Books New York, NY:

15. Owens, B. B., & Stephenson, C. 2011. *CSTA K – 12 Computer Science Standards*. New York, NY. Retrieved from http://dl.acm.org/citation.cfm?id=2325380.

16. ACM & IEEE. 2013. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. Retrieved from http://www.acm.org/education/CS2013-final-report.pdf

17. National Research Council (NRC). 2003. *Beyond productivity: Information technology, innovation, and creativity*. Washington DC: National Academies Press.

18. National Research Council (NRC). 2005. *Avoiding surprise in an era of global technology advances*. Washington DC: National Academies Press. Retrieved from http://www.nap.edu/catalog.php?record_id=11286

19. Alexandra.B.K, *Gamification in Introductory Computer Science* 2015. ATLAS Institute Graduate Theses & Dissertations. 8. https://scholar.colorado.edu/atlas_gradetds/8

20. National Research Council (NRC). (2010). *The rise of games and high performance computing for modeling and simulation.* Washington DC: The National Academies Press. Retrieved from http://www.nap.edu/openbook.php?record_id=12816.

21. Deterding, S. 2011. Gamification by design: Response to O'Reilly. *Gamification Research Network*. Retrieved from http://gamification- research.org/2011/09/gamification-by-design response-to- oreilly/

22. Narasareddygari. M.R., Walia, G., and Radermacher, A. Gamification in Computer Science Education – A Systematic Literature Review *In Proceedings of 125th Annual ASEE Conference*, June 24 - 27, 2018, Salt Lake City, Utah.

23. Richards, C., Thompson, C. W., & Graham, N. T. C. Beyond Designing for Motivation : The Importance of Context in Gamification. *In CHI PLAY '14 Proceedings of the first ACM SIGCHI annual symposium on Computer-human interaction in play* (pp. 217–226). Toronto, Canada: ACM New York.

24. Singer. L.,and Schneider.K., 2012. It was a bit of a race: gamification of version control. *In Proceedings of the Second International Workshop on Games and Software Engineering: Realizing User Engagement with Game Engineering Techniques* IEEE Press, Piscataway, NJ, USA, 5-8.

25. Moccozet, L., Tardy, C., Opprecht, W., & Leonard, M. (2013). Gamification-based assessment of group work. 2013 *International Conference on Interactive Collaborative Learning* , 171–179. https://doi.org/10.1109/ICL.2013.6644565.

26. Jorge.S. & Redondo.D., Rebeca & Ana.V . A social gamification framework for a K-6 learning platform. *Computers in Human Behavior*. 29. 10.1016/j.chb.2012.06.007..

27. Paul Denny. 2013. The effect of virtual achievements on student engagement. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* ACM, New York, NY, USA, 763-772. DOI:https://doi.org/10.1145/2470654.2470763.

28. Denny.P., Luxton-Reilly.A., and Hamer.J, The peerwise system of student contributed assessment questions, *In Proceedings of. Tenth Australian Computing Education Conference* Wollongong, Australia.

29. Li.W., Grossman.T., and Fitzmaurice.G., 2012. GamiCAD: a gamified tutorial system for first time autocad users. *In Proceedings of the 25th annual ACM symposium on User interface software and technology* (UIST '12). ACM, New York, NY, USA, 103-112. DOI: https://doi.org/10.1145/2380116.2380131.

30. Domınguez.A., Saenz-de-Navarrete.J., de-Marcos.L., Fernandez-Sanz.L., Pages.C, and Martınez-Herraiz.J.J., Gamifying learning experiences: Practical implications and outcomes, *Computers & Education*, v63 p380-392, 2013.

31. Haaranen.L., Ihantola.P., Hakulinen.L., and Korhonen.A.,. How (not) to introduce badges to online exercises. *In Proceedings of the 45th ACM technical symposium on Computer science education* ACM, New York, NY, USA, 33-38. DOI: https://doi.org/10.1145/2538862.2538921.

