

Coding as a Literacy Practice in Adult Learning Communities

Eleanor Smith

Department of Graduate and Undergraduate

Studies in Education

Submitted in partial fulfillment

of the requirements for the degree of Master of Education

Faculty of Education, Brock University

St. Catharines, Ontario

© Eleanor Smith, 2023

Abstract

This study considered how computing courses for adult learners might be customized to effectively address their reasons for learning to read and write computer code. The view of coding as a literacy practice is the key theme in this study. Street's (2006) ideological model of literacy along with the perspective of computational participation, are theoretical models used to explore coding as a literacy practice (Kafai & Burke, 2017). Through the vehicle of action research, this study focused on analyzing the delivery of an introductory web languages coding course for female immigrants. This study drew from both the student and teacher perspectives. The study used student feedback collected from online class survey questionnaires and semi-structured interviews. The study also incorporated the teacher's field notes, a course summary report, and the Teaching Perspectives Inventory survey results (Collins & Pratt, 2011). Findings from this study include these areas of insights: 1) students' views on the benefits of learning coding, 2) the language and communication challenges students faced, and 3) an overview of some effective teaching tools and approaches. Based on these findings, there is a discussion that considered possible issues related to student engagement in learning web language coding. Included are sections on implications for practice and future research.

Acknowledgments

I would like to express my sincere gratitude to everyone whose help and support contributed to this major research paper. First and foremost, I would like to thank my supervisor, Dr. Diane Collier, for not only supporting and advising me on the many topics relating to this paper but also introducing me to literacy studies in the 2019 fall course which allowed me to see educational problems in a new way. I would like to thank my second reader, Dr. Steven Khan, who offered meaningful insights and made me aware of the opportunity to participate in the Coding, Computational Modelling & Equity in Math Education Symposium in the spring of 2023. This was a highlight of my year, and attending gave me ideas that influenced this paper.

I am very thankful to the COSTI organization, particularly Patricia Veloso, Snezana Gabric, and Joseph Padro for their help in bringing me into the COSTI organization to teach their Introductory to Coding Course. More thanks go to Joseph for helping me obtain approval from the COSTI's organization to conduct my research with COSTI's students. Research opportunities like this are not easy to acquire. I consider myself lucky to have had the opportunity to conduct this research through the COSTI organization. Joseph played a role in advising on the design of the student survey which was a major part of this study.

I owe many, many thanks to all the students I worked with from the fall of 2022 until spring of 2023. It was my pleasure to have been your instructor, and I am sure you taught me much more than I taught you. Thanks to all the students who completed the course survey. Your feedback and comments provided were very helpful and motivating. Special thanks go to the four extraordinary women who interviewed with me. Your contributions made this paper a more insightful one. Your words will stay with me.

I would like to thank members of my family. My father, Dennis Goldsberry, proofread many pages of my writing over many months and always told me what he thought. I am thankful

to my late mother, Mary Goldsberry, for being a model of hard work, disciplined focus and lifelong learning. Dan, my spouse of over twenty-five years, listened to my ideas continuously and reminded me that I could complete this paper when I most needed to hear it. To my two daughters, Bridget and Aurora, I have seen tremendous growth and discipline in both of you during the many months I was working on this paper. Both of you have affirmed my choice to try and learn new things and motivated me to persevere in the writing of this paper.

Table of Contents

	Page
Abstract	ii
Acknowledgments.....	iii
List of Tables	vii
List of Figures	viii
 CHAPTER ONE: INTRODUCTION TO THE STUDY	 1
Personal Story	2
Starting Reflections on This Study	7
Outline of the Remainder of the Document	9
 CHAPTER TWO: REVIEW OF RELATED LITERATURE.....	 11
High-Level Conceptions of Coding and Literacy	11
Street's Models of Literacy: Autonomous and Ideological	22
Coding Literacy as Conceptualized Through Computational Perspectives.....	27
Matching Computational Perspectives With Models of Literacy.....	33
 CHAPTER THREE: RESEARCH METHODOLOGY	 39
Purpose of the Action Research Study.....	39
Action Research Cycle Implementation and Phases.....	40
Participants and Site.....	43
Data Collection	46
Data Analysis	51
Ethical Considerations	54
Limitations of the Study.....	55
 CHAPTER FOUR: PRESENTATION OF THE FINDINGS AND DISCUSSION.....	 58
Students' Varying Perceptions of the Benefits of Learning Coding.....	59
Language and Communication Barrier as a Distinct Aspect of the Learning Environment	66
Effective Teaching Tools and Approaches	73
 CHAPTER FIVE: SUMMARY, DISCUSSION, AND IMPLICATIONS	 90
Summary	90
Discussion.....	91
Implications for Practice	98
Implications for Future Research.....	100
Concluding Remarks.....	101
 References.....	 103

Appendix A: Online Survey Questionnaire	112
Appendix B: Sample Semi-Structure Interview Prompts	116
Appendix C: Informed Consent Online Survey	118
Appendix D: Informed Consent Interview	120
Appendix E: Email Invitation for Interviews.....	123
Appendix F: Ethics Clearance Letter	124

List of Tables

Table	Page
1. Arguments for Learning to Read and Write Code	15
2. Three High-Level Conceptions of Literacy	18
3. Teaching Practices to be Applied and Evaluated.....	41
4. Coded Data Examples.....	52
5. Important Coding Characters and Terms in the HTML and CSS languages.....	70

List of Figures

Figure	Page
1. An Autonomous View of Computational Literacy	36
2. An Ideological View of Computational Literacy.....	36
3. Macintyre (2012)'s Action research cycle	43
4. TPI Survey Results	48
5. Querying Coded Data Observations Using Sql Server Management Studio Software	54
6. Question 9 from the Online Survey Questionnaire.....	67
7. Question 11 from the Online Survey Questionnaire.....	68
8. Codepen Development Environment: Real-time view	76
9. Codepen Countdown Timer Program	79
10. Codepen Countdown Timer Program	79
11. Codepen Program with Basic Styling.....	81
12. Codepen Program with Styling.....	81
13. Question 6 from the Online Survey Questionnaire.....	84

CHAPTER ONE: INTRODUCTION TO THE STUDY

This study examined computer coding as a literacy practice in adult learning communities. Understanding how to read and write computer coding serves a broad range of beneficial purposes that include obtaining favourable employment, supporting social progress, and providing a means of communication and expression (diSessa, 2018; Guzdial, 2015; Vee, 2017). In addition, viewing coding as a literacy rather than a rote basic skill has the potential to provide greater insights into developing more effective instructional practices to engage wider audiences of people who would have uses for coding knowledge (Vee, 2017). This study incorporates an ideological perspective of literacy (Street, 2006). An ideological perspective sees coding literacy as a dynamic that changes according to the audience, and it also regards the social aspects such as the relationship teachers have with their students as critical to students' success in developing coding knowledge (Street, 2006; Vee, 2017).

For this study, I conducted action research to examine how to better support a specific audience of adult students in learning to read and write web page code. The participants were my students who are also female immigrants to Canada. They were enrolled in a 5-week introductory coding program through COSTI Immigrant Services, a non-profit organization that provides training to immigrants to help them establish their lives in Canada. One of my objectives was to develop a more nuanced understanding of how these women saw learning coding as a way to help them progress and establish their new lives in Canada. Another aim was to explore the specific challenges these students faced when learning to read and write code. Through examining their reasons for learning coding as well as the obstacles they met, I hoped to develop tangible pedagogical insights that could make me a more informed coding teacher capable of helping a wider group of students.

This first chapter provides a reflexive discussion that describes how my education, teaching experience, and other work experiences led me to examine coding as a literacy practice in adult communities. In this chapter, I have reflected on my experiences learning coding as a young adult and also my experiences teaching coding to distinctly different groups of adult students. While reflecting on my personal story played an important role in shaping the direction of my study, I have highlighted academic insights from both literacy studies and computing education that supported my inquiry.

Personal Story

In the last twenty-five years, I have learned programming languages such as Visual Basic, sql, and JavaScript and later taught these subjects in different adult education programs. I first became a computer systems/software engineering student in 1996 at Mohawk College with the objective of securing gainful employment. Prior to my computer studies, I had been successful in my humanities undergraduate studies translating ancient Roman literature. I had a general notion that my academic success with a classical language was an indication I had the intellectual ability to learn how to write coding instructions in software languages such as C. What I did not think about at this time and still have yet to fully realize are the vaguer requirements such as the ability to collaborate with others that play an important role in developing coding knowledge.

Coding may have not been the most difficult knowledge to obtain, but my own learning path was not a straight easy path nor without frustration. I remember yelling and disturbing my neighbours on Herkimer Street in West Hamilton late in the evening because, after many hours, I was unable to get my COBOL program to compile for an assignment that was due the next day. Never had a difficult passage from Livy or Virgil evoked such loud, negative emotions from me.