32. Kuh G.D. Assessing what really matters to student learning inside. *The national survey of student engagement change*. Mag High Learn. 2001;33:10–17

33. Krause, K & Coates H 2008, Students' Engagement in first-year university, *Assessment and Evaluation in Higher Education* 33 (5) pp. 493-505

34. OECD, PISA 2009 *Results: Learning to Learn—Student Engagement, Strategies and Practices*, vol. 3. Paris, France: OECD Publishing,2010.

35. Bodnar.C.A, Anastasio.D., Enszer.J.A, & Burkey.D.D, Engineers at Play: Games as Teaching Tools for Undergraduate Engineering Students. *Journal of Engineering Education*.

36. Pedreira.O., García.F., Brisaboa.N., & Piattini.M., *Gamification in software engineering–A systematic mapping. Information and Software Technology*, 57, 157-168.

37. Arksey, H and O'Malley, L (2005) Scoping studies: towards a methodological framework. *International Journal of Social Research Methodology*. pp. 19-32. ISSN 1364-5579

38. Hamari, J., Koivisto, J., & Sarsa, H. (2014). Does gamification work? - A literature review of empirical studies on gamification. *In Proceedings of the 47th Annual Hawaii International Conference on System Sciences*, (pp. 3025-3034). [6758978] IEEE COMPUTER SOCIETY PRESS. https://doi.org/10.1109/HICSS.2014.377

39. Bruner J. 1990. Constructivist Theory. http://tip.psychology.org/bruner.html.

40. Huitt W. 2003. Constructivism. Educational Psychology Interactive. http://chiron.valdosta.edu/whuitt/col/cogsys/construct.html.

41. Dawn McKinney and Leo F. Denton. Developing collaborative skills early in the CS curriculum in a laboratory environment. *In Proceedings of the 37th SIGCSE technical*

symposium on Computer science education ACM, New York, NY, USA, 138-142. DOI: https://doi.org/10.1145/1121341.1121387.

42. Yerion K.A., and Rinehart J.A., Guidelines for Collaborative Learning in Computer Science." *ACM SIGSCSE* Bulletin 27(4).

43. Williams L., and Kessler R.R., . The Effects of “PairPressure” and “Pair-Learning” on Software Engineering Education. *In the Proceedings of 13th Conference on Software Engineering Education & Training*, 2000.

44. Gokhale A.A., Collaborative Learning Enhances Critical Thinking. *Journal of Technology Education* 7(1): 22-30.

45. Donna Teague and Paul Roe. 2008. Collaborative learning: towards a solution for novice programmers. *In Proceedings of the tenth conference on Australasian computing education* - Volume 78 (ACE '08), Simon Hamilton and Margaret Hamilton (Eds.), Vol. 78. Australian Computer Society, Inc., Darlinghurst, Australia, Australia, 147-153.

46. Vincent.T., 1988. *Leaving College: Rethinking the Causes and Cures of Student Attrition. Contemporary Sociology*. 17. 10.2307/2069700..

47. Johnson, D. W., Johnson, R. T., & Smith, K. The state of cooperative learning in postsecondary and professional settings. *Educational Psychology Review*, 19(1), 15-29. https://doi.org/10.1007/s10648-006-9038-8.

48. Pierre.D., & Sanna.J., &Frank.F., & Balacheff. N., &Sten.L., & Jong.D. T., & Lazonder, A &Sally.B., The evolution of research on computer-supported collaborative learning: from design to orchestration.

49. Salas, E., Sims, D. E., & Burke, C. S. Is there a "Big Five" in Teamwork? *Small Group Research*, 36(5), 555-599. http://dx.doi.org/10.1177/1046496405277134.

50. Wiley.D., Learning Objects, Content Management and E-Learning, *Content Management for E-Learning*, N. Ferran and J. Minguillo´n, eds., chapter 3, Springer, 2011.

51. Wiley, David. A. Connecting Learning Objects to Instructional Design Theory: A Definition, a Metaphor, and a Taxonomy. The Instructional Use of Learning Objects *Agency for Instructional Technology*.

52. Clarke.P.J., Davis.D.L., Chang-Lau.R., Kiper.J., Fu.Y., and Walia.G.,. Using WReSTT Cyberlearning Environment in the Classroom. *The 124th ASEE Annual Conference & Exposition - Computers in Education Division*, June 2017. Paper ID: 20158.

53. IEEE Learning Technology Standards Committee (LTSC). *IEEE standard for learning object metadata*. IEEE Std 1484.12.1-2002, pages i–32, 2002.

54. Clarke, P.J., Allen, A. A., King, T. M., Jones, E. L., & Natesan, P. Using a Web-Based Repository to Integrate Testing Tools into Programming Courses. *In Proceedings of the ACM international Conference Companion on Object Oriented Programming Systems Languages and Applications Companion.* SPLASH 2010, pages 193-200.

55. Kitchenham.B.A and Charters.S., 2007. *Guidelines for performing Systematic Literature Reviews in Software Engineering*. Technical Report EBSE 2007-001. Keele University and Durham University Joint Report.

56. Narasareddygari.M.R., Walia.G , and Radermacher.A Gamification in Computer Science Education: a Systematic Literature Review. *In the Proceedings of the 125th ASEE Annual Conference & Exposition. – Computers in Education Division*, June 2018. Paper ID: 22808

57. Hooper, S., & Hannafin, M. J. The effects of group composition on achievement, interaction, and learning efficiency during computer-based cooperative instruction. *Educational Technology Research and Development*, 39(3), 27–40.

58. Hiltz, S. R. *The virtual classroom: learning without limits via computer networks*. Norwood, NJ USA: Ablex Publishing Corporation.

59. Drave W A (2000) *Teaching online LERN Books*, River Falls, Wisconsin.

60. Hurst, B., Wallace, R. & Nixon, S.B., 2013, 'The impact of social interaction on student learning', *Reading Horizons* 52(4), 375-398.

61. Matocha.J.,, Camp.T, and Hooper.R., Extended analogy: an alternative lecture method. *In Proceedings of the twenty-ninth SIGCSE technical symposium on Computer science education*, Daniel Joyce and John Impagliazzo (Eds.). ACM, New York, NY, USA, 262-266. DOI: https://doi.org/10.1145/273133.274309.

62. Hamari, J., Koivisto, J., & Sarsa, H. (2014). Does gamification work? - A literature review of empirical studies on gamification. *In Proceedings of the 47th Annual Hawaii International Conference on System Sciences*, (pp. 3025-3034). [6758978] IEEE COMPUTER SOCIETY PRESS. https://doi.org/10.1109/HICSS.2014.377

63. KhaleelF.L., Sahari.N., Meriam.T.S., Ismail.A., The study of gamification application architecture for programming language course, ACM IMCOM 2015 – *In the Proceedings of Association for Computing Machinery*, Inc, 2015. a17.

64. Jurado.J.L., Fernandez.A., and Collazos.C.A.. 2015. Applying gamification in the context of knowledge management. *In Proceedings of the 15th International Conference on Knowledge Technologies and Data-driven Business*. ACM, New York, NY, USA, , Article 43 , 4 pages. DOI=http://dx.doi.org/10.1145/2809563.2809606

65. Tiffany.B., Heather.R., Eve.P., Amanda.C., Alex.G., Game2Learn: building CS1 learning games for retention. *In the Proceedings of the 12th annual SIGCSE conference on Innovation and technology in computer science education*. New York, NY, USA: ACM, S. 121-125.

66. Buisman, A.L.D., van Eekelen, M.C.J.D.: Gamification in educational software development. *In the Proceedings of the Computer Science Education Research Conference*, VV, 9–20 (2014)

67. Barata, G., Gama, S., Jorge, J., Gonçalves, D., 2013. Improving participation and learning with gamification. *In: Proceedings of the International Conference on Gameful Design, Research, and Applications* (Gamification '13). ACM, New York, pp. 10–17.

68. Krause.M., Mogalle.M., Pohl.H., and Williams.J.J., A Playful Game Changer: Fostering Student Retention in Online Education with Social Gamification. *In Proceedings of the Second (2015) ACM Conference on Learning @ Scale (L@S '15).* ACM, New York, NY, USA, 95-102. DOI: https://doi.org/10.1145/2724660.2724665.