As a Classics student, I often made mistakes in my translations and missed meanings in the literature. Led by experienced professors, the social exchanges in class discussions increased my understanding and my interest in ancient literature. Looking back, I recognize that my professors, having the benefit of pedagogical content knowledge developed through decades of study and teaching experience, had a nuanced understanding of how to guide students through ancient literature (Shulman, 2013).

In contrast, my path to learning coding languages over six semesters was less stimulating, often testing my motivation to persevere. Class sessions were often tedious focusing on low-level details and syntax. My teachers, subject matter experts from industry rather than masters at teaching, would work through examples that appeared simple when demonstrated but gave me a lot of trouble when I tried to replicate myself. Despite a learning process that often felt unproductive, I did experience elation when I discovered how to correct syntax and logic errors to produce a functioning program. My motivation increased as it occurred to me that I was learning a creative skill. Over the years in actual practice in the workplace, it was a good feeling for me to recognize that I had a creative skill that allowed me to use computing languages in ways that not everyone around me could. A leading computing education researcher, Mark Guzdial (2015) has mentioned a study in which women in particular were surprised to note that they found coding a creative activity.

After feeling I had endured a lot of suffering in making my way through a computing technology program, I did achieve success in securing a job in technology soon after graduating at the beginning of the millennium. My first corporate job was as an Oracle database administrator at Dofasco (now ArcelorMittal Dofasco); this job required me to work as a member of a team supporting highly specialized database software that was critical for the

functioning of steel mill operations. In this role, I gained a lot of invaluable experience troubleshooting and optimizing code written by large teams.

A few years later while working full-time, I found myself back in the classroom teaching one of the same coding languages, Visual Basic, that I struggled with early on in my studies. During my early working years, I taught a beginner software development class through Mohawk College's "night school" to mostly reluctant coders from the chemical engineering program who needed this coding course to graduate. These students saw the "night school" option as the less painful option to obtain their credit. In retrospect, I would say these classes went well and were productive, and the structure, 1-hour lecture followed by 2 hours of lab work, privileged hands-on practice over lecturing. I also felt a connection with these students who often expressed how annoyed they were with having to take this class. Just a few years before I was in a similar position to them. I gave students a chance to complain during class about what they did not like about the class and coding. In class, I did my best to respond to them and had the chance to work individually with students during the lab period. Because many of these students responded well to my guidance and feedback, I found teaching these students both rewarding and enjoyable. It was so much better for me to provide coding instruction rather than to receive it. I was able to help a good many of them not only get their course credit but also write and understand some useful programs. One of the programs they developed in the course was one that was quite relevant to them: they wrote software that would calculate their course grade. In order to progress through their studies, they needed a passing grade in this course.

Due to the demands of starting a family while working full-time, I took a ten-year break from teaching. At the end of this ten-year period, I still enjoyed working with specialized technology but found corporate life draining and unsatisfying. I recognized my desire to make a

career change and recalled my days teaching in Mohawk's night school program fondly. To support a transition into teaching, I started my Master of Education program at Brock University and sought teaching engagements that would provide me with practical experience. In 2018, while starting my graduate education studies, I had the opportunity to deliver other coding instruction such as SQL, a data query language. These students were highly capable working professionals who were distinctly different from the students I taught a decade earlier. This engagement was part of McMaster University's Big Data and Data Science continuing education program. Many of these students were highly focused and articulate professionals seeking skills development that would give them a competitive advantage in the workplace. It was daunting teaching them because they were especially discerning and perceptive. To make learning more effective for them, I needed to gain their interest and respect. I could not bore them with the basics or overload them with many examples. I carefully selected a few good real-world coding examples that required them to extend basic principles.

In 2022, I had a unique opportunity to introduce front-end web coding to women who recently immigrated to Canada. I taught these students through an introductory coding course that was part of COSTI's immigrant services program that helps immigrants to Canada establish new lives for themselves. Many of these women were highly educated and had fled violence and chaos in countries such as Afghanistan, Ukraine, and Venezuela. Many of these students were learning English at the same time; because they were developing English language knowledge, it was often difficult to communicate with them. My experiences teaching these women to code is the focus of my study. Throughout this study, I learned more about their purposes for learning coding and made teaching adjustments based on these observations. Through my engagements

teaching in different programs, I realized I had to make adjustments to support each group of students.

Despite their distinct characteristics and needs, there was a common thread between different groups of students I worked with. I encountered students in all these programs whose learning experiences matched mine from twenty-five years ago and who reported high levels of frustration. Among all of these groups of students, I saw the joy after struggling that some students experienced in this learning process. I did have other students give up, perhaps deciding that learning coding is too laborious to learn with too little reward and not for them. It is hard to discern the exact reason or the combination of reasons a student decides to give up on learning coding. While some of these reasons may not be connected to instruction, I think drawing from my own experiences, suggest that it is quite likely at least some of these students may have lost their motivation for learning coding due to ineffective teaching and learning practices. From my own experience when I was a student learning different coding languages at Mohawk College, I felt that a teaching and learning emphasis on low-level details, decontextualized examples, and laborious efforts to write programs from scratch significantly dampened my motivation. Over the years, I have had students from different beginner coding courses indicate that they are giving up because the instruction I provided did not work for them. One of the most challenging aspects of teaching immigrant women in this study was witnessing the typical steep drops in participation in all of the sessions. While I have never forgotten my own frustrations in learning coding, I recognize that reflecting on my own experiences as a student and teacher is not enough to provide me with insights on improving coding instruction.

Starting Reflections on This Study

Years before I started my MEd program and worked with the participants of my study, I wondered about developing effective ways to teach coding languages to people like me and my students, adults learning how to code to increase their job prospects and professional capabilities. Guzdial (2019) discusses in his blog entries about how to make JavaScript, a language noted for its idiosyncratic features, “teachable.” I wonder if it would help for teachers to adjust the focus from making a coding language “teachable” to making the language “learnable” for students. One starting idea is for instructors to see themselves more as facilitators or coaches rather than supremely skilled knowledgeable authorities. Those of us like myself who find ourselves teaching coding languages need to offer much more to students than a well-rehearsed demonstration of generating coding instructions. Even the most disciplined and determined among us can only sustain focused attention in such demonstrations for a limited amount of time. I think it does help for the instructor to have techniques to engage in dialogue with students while demonstrating a coding example. I think back to my first-year course taking ancient Greek and how the experience of learning this language was not strained nor mechanical and actually enjoyable for me. This professor, with many years of teaching experience, and who had even taught my mother thirty years earlier, had a strong presence and gave each student a chance to provide a response to drill-type questions. During these exercises, the professor would pause if there was something that seemed to be difficult and would add dialogue and even stories. What he did achieve in his teaching was a social practice around learning a rote skill. I looked to this example from my own life as a model for teaching coding languages as a social practice.

From my own experiences as a student and a teacher, coding instruction tends to be limiting and limited. There were plenty of times during this past year when I was providing

coding instruction that I recognized that I was going through the motions to cover the content and failing to engage many students. The coding courses I took and that I have taught were narrowly focused and did not explicitly address important aspects such as abstract thinking and working collaboratively that are essential for productivity in the workplace. This idea that coding should be taught as a social practice is present in Kafai and Burke's (2017) conception of computational participation. Principles of computational participation challenge a linear approach of expecting students to learn coding by writing their first programs from scratch. When I decided to learn how to write code, I did look forward to the prospect of learning a potentially transformative skill. After a few weeks of class instruction spent on trivial and basic examples, I felt really worn down with my curiosity for learning the coding languages significantly dampened. As an alternative approach to learning coding languages, computational participation privileges modifying code over writing programs from scratch. In today's software-saturated, open-source environment, it is important to be able to comprehend, modify, and alter existing programs (Kafai, 2016). My industry experience supports this idea; it is rare in the workplace to build a software program from scratch. Instead of developing software in-house, workplace organizations often buy software programs and place their efforts toward customizing the software to meet their own specific business needs. Had my teachers shown some real-world examples of software programs written in the language we were learning; they might have helped make learning coding more engaging.

Recognizing coding as a literacy practice offers more suggestions for improving the learning experience for more students. Street's (2006) literacy theories might uncover ideas that would help improve teaching coding. For example, applying Street's theories to coding means recognizing that learning coding is highly dependent on the social exchanges among students and

teachers. According to Street's (2006) framework, coding literacy is also closely connected to the specific objectives of those who wish to read and write code. Looking back on my experience, I adjusted the lessons based on my students' purposes. I tailored content that I thought would best engage each audience. For example, my group of reluctant coders at Mohawk College came to the class most interested in meeting their program requirement so they worked on writing software that would calculate their grade.