69. Lehtonen.T., Aho.T., Isohanni.E., and Mikkonen.T., On the role of gamification and localization in an open online learning environment: javala experiences. *In Proceedings of the 15th Koli Calling Conference on Computing Education Research* (Koli Calling '15). ACM, New York, NY, USA, 50-59. DOI:https://doi.org/10.1145/2828959.2828973.

70. Harrington.B., 2016. TrAcademic: Experiences With Gamified Practical Sessions for a CS1 Course. *In Proceedings of the 21st Western Canadian Conference on Computing Education*. ACM, New York, NY, USA, , Article 25 , 2 pages. DOI=http://dx.doi.org/10.1145/2910925.2910932

71. Wallis.P, Martinez.M.S., Motivating skill-based promotion with badges. *SIGUCCS* 2013: 175-180.

72. Auvinen, T., Hakulinen, L., & Malmi, L. (2015). Increasing students' awareness of their behavior in online learning environments with visualizations and achievement badges. *IEEE Transactions on Learning Technologies*, 8(3), pp. 261-273. Available online: https://doi.org/10.1109/TLT.2015.2441718..

73. Denny.P., The Effect of Virtual Achievements on Student Engagement, *In Proceedings of CHI 2013: Changing Perspectives*, April 27–May 2, 2013, Paris, France, pp. 763-772.

74. Ibanez, M., Di Serio, A., & Delgado-Kloos, C. (2014). Gamification for engaging computer science students in learning activities: a case study. *IEEE Transactions on Learning Technologies*, 7(3), 291–301

75. Fitz-Walter.Z., Tjondronegoro.D., Wyeth.P., Orientation Passport: using gamification to engage university students. *In the Proceedings of OZCHI 2011*: 122-125.

76. Claypool.K.,Claypool.M., Teaching software engineering through game design, *In the Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education*, June 27-29, 2005, Caparica, Portugal

APPENDIX. DEBUGGING LO

This Learning Object provides an idea on different types of errors and gain an understanding on the process in which the students can go about debugging them. The title of the LO is Debugging. This gives practical knowledge of the errors that could occur during writing computer programs. The LO is included with a Reference page. Additionally, there is a practical and a real quiz to judge the knowledge of students. The general page consists of Name, Subject of LO and description of the LO would let the users know what they would be learning about the LO, Date created, Authors of the LO and also, they can rate the LO. First page provides the introduction to the types of errors and how they could appear in the coding are explained in this page.

Debugging

Debugging

Previous | Next

Debugging Intro - Page 1 of 6

Computer code is written by humans. Humans are less than perfect, likewise their computer programs are also less than perfect. Computer code may contain different types of bugs.

They can be classified as

- Syntactical errors
- Semantic errors

Let us get a brief idea on these errors that are mentioned above:

Syntactical errors: The most commonly encountered errors are syntactic errors. They occur when the written code does not follow the rules of the language. These kinds of errors can be easily detected while using compiler or interpreter, which will produce an error message. Also, when using an IDE, these errors are highlighted indicating that there is something wrong with that part of code. The image below shows an example of an IDE highlighting improper syntax:

```
String name="name1";
int number=1
System.out.println("Name is: "+ name+ " Number is: "+number);
```

The red underline indicates that there is a syntactical error in the code.

Semantic Errors: When the written code is in the correct format, a program may still not give expected results. This may be because of a programmer using an incorrect operation, or performing operations in an incorrect order. These types of errors are called semantic errors and are not caught by the compiler or interpreter.

```
public double Area(double width, double height){
    return width+height;
}
```

If we consider the above example, here the semantic error is that there should be a **multiplication** operator between width and height instead of an **addition** operator. This code will compile and run, but will produce an incorrect result. Semantic errors eventually end up in wrong output. Because these errors are an error in the logic of the program, they are also known as logic errors.

Figure A.1: 1st page of Debugging LO

Second page talks about what is debugging and what could be done using the debugger.