In addition, scholars such as diSessa (2018), Wing (2006) along with Kafai and Burke (2017), and Tissenbaum et al. (2019) have all contributed to creating a computational vocabulary that gives us new ways to discuss and examine issues that occur when teaching students to code. What I hoped to do in this study was to combine an analysis of my own experiences with literacy theory and the major computational perspectives to develop a less limiting and limited instructional coding practice.

Outline of the Remainder of the Document

Chapter 2 is the literature review that provides a high-level conceptual discussion of coding literacy and Street's (2006) autonomous and ideological models of literacy. The literature review also includes a discussion of the major computational perspectives that include computational literacy, thinking, participation and action (diSessa, 2018; Kafai & Burke, 2017; Tissenbaum et al., 2019; Wing, 2006). The final section of the literature review maps the major computational perspectives to Street's (2006) models of literacy.

Chapter 3 describes the research methodology I used in examining computer coding as literacy. This chapter includes my research questions and highlights the action research model I used. Included is an outline of the phases of my action research cycle. The following sections

include a description of the participants and site, data collection, and analysis. This chapter also discusses the ethical considerations and acknowledged limitations of my study.

Chapter 4 is the presentation of my findings and discussion based on my experiences teaching web coding to female immigrants. The chapter is divided into three sections that include 1) students' varying perceptions of the benefits of learning coding, 2) language and communication barriers as distinct aspects of the learning environment, and 3) effective teaching tools and approaches.

The final chapter, Chapter 5, expands upon the findings and discussion of Chapter 4. This chapter provides a summary and discussion that draw from academic theory to offer possible explanations for the steep drops in student participation. It includes sections on implications for practice as well as for future research. The paper closes with concluding remarks on what I have learned through this research.

CHAPTER TWO: REVIEW OF RELATED LITERATURE

This literature review will explore computer coding as a literacy practice and will discuss the implications of this view for developing and improving educational practices for adult audiences. Both broad topics, computer coding and literacy, lend themselves to a range of purposes, meanings, and perspectives. To establish the theoretical focus for this discussion, the first section will provide an introductory discussion on coding, a summary of the main arguments for learning coding and academic conceptions of literacy that include Street's (2006) autonomous and ideological models of literacy. Included in this section will be real-life examples of coding education programs for adults that illustrate how the development of coding knowledge varies according to the objectives of its participants. The next section will outline major computing perspectives, such as computational literacy, participation, action and thinking (diSessa, 2018; Kafai & Burke, 2017; Tissenbaum et al., 2019; Wing, 2006). These computing perspectives will be mapped to Street's (2006) autonomous and ideological models of literacy. Through examining coding as a literacy practice, this literature review will provide insights from scholarly literature to inform and improve coding education programs for adult audiences.

High-Level Conceptions of Coding and Literacy

An Overview of Coding

The term coding refers to the reading and writing activity required to map ideas or desired actions to instructions that a computer can interpret and execute (Berry & Kölling, 2014). As one way that humans are able to communicate through computer software, coding is a distinct "species of writing" (Vee, 2017, p. 5). While this emphasis on writing recognizes coding as a generative activity, it is also an interpretative one since coding requires reading ability. Reading code is required for participation in code reviews on collaborative software development projects

(Burke et al., 2016; Kafai, 2016). Reading code is also regarded as a required skill of high-school computer science teachers; these teachers need to be competent at reviewing students' code to provide constructive feedback (Ericson et al., 2015; Guzdial, 2015).

With no lingua franca for code, listing all the languages, tools and applications of coding would be a never-ending task (Vista, 2020). There are thousands of languages and code development supports such as open-source libraries and integrated development environments (IDEs) that are designed for specific uses and contexts (Vee, 2017). Some coding languages commonly included in adult educational programs are Python, Java, SQL, HTML, CSS and JavaScript (Byrd, 2020; Royal, 2017; Guzdial, 2015).

Coding languages are “infinitely adaptable” and “semantically broad, diffuse, and ambiguous” (Portnoff, 2018, p. 35). These descriptors suggest that coding languages may be quite malleable and useful but are complicated to access. Such diversity both in choice and applications of coding languages provides creative opportunities and learning challenges for those aspiring to learn the art of coding. A technological avenue for creativity, coding requires multiple levels of knowledge and skills that include logic, language syntax, tooling and working collaboratively (Franklin, 2019; Guzdial, 2015; Vee, 2017). In order to create a software product, coding knowledge also needs to be coupled with domain or subject matter knowledge. Creating applications that monitor health or financial situations require the translation of specialized field metrics into computer code. Successfully encapsulating business requirements into instructions that a computer can interpret and execute depends on the coder's understanding of the business rules.

A common perspective of coding is as a computing technique to streamline complex mathematical calculations (Guzdial, 2015; Tedre et al., 2018; Wing, 2006). Other emerging and

evolving perspectives of coding are as ways to help improve daily life and to facilitate learning, expression and communication (Burke et al., 2016; Guzdial, 2015; Jacob & Warschauer, 2018; Voogt et al., 2015; Vee, 2007). Analyzing family spending, learning mathematical concepts, creating a digital story and developing a website to support a social cause are some endeavours benefiting from coding activity (diSessa, 2018; Guo, 2017; Vee, 2017). Writing an Excel macro to format data, programming the thermostat in your home, and analyzing academic literature are more examples that demonstrate coding's wide range of uses (Columbia, n.d., Guzdial, 2015)

Compelling Arguments for Learning Coding

Included in the academic literature are discussions of the main arguments for learning to read and write code. This part of the discussion draws from the complementary research of Guzdial, whose expertise is computing in higher education, and Vee, whose expertise is literacy theory. Both argue that coding is something that most, if not all, of us should learn. They have developed their arguments through reviewing important educational thinking that includes the pioneering insights of Papert to the more recently published work of Kafai and Burke (2017). While Guzdial (2015) provides a practical perspective on contextualizing coding instruction for different adult audiences, Vee (2017) gives a theoretical and more expansive view in recognizing coding as an important mode of writing and communication. Vee (2017) cites four arguments that include: "individual empowerment," "learning new ways to think," "citizenship and collective progress" and "employability and economic concerns" (p. 76). Guzdial (2015) also identifies similar reasons such as acquiring jobs, "learning about the world" and "broadening participation" to include women and marginalized people in computing education and projects (pp. 179-180). Guzdial also highlights computational literacy as a key reason for learning to code. Computational literacy has multiple meanings that include the use of computing to increase

productivity, aid learning and support communication and expression (diSessa, 2018; Guzdial, 2015; Vee, 2017). Securing economic stability through acquiring a software development job is one of the more pragmatic reasons for learning to code. As an example, Byrd (2020) describes a coding bootcamp program for marginalized Americans that prepared them for entry into corporate workplaces. For this group, coding was a route to obtaining employment and economic security.

Beyond the employment argument, there are other compelling reasons for learning coding (Guo, 2017; Guzdial, 2015; Vee, 2017). For example, coding can transform the way we think and increase our knowledge of different subjects. diSessa (2018) conceptualizes coding as an effective way to illustrate mathematical and scientific concepts to school-aged children. Not limited to math and science domains, coding activity also develops children's language literacy through digital story-telling coding platforms such as Scratch (Burke et al., 2016). Adults from different professions could code to increase their professional efficiency and effectiveness. Being able to perform text-mining of academic texts can increase the productivity of historians (Columbia, n.d., Guzdial, 2015).

Another good argument for coding involves social progress. Coding pursuits are instrumental in addressing social problems throughout the world. In 2021, the Vaccine Hunters Canada initiative that used code to extract and consolidate information from hospital websites helped millions of Canadians receive timely vaccine updates during the crisis of the pandemic (Shephard, 2023). School-aged Indian girls have coded mobile applications to manage local problems such as water collection and gang rape (Tissenbaum et al., 2019). One of their applications, Paani, helped families avoid long wait-times by alerting them when it was their turn to obtain water from communal sources (Ranjen, 2015; The Logical Indian, 2016). Another

application, Women Fight Back, increased women's security by allowing women to electronically call for help when they perceived physical threats to their safety (Ranjen, 2015; The Logical Indian, 2016). The example of these girls using coding to solve local problems shows coding as a powerful agent for vulnerable groups. Through coding, these girls were able to take control in improving their lives.

Through all of these examples, coding is capable of fulfilling a range of purposes and supporting multiple audiences (Vee, 2017). Coding's many uses support the argument that coding is an important literacy that most of us should develop. The table below (Table 1) provides a consolidated list of academic arguments and examples for learning coding.