What is debugging - Page 2 of 6

Debugging a program is the process of detecting and removing semantic errors from a program. While this can be done by hand or by other methods, it is more effective to use a type of program called a debugger. A debugger is a powerful tool that helps a programmer to see if their program is working as expected and to help programmers detect the semantic errors present in the code. Because computers are capable of performing millions or even billions of calculations every second, it is impossible for a human to follow the flow of a program in real time. Debuggers allow programmers to pause the execution of a program while it is running, manually control the processing of instructions, and to inspect or modify the memory contents of a program.

Using a debugger, we can do following:

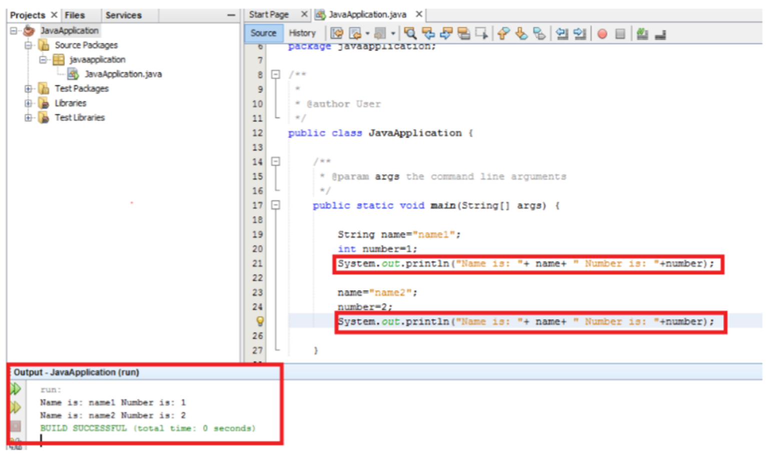
- Inspect the values stored in variables.
- Insert break points to indicate where program execution should pause.
- Manually control the execution of instructions

Knowing the values of the variables:

Let us write a program where there are two variables

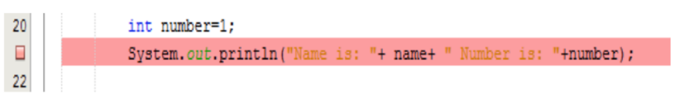
1. String
2. Integer

The original program along with the result is given below:



Before running the debugger, it is first necessary to set a break point so that we can indicate on which line of code the computer should stop executing instructions and hand over control to the user. In this example, we use the NetBeans IDE, but the approach is similar in other environments.

If we would like to pause execution of the code at the first print statement, then we would set a break point for that line by clicking on the corresponding line number (or right-clicking and selecting toggle breakpoint). We can tell that a breakpoint has been set because a red square appears over that line number and that line gets highlighted in a pink color. We can have multiple breakpoints in our program, but breakpoints can only be placed on lines in our program that contain executable code.



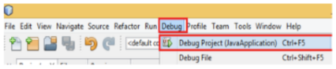
Once we have set a breakpoint, we can start the debugger and debug the code.

Before getting into debugging, let us get a brief know about the windows that appear during debugging.

Figure A.2: 2nd page of the LO

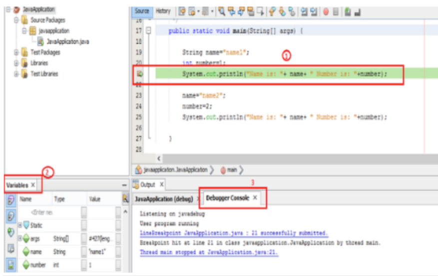
Page 3 talks about how to start debugging which means where we could find the debugging window. If the window closes unexpectedly, how to open the window. This is shown below:

After setting a break point, go to the menu bar and select Debugà Debug Project (....)



Note that if your project contains multiple classes with main methods, you may want to select Debug File instead.

Then two other windows will be opened. If these windows do not appear or you accidentally close them, you can reopen them by selecting the appropriate options from the Windows à Debugging sub-menu.

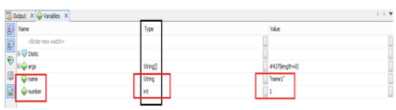


1) The break point where we want to halt execution of the code. The green highlighting lets us know that the debugger has reached the line of code in the program and will not proceed until instructed.