Table 1

Arguments for Learning to Read and Write Code

Arguments for learning coding	Key examples from academic literature	Main Theme from Guzdial (2015) and Vee (2017)
Provide new ways of problem-solving and learning mathematical, scientific concepts	Boxer Bootstrap Logo (diSessa, 2018; Guzdial, 2015; Vee, 2017)	Computational literacy Individual empowerment Learning
Provide new ways of communication and expression	Scratch platform for creating digital narratives Markup & styling languages, HTML and CSS to create visually expressive websites	Computational literacy Individual empowerment

	(Burke et al., 2016; Byrd, 2020)	
Increase personal and professional productivity	<p>Liberal arts researchers who mine text</p> <p>Graphic designers who want use to coding to save time in their design work</p> <p>High-school teachers who want to use coding in their teaching</p> <p>(Columbia, n.d.; Guzdial, 2015)</p>	<p>Computational literacy</p> <p>Individual empowerment</p>
Increase job opportunities	<p>Teachers wishing to teach computing</p> <p>Marginalized groups seeking economic prosperity</p> <p>(Byrd, 2020; Ericson et al., 2015; Guzdial, 2015)</p>	<p>Employability</p> <p>Jobs</p>
Support social progress and solve community problems	<p>Vaccine Hunters Canada</p> <p>Paani (queue management for water collection in the Indian slum of Dharavi)</p>	<p>Broadening participation,</p> <p>Social progress</p>

	<p>Women Fight Back application (emergency calling for women in the Indian slum of Dharavi)</p> <p>(Ranjen, 2015; Shephard, 2023; The Logical Indian, 2016; Tissenbaum et al., 2019)</p>	
--	--	--

Conceptions of Literacy

Like the term coding, literacy holds multiple meanings and connotations that can make it challenging to summarize. Looking at both its functional and rhetorical aspects is a way to begin to analyze literacy (Vee, 2017). To start with its functionality, literacy is a dynamic form of knowledge that enables “individuals and social groups to extend their understanding of themselves and their world” (Vincent, 2000, p. 24). Furthermore, possessing literacy means that one has the capability to choose and utilize the appropriate communication methods and tools to suit a particular purpose (Coiro et al., 2014; Vee, 2017). Uber-coder and founder of the Vaccine Hunters Canada initiative, Andrew Young, epitomizes the functional qualities of literacy through applying his training as a web developer to write useful code that collected critical health information from various organizations’ websites (Shephard, 2023). Leveraging his professional skills, Young exploited his knowledge of coding languages in response to the challenges of obtaining timely health information experienced in Canada during the pandemic in 2021. Demonstrating its functional quality, this example shows literacy as involving the ability to adapt knowledge in response to a pressing situation.

Not only a contextual and flexible form knowledge that people develop and use in the pursuit of personal and/or group objectives, literacy is also a rhetorically powerful, social influence as shown through probing Scribner's, Vee's and diSessa's high-level perspectives of literacy (Table 2). These authors help inform how coding knowledge is emerging as a literacy on par with "mass textual literacy" that is practised universally and embedded in formal education (diSessa, 2018, p. 4). Scribner's conception is a general but insightful one, not part of a broader computing discussion. Both Vee and diSessa define literacy in ways that conceptualize coding as a literacy practice. While their descriptions do not explicitly separate the functional and rhetorical aspects of literacy, an explication of these three conceptualizations (see Table 2 below) draw attention to the rhetorical aspects of literacy.

Table 2

Three High-Level Conceptions of Literacy

Author	Literacy description
Scribner (1985)	simultaneously adaptive, socially empowering, and self-enhancing (p. 18)
Vee (2017)	widely held, socially useful and valued set of practices with infrastructural communication technologies (p. 27)
diSessa (2018)	a massive social/intellectual accomplishment of a culture or civilization, where many competing forces, over decades or centuries, eventually settle on a particular representational form for widespread learning, use, and subsequent value (p. 7)

Each authors' description draws attention to different facets of literacy. Scribner (1985) calls her conception "ideal literacy" (p. 4). Her compact and succinct conception views literacy as holding its power through the combination of "adaptive, socially empowering and self-enhancing" qualities (p. 14). Unique to Scribner's (1985) conception is the emphasis on the

individual through the use of the word “self-enhancing” (p. 14). Her conception stands out in acknowledging the personal rewards of possessing literacy. Implicit is the notion that having a particular knowledge or skill set may confer special recognition for an individual. Applying coding knowledge to address social problems can establish a heroic reputation for an individual. Such is the case of Young, founder of Vaccine Hunters Canada, who is recognized as a hero for writing code that automated real-time communication with healthcare websites to obtain the latest vaccine information (Shephard, 2023). Communicating with websites in a way that many of us would not be able to or even describe at a high-level, Young serves as a role model for developing coding knowledge or literacy. Through this example, coding literacy is an adaptive type of computing knowledge.

As another example, literacy’s self-enhancing quality appears in Code.org’s rhetoric on the value of learning to code. Household names, Facebook leader Mark Zuckerberg and Microsoft’s founder Bill Gates, are spokespeople for Code.org and support a persuasive exaggerated narrative that coding literacy is a superpower that has the potential to bring one great influence and recognition. With their identities as self-taught coders, Zuckerberg and Gates serve as high-profile examples of the personal rewards of acquiring coding literacy (Vee, 2017).

In contrast, the definitions highlighted by Vee and diSessa do not focus on literacy in the terms of the individual. Instead, they focus on literacy as a collective social force that has a profound influence on educational thought and practice. Vee(2017) attaches material elements to literacy through mentioning “infrastructural communication technologies” (p. 27). The internet, Google’s software and mobile devices are some examples of everyday technologies that profoundly changed how people communicate and obtain and create information. These infrastructural elements require the acquisition of new practices, knowledge, and skills. diSessa

is more explicit than Vee in citing educational needs by recognizing literacy as requiring “widespread learning” (p. 7).

By calling literacy “massive,” diSessa’s (2018) definition stands out in drawing attention to the grand scale of literacy. Stressing the time dimension, diSessa sees literacy evolving over long periods of time to achieve a collective consensus that a type of knowledge should be universally valued. His definition refers to political dynamics at play by mentioning “many competing forces” that would include different levels of government, corporations, educational organizations, and public sentiment (diSessa, 2018, p.7). Through this description diSessa, like Scribner, presents literacy as something that is malleable and adaptable to changing circumstances and social conditions.

Showing the powerful rhetoric of literacy, Scribner, Vee and diSessa emphasize literacy as a dynamic social influence. diSessa’s language, in particular, describes how literacy achieves much of its power through social perceptions that a type of knowledge is increasingly valuable (Vee, 2017). Evolving over time, literacy develops slowly in response to social and economic conditions, technological advancements, and also through organizational and political interests. Revealing indicators that coding is emerging as a literacy on the magnitude of what diSessa describes are the increased educational efforts, both formal and informal, that support coding literacy. The proliferation of code bootcamps for adults, the development of university coding courses for liberal arts majors and out-of-school educational initiatives such as Code.org, Canada Learning Code and BlackGirlsCode for children provide evidence that a mass coding literacy movement is building (Byrd, 2020; Columbia, n.d., Guzdial, 2015; Vee, 2017).

Negative Implications of Literacy's Rhetoric

As social awareness of coding literacy grows, there is the need to look beyond these high-level conceptions that do not address the problematic subtext attached to literacy. Labelling coding knowledge as a literacy on par with textual literacy greatly increases the social pressure to acquire this type of knowledge (Vee, 2017). Under this assumption, those who are not able to code could be considered illiterate and deficient. Such labelling would be less than motivating for those wishing to learn coding but experience challenges acquiring this knowledge. At best, a societal perception of non-coders as lacking would provide negative reinforcement for learning coding. This popular rhetoric of coding literacy can be even more damaging if there are limited options for developing it. As coding is becoming acknowledged as a literacy, narrow curricular approaches to coding that failed to engage many learners have been common in educational settings including higher education (Guo, 2017; Guzdial, 2015; Vee, 2017). Recognizing coding as a literacy, something that involves developing adaptive knowledge, is a way to challenge the dominance of the isolated skills-based instructional approaches (Street, 2006).

Examining social problems that have emerged from mass textual literacy efforts provides insight to better inform coding educational practices. In the case of mass textual literacy, literacy scholars have regarded its formal schooling practices as classist, detrimental and alienating to minority groups and communities (Carter, 2006; Heath, 1983; Street, 2006; Vee, 2017). Formal education tends to reflect the values of the dominant culture. This situation, in turn, benefits mainstream western groups far more than culturally diverse groups whose values, needs and literacy practices are often overlooked (Carter, 2006; Heath, 1983; Street, 2006; Vee, 2017). For example, Heath (1983) found that the schooling system in Piedmont Carolinas failed to engage and stimulate children from a working-class African American community who already had

developed “creative, highly analogical styles of telling stories” before entering formal schooling (p. 73). Decontextualized schooling practices characterized by a focus on “print in isolation” did not leverage the narrative strengths of these children (Heath, 1983, p. 70). Sadly as a result, many of these students lost interest in school activities and failed to flourish academically in the later grades (Heath, 1983).