2) The Variables window shows all variables that exist within the current program scope, what values they have, and other information about them.

3) The Debugger Console window provides messages from the debugger. In this example is telling us the debugger has reached the breakpoint at line 21 in the code and has stopped executing code.

Now that we discussed how to add a breakpoint in our program and start the debugger, we will discuss the Variables window in detail.



The leftmost column shows a list of identifiers for all variables in the current scope of the program. Note that some of these variables have a + symbol next to their name which can be clicked to expand them. The Type column indicates the data type for each variable and the rightmost Value column indicates what value is being stored in each variable at that point of time.

In our example, at line 21, we have two variables, *name* and *number* whose data types are String and int respectively. We can see that the value of *name* is currently "name1" and that the value of number is 1.

If you couldn't find the variables window, then go to **the window à Debugging sub-menu**.

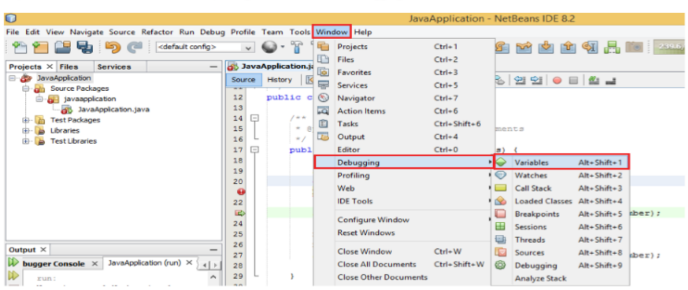


Figure A.3: 3rd page of the LO

Fourth page talks about how the Debugger Console Window looks like.

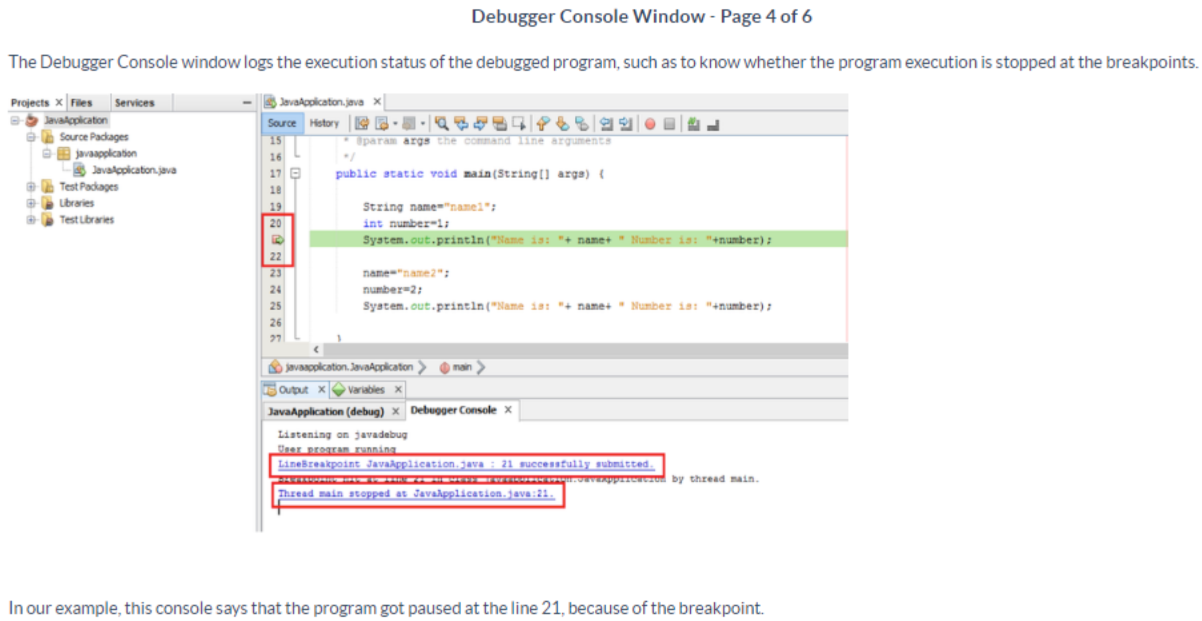


Figure A.4: 4th page of the LO

Fifth page explains how a user could walk through the code in different ways using Step Over, Step Over Expression.

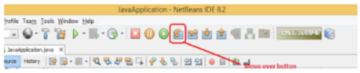
Walking through the code - Page 5 of 6

Once the execution reaches the breakpoint and gets paused, we have several ways to resume the execution. The resume can be done in different ways and these are discussed below:

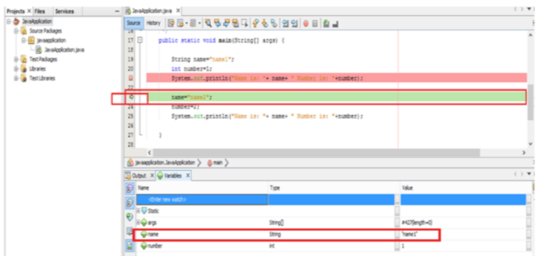
Step Over:

This executes the current line of code and moves to the next line of code in the file.

This is present in the top panel, as shown below:

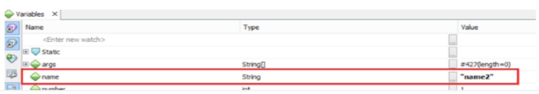


When we press this button, the compiler moves to next line of code:

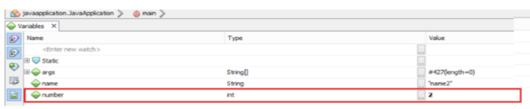


The green bar indicates that the control has moved to the next step.

Once the name variable line has been executed and when the variables window is checked, you can see that the value of the name variables has been changed.



Once again when we move over to next line of code, ie., to number variable and this number variable statement is executed, you can also see the changed value of number variable in the "variables window".

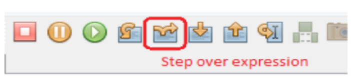


The main use of this is, when the program is halted one can check if the variables are carrying expected values or is the logic functioning as expected using the variables window, watch window or different types of windows presented by **NetBeans**.

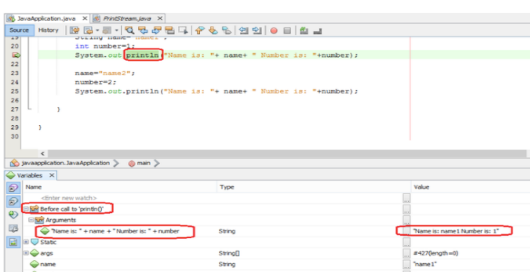
These windows as discussed before are presented in the window panel à debugging à sub-menu windows that need to be displayed.

Step Over Expression:

Step Over Expression is completely different from Step over button, which we discussed before.



What step over did is, it executed the entire "println" expression to print the values of the variable. But, step over expression will execute each part of the println statement individually.



In the above figure, we can see that the println statement is highlighted, so in the next step it is going to execute "("Name is: "+ name+ " Number is: "+number)", one by one inside this line.

If we now look at the variables window, you can see "Before call to 'println ()', where the program is suspended before this is called.

In the below "arguments' you are going to see the expression that is getting to be executed and it gives you a rough idea of what value would that expression would give you.

Figure A.5: 5th Page of the LO

Finally, the last page which is page 6 provides a brief overview of what we was explained in the LO.

This gives a brief overview of what we have done till now.

From the previous example,

1. Put **breakpoints** wherever you think needed.
2. Once the program pauses at the breakpoints, now **move over** to next lines of code to see if the code is functioning as expected or not using different windows provided by debugging. For example, variable window, debugger console, output window, etc.
3. If not, change that piece of code and make sure that it is providing required output.
4. Repeat the steps until you find it ok.

Figure A.6: Overview of the LO

Once the content is developed, next comes the hardest part of developing LO which is building the quiz part. Each LO that is developed in SEP-CyLE has two components of quiz, which are the practice quiz and real quiz which are the next sections of LO. This is the assessment for the students to see how well they understood the content of the LO. So, for debugging in the practice quiz, there are 10 multiple choice questions along with the correct answers. The main motivation of this practice quiz is to get the student ready for the real quiz that will be part of their virtual points. The figure below shows the practice quiz questions one through 5 along with the multiple-choice answers.