In addition, popular ideas about assessing textual literacy helped introduce standardized testing practices that privilege “particular contexts, identities, and knowledge” over others (Carter, 2006, p. 98). With all the uses and variations of coding languages, standardized coding assessments would likely be as limited and narrow and would favour some uses over others. While coding literacy is a social influence that requires more than facile acknowledgments, its development requires substantive insight to build and support flexible and contextualized educational practices (Carter, 2006; Guzdial, 2015; Vee, 2017). In this discussion, coding literacy refers to the ability to read and write computer code to fulfill a wide range of purposes (Vee, 2017). In addition, coding literacy can vary significantly according to the characteristics and objectives of its audience (Guzdial, 2015; Street, 2006, 2009; Vee, 2017). Outlined in the next section, Street’s (2006) work provides a theoretical context for increasing interest and sustaining participation in developing coding literacy among diverse groups of people.

Street’s Models of Literacy: Autonomous and Ideological

Contrasting Street’s autonomous and ideological models of literacy offers insight into ways to facilitate coding literacy among different audiences. Reviewing Street’s (2006) autonomous model of literacy helps highlight less effective ways to develop coding literacy (Carter, 2006; Street, 2006). The autonomous model sees literacy development as independent from social context. It regards the acquisition of literacy as inherently cognitive, asocial, and

general-purpose. This limited approach to developing literacy is also culturally biased because it imposes “western conceptions of literacy onto other cultures” (Street, 2006, p. 2). As pointed out by multiple writers, educational implementations reflecting the autonomous perspective ignore the specific literacies of non-western cultures (Byrd, 2020; Carter, 2006; Heath, 1983). This assumption poses a problem in literacy development because the specific literacies of these groups can serve as scaffolds for developing other literacies such as coding. As shown in Byrd’s (2020) study of African Americans participating in a coding bootcamp program, this group had developed computing knowledge informally through out-of-school experiences. Their computing knowledge was “overlooked funds of coding knowledge” that supported their development of learning web page coding through their participation in a coding bootcamp (p. 446).

As an alternative to the autonomous perspective, the ideological model sees the acquisition of literacy as heavily dependent on the social environment or context in which it is to be practised (Street, 2006). The acquisition of literacy is “always embedded in social practices” and varies according to the particular needs of its audience (Street, 2006, 2). Under the ideological view, literacy involves adaptive decision-making in applying “forms and functions of literacy” to achieve a particular purpose (Coiro et al., 2014, p. 5). For example, those wishing to secure a job to work on corporate websites would learn different coding languages, tooling, and technologies than the young Indian women from Dharavi who experimented with mobile coding technology to manage their daily challenges, such as collecting drinking water. Having different aims, these groups also require different sets of social knowledge. While the young Indian women need to understand their own local customs and rules involved in water collection to map them to code, those interested in obtaining a job need to understand how to market their technical knowledge to employers.

The ideological view also recognizes that the power dynamics and social exchanges among teachers and students have a major effect on literacy development. Genuine trust and respect among teachers and their students increase the productivity of literacy development (Street, 2006; Street, 2009; Vee, 2017). The development of the young women from Dharavi depended on productive relationships with their mentors and knowledgeable authorities (Tissenbaum et al., 2019). In addition, it really helps when teachers find a way to leverage students' experiences in the classroom. Carter (2006) provides one such example of engaging students in writing. In her writing course, students drew from their own experiences and areas of expertise from “work (waiting tables, styling hair, building homes, designing web pages) and play (quilting, painting, playing video games)” (p. 101). Such a pedagogical approach that taps into the lived experiences of learners increases literacy development and demonstrates respect for learners through valuing their own perspectives.

Real-life Examples of Ideological Approaches to Coding Literacy

The next section will provide a discussion of two real-life examples of audiences who illustrate distinctive qualities of an ideological approach to developing coding literacy. An important premise of the ideological model is that general-purpose or standard academic coding instruction is not going to meet the objectives of everyone who wishes to pursue coding literacy. These examples include two groups who have distinctly different objectives for learning coding. This first group involves high-school computing teachers who are learning coding with the objective of increasing their pedagogical content knowledge. The second group involves racialized students who are learning to code for socio-economic advancement. Each group will need a coding curriculum tailored to their own needs and objectives. While both examples do

illustrate the effectiveness of Street’s ideological model of literacy, the example involving racialized participants shows that ideological approaches to literacy can be problematic.

Coding Literacy for the Purpose of High-School Teaching

Guzdial (2015) and Ericson et al. (2015) highlight the US community of high-school computing teachers who require coding literacy to better support “the widespread learning” of computing practices (diSessa, 2018, p.7). A coding literacy program for this audience differs significantly in focus from typical educational programs that prepare those aspiring for software development jobs in industry (Ericson et al., 2015; Guzdial, 2015; Vee, 2017). Unlike professional software developers, high-school computing teachers don’t need to be particularly efficient at writing code from scratch to be effective in teaching coding to their students (Ericson et al., 2015; Guzdial, 2015).

Their coding literacy program distinguished itself in focusing on the needs of teachers whose main objective involved increasing their pedagogical content knowledge (Ericson et al., 2015). This audience of high-school teachers lacking a computer science background did not feel comfortable with the idea of teaching coding to a younger generation (Ericson et al., 2015; Guzdial, 2015). This community of adult learners likely included those who likely saw themselves as “digital immigrants” and felt less equipped to teach coding to their younger “digital native” students (Jenkins, 2007). More importantly, this group did not need to be efficient at writing software applications. Rather they needed to be efficient at reviewing their students’ code. Instead of learning the more complicated software languages and tools that software engineers use to build software, their curricular program used web-accessible tools and editors with drag and drop features. Avoiding unnecessary frustrations involving using specialized software, these accessible tools allowed teachers to efficiently learn coding syntax

and concepts. Materials used such as the interactive editors contributed to the timely development of their ability to review code which increased their confidence to engage in productive interchanges with students. This example demonstrates the strength of an ideological approach in specifically targeting its audience's objectives.

Coding Literacy to Support Socio-Economic Advancement

While the next real-life example shows the effectiveness of an ideological approach in helping its participants meet their objectives, it also exposes another dimension of an ideological approach to coding literacy development. This example includes low-income and African American adults who through training in a non-profit coding bootcamp are better positioned to advance their socio-economic status (Byrd, 2020). The program's objective involved increasing the economic prospects of its marginalized and racialized community of adult learners. Participants' strong motivations to improve their economic prospects influenced the educational direction of this three-month intensive coding literacy program (Byrd, 2020). Requirements for this group included not only acquiring working knowledge of web development coding languages and practices, but also included developing social knowledge of corporate workplace culture (Byrd, 2020). Clearwater educators supplemented its coding curriculum with other soft skills training to help their students adapt to the foreign culture of workplace environments. For Clearwater's learners, coding literacy requires their social awareness of employer expectations. Implicit in Clearwater's program design was an uncomfortable notion that these students must conform and submit to a dominant's culture to have a chance at fulfilling their objectives of having a better economic life. Byrd (2020) vaguely and curiously alludes to Clearwater's efforts to equip their students with the required social skills:

To better prepare students for the culture of whiteness in the tech industry, Clearwater developed a new curriculum that emphasized teaching soft skills and

expects students to behave as workers in the classroom. (p. 433)

Although the design of this literacy program follows Street's advice and did reflect an understanding of the participants' reasons for pursuing coding literacy, this example shows that ideological implementations of literacy can be problematic (Vee, 2017). In contrast with the previous example of high school computing teachers, this example exposes a dark side of an ideological approach to literacy. While certainly more effective in targeting the specific needs of participants, not all ideological approaches to coding literacy are socially progressive, particularly for those aspiring for economic advancement.

Coding Literacy as Conceptualized Through Computational Perspectives

This next section will further examine how aspects of Street's autonomous and ideological models of literacy map to the major computing perspectives of computational literacy, participation, action and thinking (diSessa, 2018; Kafai & Burke, 2017; Tissenbaum et al., 2019; Wing, 2006). Examining the intersection between literacy theory and these computational views serves to strengthen the argument that educational programs should regard coding as a contextualized literacy practice.

Present in the scholarly computer coding literature are references to these terms: computational literacy, computational thinking, computational participation, and computational action (diSessa, 2018; Kafai, 2016; Tissenbaum et al., 2019; Wing, 2006). While all of these perspectives see coding as providing access to "new ways to think," they hold distinct meanings and implications (Vee, 2017, p. 76). As an alternative frame for computational thinking, computational participation and action are emerging ideals that see coding as a vehicle for social impact and progress. These two newer perspectives also stand out as challenging traditional-leaning notions of coding as a cognitive, asocial activity (Kafai, 2016; Tissenbaum et al., 2019).

The oldest term, diSessa's (2018), computational literacy, provides a broad perspective for developing contextualized coding programs (Guzdial, 2015; Vee, 2017). Wing's (2006) seminal term, computational thinking, sometimes equated with computational literacy, focuses on the cognitive and dispositional qualities required for learning to read and write code (diSessa, 2018; Jacob & Warschauer, 2018; Vee, 2017). While there are academic debates involving the relevance and limitations of these perspectives (Floyd, 2022), below is an interpretation of how these perspectives help support the view of coding as a literacy practice.

Computational Literacy: A High-level Objective

Computational literacy is an overarching objective of computing and coding education programs. Having computational literacy reflects the ability to use computing skills such as coding for the purposes of expression and communication (diSessa, 2018; Guzdial, 2015). For instance, knowledge of markup and styling languages, HTML and CSS, allows the creation of visually expressive websites (Royal, 2017). Using these coding languages to create striking web displays would suit a broad range of uses and audiences. Using HTML and CSS languages to communicate an idea on a webpage would qualify as computational literacy. Expression through the medium of web pages is a capability that arguably most of us should have given the ubiquity of internet communications.

More importantly, what constitutes computational literacy is contextual and dependent on the environment and culture in which it is practised. Computational literacy varies according to tools, practices, and purposes of a group and implies computing productivity (Vee, 2017). In workplace contexts, possessing computational literacy means having a solid understanding of the limits and capabilities of an actively used software management system (Guzdial, 2015). In learning situations, computational literacy means being able to exploit computing/coding

platforms to learn another discipline or to explore a complex subject (diSessa, 2018; Guzdial, 2015). At the essence of diSessa's (2018) conception of computational literacy is a transformative type of learning. As examples, coding languages such as Boxer and Scratch are two coding languages that aid distinctly different types of learning. Boxer facilitates and illustrates mathematical concepts while Scratch helps develop storytelling ability (Burke et al., 2016; diSessa, 2018; Guzdial, 2015). As a broad concept with a high degree of variability, computational literacy *should be* a high-level, important objective of coding education programs. Engaging in computational participation, action, and thinking are possible ways to achieve computational literacy.

Pathways to Achieving Computational Literacy: Participation, Action and Thinking

The next section will provide an overview of major computing perspectives that show various pathways for the development of computational literacy. Each perspective: computational participation, action, and thinking provide insights and represents possible approaches to achieving computational literacy (Kafai & Burke, 2017; Tissenbaum et al., 2019; Wing, 2006).

Computational Participation: Coding as Collaborative Exchanges

Computational participation (Kafai & Burke, 2017) is a recent progressive perspective that sees learning coding as a practice heavily dependent on collaborative exchanges. Regarding coding as a dynamic and social activity, this view aligns with the high-level conceptions of literacy discussed in section 1. Kafai (2016) lists three ways that contemporary coding practices exemplify computational participation. These include 1) coding shareable applications, 2) developing code through knowledge available in online coding communities and 3) remixing code. Common contemporary uses of coding include creating digital stories and video games that

are developed with the main intention of sharing with others (Burke et al., 2016). There is a plethora of online community resources that support the development and sharing of code. For instance, GitHub and CodePen are two websites that support collaborative coding. While GitHub hosts large-scale open-source code projects, CodePen, classified as a coding playground, is browser software that provides snippets of code that novice coders can download and modify (“CodePen,” n.d.; GitHub, n.d.; Sonmez, n.d). Not only increasing “potential for innovation,” this remixing practice challenges a commonly held view involving the importance of being able to independently create programs from scratch (Guzdial, 2015; Kafai, 2016, p. 27). This idea that privileges remixing code differs from common educational practices that measure students’ competency through individual assignments (Benda et al., 2012; Ericson et al., 2015; Guzdial, 2015; Kafai & Burke, 2017).

Facilitating learning coding through collaborative group activities is not an easy or trivial endeavour and is a current educational challenge that requires more thought (Kafai, 2016). One issue involves the remixing principle. Giving course credit for code students have borrowed from others' conflicts with traditional educational practices that say students must write all their own code. As Kafai (2016) suggests, teaching the collaborative practices of coding needs to extend beyond providing students with opportunities for group work. Ericson et al. (2015) provide an example of how technology might support computational participation. They describe an eBook material used in CS10K project, an online education program to prepare US teachers for delivering coding instruction. Its eBook contained an interactive environment that allowed their learners to form groups and complete activities collaboratively (Guzdial, 2015).

Computational Action: Coding for Social Impact

Computational action is the most recent potentially disruptive and transformative perspective that expands on Kafai's (2016) conception of computational participation (Tissenbaum et al., 2019). Under this perspective, the main purpose for learning coding is meaningful, social impact. The grassroots Vaccine Hunters Canada application that helped millions of Canadians receive timely Covid vaccine updates would be a recent example of using coding for meaningful social impact (Sheppard, 2022). There is also the example of the young women in Dharavi, a slum in Mumbai, who with little coding background, created mobile applications, such as the Women Fight Back app to protect their physical safety when travelling alone (Ranjen, 2015; The Logical Indian, 2016; Tissenbaum et al., 2019).

One way to look at Tissenbaum et al.'s (2019) conception of computational action is as a starting point for learning coding. This progressive perspective is reflective of the real-world situations in which teams to support a business objective learn what they need to know as they develop a solution. This view upholds the premise that inspiring projects and uses for coding, not building general-purpose, cognitive coding skills, best drive learning and should be the focal point of a coding curriculum. Adopting this perspective requires substantial shifts in educational thought. Enacting the principles of computational action requires both teachers and students to be more comfortable with the ambiguities and unknown factors in working on projects that do not have a set of solutions. Probably for many educators, it is unclear as to how to scaffold learning for such projects. Like computational participation, implementing computational action in a curricular program promises to provide a more rewarding learning experience but requires more insight to be effective in practice (Kafai, 2016; Tissenbaum et al., 2019).

Computational Thinking: Coding as Disciplined Thinking

As a traditional perspective, computational thinking involves the cognitive and dispositional activities required to achieve computational literacy. With regards to computational literacy, participation, and action, Wing's (2006) computational thinking perspective contains the most tangible specifics that help qualify the thinking that coding requires. Through the frame of computational thinking, coding requires a particular way of thinking that is based on computer science principles. One way that Wing (2006) summarizes computational thinking is as "a range of mental tools that reflect the breadth of the field of computer science" (p. 33). For example, computer science principles of abstraction, parallel processing, pattern recognition, decomposition, recursion, and automation comprise computational thinking (Jacob & Warschauer, 2018; Wing, 2006). While Wing (2006) theorizes that computational thinking can be extended for problem-solving and analysis outside of computing contexts, the value in this conception is that it articulates the specific ways of thinking that aid problem-solving through coding languages (Guzdial, 2015).

Computational thinking requires a creative focus or disposition (Wing, 2006). Those who have had more exposure to coding are more likely to appreciate the creative thinking that coding requires (Guzdial, 2015). As "a way that humans, not computers think," computational thinking relies on human imagination "to tackle problems we would not dare take on before the age of computing" (Wing, 2006, p. 35). Referenced in Table 1, Vaccine Hunters Canada, Women Fight Back and Paani (queue management mobile application for water collection in the Indian slum of Dharavi) are a few examples of applications that show people using coding as creative responses to problems in their daily lives ((Ranjen, 2015; Shephard, 2023; The Logical Indian, 2016).

Developing computational thinking, like the broader perspective of computational literacy, is a commonly cited reason for learning to read and write code (Byrd, 2020; diSessa, 2018; Floyd, 2022; Guzdial, 2015; Voogt et al., 2015). Connected to discussions of acquiring this way of thinking are the mental models of core coding concepts that serve as learning scaffolds. According to some computing education researchers, novice coders must develop mental models known as notional machines in order to think computationally (Du Boulay et al., 1981; Fincher et al., 2020; Guzdial, 2015; Sorva, 2013). An arcane-sounding term, a notional machine acts as a conduit both for the development of computational thinking and of computational literacy. Fincher et al. (2020) have created an educators' online space through GitHub that provides visual examples of notional machines to illustrate core coding concepts such as variables, arrays, recursion, and control structures. As scaffolds for building cognitive coding skills, notional machines reflect a commonly held view of coding primarily as a mental activity.