Practice Quiz:

1. Among the following options, which has semantic error?
   a. Int a=10
   b. calculateArea(double w, double h){
      w+h;
      }
   c. System.out.println("");
   d. None of the above.

2. Which window shows about the current line of execution?
   a. Output
   b. Debugger Console
   c. Variable
   d. All of the above

3. Which button is used to move to the next line of code?
   a. Step into.
   b. Step out
   c. Step over expression
   d. Step over

4. What kind of error is the following expression?
   X= {2+6;
   a. Semantic error.
   b. Syntactic error.
   c. Both a & b.
   d. None of the above.

5. When the breakpoint is added to the code, then the color of the statement would be _____?
   a. Red.
   b. Green.
   c. Yellow.
   d. Pink.

Figure A.7: Practice Quiz Part-1

The figure A8 illustrates the multiple-choice questions 6 to 10. This is the continuation of the practice quiz from above figure A7.

6. What kind of error is the following?
   int i = “hello”;
   a. Syntactic error.
   b. Semantic error.
   c. Both a & b.
   d. None of the above.

7. A debugger is used to detect _____________?
   a. Syntactic errors.
   b. Semantic errors.
   c. Both a & b
   d. None of the above

8. With the debugger, we cannot do the following:
   a. Inspect the values stored in variables.
   b. Inspect break points to indicate where program should be paused.
   c. Find if there are any syntactic errors in the code.
   d. All of the above

9. The columns that are present in the “Variables” window?
   a. Name.
   b. Type.
   c. Value.
   d. All of the above.

10. Where can we find the variables window in the menu-bar?
    a. Debug→ variables.
    b. Window→ Output.
    c. Window→Debugging→ sub-menu
    d. None of the above.

Figure A.8: Practice Quiz Part-2

The next section of the LO is the real quiz. This quiz also contains 10 multiple choice questions which also includes correct answers. The main objective of this quiz is to determine the student’s retention of the learned content. Below figure shows the real quiz questions from 1 through 10.

Real Quiz:

1. Among the following options, which has syntactic error?
    a. Int number;
    b. If(a=10)
    c. For(i=0 i<10;i++)
    d. None of the above.
2. Which window tells you about the values of the variables?
    a. Output
    b. Variable
    c. Debugger Console
    d. All of the above
3. Which button is used to execute each part of the expression?
    a. Step over expression
    b. Step into.
    c. Step out
    d. Step over
4. When we are going through the execution of the code, ______ color bar indicates that it is getting executed.
    a. Red.
    b. Green.
    c. Yellow.
    d. Pink.
5. What would be the output of the following code?

```
int s = sum (10,5);
public static int sum (int a, int b) {
  return a * b;
}
```

    a. 15.
    b. 5.
    c. 50.
    d. 2.
6. With the debugger, we can do __________
    a. Find if there are any missing colons in the statements.
    b. If the syntax of the for-loop statement is correct or not
    c. To see if the method is functioning as expected or not
    d. None of the above
7. What kind of errors can be detected using compiler?
    a. Division by zero.
    b. Errors in expressions.
    c. Type incompatibility.
    d. None of the above.
8. Syntax errors are _____
    a. Errors that are due to improper use of rules of language.
    b. Errors that are due to the wrong usage of code.
    c. Both a & b.
    d. None of the above.
9. Which of the following is semantic error?
    a. Array out of index.
    b. Division by zero.
    c. Both a & b.
    d. None of the above.
10. A debugger allows us to
    a. Execute the program one at a time.
    b. Execute the program until it reaches the breakpoints.
    c. Helps to know the content of the variables anytime during the execution.
    d. All of the above.

Figure A.9: Real Quiz

After the quiz section, we have references section where the author can add the references from which all the data has been gathered while creating the learning objective. Final section of each LO would be comment section, where the students can comment if they liked the LO or not, suggestions from students are accepted in order to improve the quality of an LO.