Matching Computational Perspectives With Models of Literacy

Street's autonomous and ideological models of literacy (outlined in Section 1) can be mapped to various approaches to developing computational literacy. While the autonomous view places the mastery of cognitive skills on the forefront, the ideological model centres on the social practices of developing literacy. Applying Street's (2006) autonomous and ideological models of literacy help illuminate distinctions between familiar and progressive notions around developing computational literacy. Figures 1 and 2 show how developing computational literacy and thinking differ under the autonomous and ideological perspectives. The autonomous approach (Figure 1) is characteristic of formal schooling in western culture (Carter, 2006; Street, 2006). Under a purely autonomous perspective to

achieving computational literacy, developing cognitive coding skills or computational thinking is the central learning vehicle. The progressive perspectives of computational participation and action are not part of the autonomous approach to achieving computational literacy. Implicit is the idea that people cannot begin to code for social impact until they achieve cognitive mastery of the core coding skills (Street, 2006). The autonomous model, reflective of Maintown's learning practices Heath (1982) documented, is a model that supports linear, predictable but limited learning experiences in classroom settings. While the autonomous approach is efficient, it is less likely to support transformative or even collaborative learning. Benda et al. (2012) document the difficulties encountered by seven adult learners enrolled in an online coding course. All of these students, many who struggled, saw their learning primarily as an individual pursuit. The lack of collaborative exchanges noted by Benda et al. (2012) is not only a problem because collaboration supports learning, but it is also an essential activity in real-world coding projects (Tissenbaum et al., 2019).

In contrast, the ideological view (Figure 2) of computational literacy promises greater rewards for its participants. A software product that students are invested in rather than a mark or a certificate is reflective of an authentic learning experience. Under an ideological perspective, engaging in a meaningful coding project without full mastery of cognitive coding skills is the first step to achieving computational literacy. Implicit in the ideological model is the idea that cognitive coding skills (i.e., computational thinking) are developed organically through an engagement in a meaningful purpose and through group collaborations. Isolated drill practices, canned problems and notional machines are learning tools and ad hoc scaffolds, not the focal point of the learning experience. At the centre of the ideological approach to computational

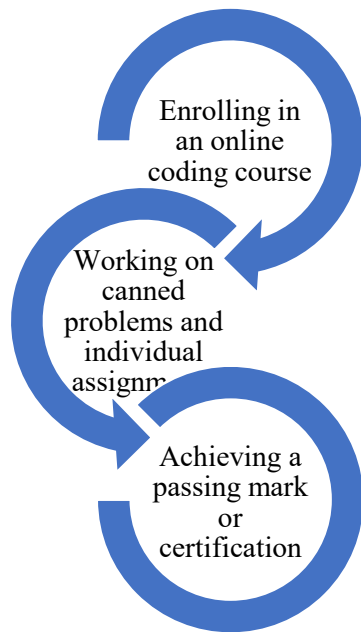
literacy is computational participation. This idea, at least, theoretically addresses a major flaw in the autonomous approach that neglects the synergies of group learning.

While the ideological model appears to be a more realistic model, it may not lend itself easily to standardized, modularized units of study that tend to please and placate educational administrators. Designing and delivering such a model would be quite labour-intensive for educators. An ideological curriculum would contain ambiguities and require instructors with highly developed pedagogical content knowledge. In addition, both Kafai (2016) and Tissenbaum et al. (2019) recognize that learning coding based on computational participation and action perspectives require major pedagogical shifts to inform the delivery of such a model. One idea to incrementally improve coding education in classroom environments is to adjust the focus from computational thinking activities and to include more activities reflective of computational participation (Kafai, 2016).

The figures below serve to acutely highlight key distinctions between autonomous and ideological models of literacy. Hybrid approaches to achieving computational literacy would achieve a good balance between student value and educational effort. For example, adding work and practice accessing open-source coding resources to a typical online certificate course would be adding an ideological principle to a skills-based coding course. There would be many possible ways to blend such approaches that would depend on the objectives and identities of the learning audience as well as the creativity of the instructor. There could certainly be a deeper examination on developing pragmatic instructional models that help different groups of students learn the required cognitive coding skills through social, collaborative methods (Kafai, 2016; Guzdial, 2015).

Figure 1

An Autonomous View of Computational Literacy

**Figure 2**

An Ideological View of Computational Literacy



Concluding Thoughts

Learning to read and write computer code would benefit multiple audiences of adult learners from social activists to those looking to acquire corporate employment. In addition to coding's range of audiences and purposes are many choices of languages and tooling. Given all these variables, there is a compelling argument for educators to recognize coding as a literacy practice. Such a view implies that there is a need to develop instructional coding practices contextualized specifically for the uses and needs of its audiences. Viewing coding as a literacy also serves to warn educators of the problematic outcomes of narrow and isolated skills-based assessments that tend to occur when a type of knowledge becomes recognized as a literacy. Street's (2006) ideological view of literacy helps counter such limiting approaches to coding instruction. Street's view upholds the notion that coding instruction should not be the same for everyone who has a purpose for coding. Guzdial (2015)'s example of the high-school computing teachers obtaining pedagogical content knowledge and Byrd's (2020) example of African Americans seeking entry into corporate America are both examples of Street's ideological model. Having different goals and identities, their instructional paths to learning coding are distinctly different. Byrd's example, in particular, warns that ideological implementations can be problematic when one group seeks literacy for the purpose of increasing their social and economic status (Street, 2009).

Finally, examining the computational ideals of literacy, thinking, participation and action help to further strengthen the conception of coding as a literacy practice. Through these perspectives comes the computational language, or terminology that helps articulate the importance of learning to read and write in a particular coding language. Achieving computational literacy might be seen as a general objective of learning to read and write code.

Possessing computational literacy involves having the knowledge of both the value and the limits of a coding language or technology (Guzdial, 2015). Engaging in computational thinking, participation and action are pathways to achieving computational literacy. While computational thinking helps define the cognitive and dispositional abilities required by coding, computational participation and action serve to highlight the social interactions and productive outcomes of being able to read and write code. Further exploring connections among literacy theory and these computational perspectives would provide ways to inform and improve current coding instructional practices that in turn could serve to increase broader participation in computing education.

CHAPTER THREE: RESEARCH METHODOLOGY

This chapter focuses on describing the methodology I used to examine computer coding as a literacy practice in adult learning communities. I engaged in action research to explore how I might apply insights both from my academic study and from my experience to a real-life teaching engagement. My study involves an analysis of my engagement teaching an online introductory web coding course to female newcomers to Canada. This chapter will discuss the purpose of the action research, the action research model I chose, and a brief description of how I have organized my project into phases. Also included in this chapter are the sections describing the participants and site, data collection, and data analysis. Lastly, this chapter will provide a discussion of the ethical considerations and limitations of this study. This study will focus on these questions:

- 1) Why do adult learners decide to develop their coding literacy?
- 2) What are the major obstacles for adult learners in developing their coding literacy?
- 3) Which tools and pedagogical approaches best sustain adult learners' motivation to develop coding literacy?

Purpose of the Action Research Study

Undertaking action research is a way that I, as an educational practitioner, can improve my ability to provide coding instruction (Clark et al., 2020; McNiff, 2013). I have chosen Macintyre's (2012) action research cycle framework to develop a more thorough understanding of how to support my adult students in learning to read and write computer code (Figure 3). In addition, action research is an appropriate vehicle to examine a "real live issue" that requires attention (McNiff, 2013, p. 90). The pressing issue that I encounter in my teaching practice is that

more than a few of my students who have shown an interest in learning coding give up on learning coding in my courses. Through my engagement teaching in COSTI's program, I noted steep drops in student participation. In one of the first classes I taught at COSTI, ten out of fourteen students stopped attending after three sessions. Often there were no explanations given for students' withdrawal. This was troublesome to me as a teacher but was also a curious situation as a researcher. Through my academic review of computing literature, I know that I am not the only teacher who has experienced this situation. Benda et al. (2012)'s study captured the perspectives of adult students enrolled in an online introductory coding course. In their survey of seven students, three of them either failed or dropped out of their coding course. In addition, computing education researcher, Guzdial (2015) cites low success rates in post-secondary coding courses from multiple US studies, especially among non-computer science students, those who do not identify as coders or programmers. While my concern about engaging students in computing is part of a broader problem, I wish to examine how I might change my thinking to develop tangible insights that I can use to improve my own teaching. McNiff (2013) clarifies that action research should focus on "taking action in your own thinking, so that your actions in the world work for other people's benefit, and therefore for your own" (p. 103). In this study, I aim to support a change in my thinking by examining how I am delivering course content and responding to students. Through this study, I hope to gain insights that will help me be more capable of sustaining a few more of my students' motivation in developing their coding literacy.

Action Research Cycle Implementation and Phases

I have chosen to utilize an action research model from Clark et al.'s (2020) interpretation of Macintyre (2012). I reviewed a few variations of action research cycles and chose this one as it best suited my study. Clark et al. (2020) have highlighted this model as a flexible, naturalistic

one that uses one's own reflections as a way into the research. At the start of this research when I was uncertain of my focus, I examined how my education, teaching experiences and other work experiences have influenced my interest and thinking on the teaching and learning of coding languages. This reflexive discussion included in Chapter 1 serves as phase 1 in my action research cycle.

Phase 2 involved gathering, researching, and consolidating scholarly insights that make up my literature review. These are the scholarly insights that I think will help me provide better coding instruction. In my literature review, I drew from both literacy theory and recent computing education research.

In phase 3, I have identified the teaching practices (Table 3) that I think will be effective for delivering coding instruction to the participants in my study. My experiences as a student and a teacher as well as my own research informed these teaching practices. These practices were used in the following phase.

Table 3

Teaching Practices to be Applied and Evaluated

What	How	Why
Give students a means to express their feelings and opinions about what they are learning and their objectives	Private note to instructor Note about oneself on Padlet shared with the class AnswerGarden to solicit anonymous responses to class topics End of class Survey	To understand the specific objectives of my students in supporting the development of their coding literacy (Guzdial,2017; Street, 1998; Vee, 2017)
Use browser software as the coding environment/coding editor	Codepen and JSFiddle, online coding sandboxes	To have everyone use the same tools and environment

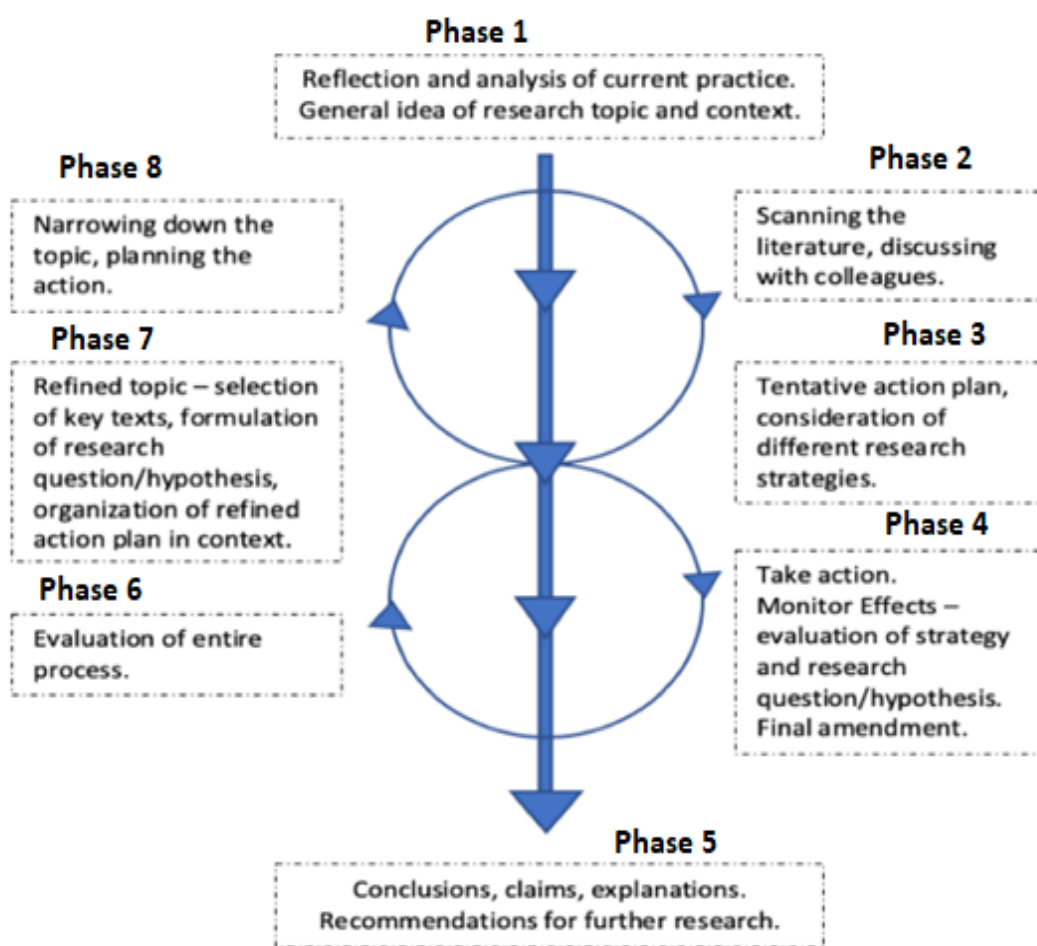
		To support the sharing of student work (Kafai, 2016)
Provide aids and materials to support my online class instruction	Powerpoint document Word documents with links to drills and activities for each class session	To provide multiple access points to the subject matter for my diverse group of students (Rao & Meo, 2016)
Modify existing software programs that are accessible in Codepen	“Forking” (copying) shared programs on Codepen	To demonstrate the remixing principle computational participation (Kafai, 2016)
Provide instruction that allows for student experimentation in class	Instructor short demonstrations followed by students’ hands-on practice	To support hands-on learning
Give students time to work with each other on coding activities	15 to 20 min sessions in Zoom’s break room	To demonstrate the collaboration aspect of computational participation (Kafai, 2016) To streamline instruction To allow for students to connect with each other and to make a space for themselves

Phase 4 involves the implementation of my tentative action plan. This implementation occurred in two teaching cycles starting in January 2023. This phase marks the initiation of gathering data from my students. In this phase, I also regularly took field notes after my teaching sessions. Phase 5 will focus on the analysis of the data collected in the previous phase and also will involve an examination of the results of this data analysis. The deliverable from phase 5 will be the findings and discussion provided in Chapter 4. Finally, the last phases 6 through 8 will include evaluation, refining and narrowing of my topic to be highlighted in Chapter 5. This study

teaching coding to female immigrants is an enactment of one cycle of Macintyre's action research model. Please refer to Figure 3 for an illustration of the action research cycle.

Figure 3

Macintyre (2012)'s Action research cycle



Source: Clark et al. (2020, p. 14).

Participants and Site

The participants in my study were female immigrants to Canada, who had enrolled in the COSTI organization's introductory coding course. As part of their immigrant services program, the COSTI organization provides English language instruction, technology training, and job

assistance to immigrants to help them establish their new lives in Canada. I was the instructor for their introductory coding course that was delivered through the online medium of Zoom from September 2022 until April 2023. Over the seven months that I taught online through this program, I met female students from these countries: Ukraine, India, Peru, Brazil, Afghanistan, Venezuela, Colombia, Iraq, Iran, Jamaica, China, Belarus, Japan, and others. In other words, there was no dominant national identity in this learning environment. Some of the students I met had only lived in Canada a short time, having had their lives in their native countries recently uprooted. Some of them were accomplished women holding university degrees; a few of them even held degrees in computer science from their countries of origin. I also noted diverse educational and professional backgrounds. Among my students were a doctor, a lawyer, an accountant, a police officer, a few early childhood educators, and a civil engineer. Some of my students were mothers of young children wanting entry and access to the professional workforce. There were also some older women who had grandchildren.

Some struggles and challenges of these students included learning the English language as well as finding suitable housing and employment. Some students had unstable internet connections which on occasion disrupted their participation. Another aspect that made teaching this course more difficult was the fact that students used their own devices such as Mac books, Chrome books and Windows-based computers. Standard software developer code editors used in previous cycles such as Sublime were operating system-dependent and would not work on all device types. To make device and software differences less of a distraction, I made the decision to have students use self-contained online coding editors, such as Codepen, that would work on all devices. This was part of my tentative action plan from phase 3 of the action research cycle.

This course involved 5 hours of online class time weekly over the course of five weeks. Each lesson was 2.5 hours, and there were 10 lessons in total. The coding languages taught were HTML, a markup language, and CSS, a styling language. In addition, there was brief exposure to JavaScript, an interpreted web programming language. Using both HTML and CSS languages, students would add content and information and practice creating visually appealing web pages. For each session, I prepared visuals through PowerPoint, a lab document, and often included short YouTube videos. For the presentation of short videos, I developed the habit of turning on the captions and slowing the speed of the videos. Each session included short demonstrations and a chance for students to practise coding. I invited students to share their screens and demonstrate what they had accomplished and more importantly, what they were missing. While the sharing of screens was something not everyone felt comfortable doing, it was a practice that I regularly encouraged so that they received personalized guidance and direction in these sessions. Their sharing of work also helped me understand how students were progressing so that I could adjust the instruction appropriately.

According to COSTI's program manager, the intent of this course was to give the participants exposure to coding that would provide them with some experience to make an informed decision regarding pursuing more computing education. At the end of the 5 weeks, students received a certificate if they attended 80% of the classes and created a webpage as their summative assessment. Requirements of their assessment included using code to add multimedia content (memes, images, videos) to their web page as well as articulating what they have learned about the web coding languages. As final feedback, I would write personalized notes highlighting what I noted from their web page as well as from their participation in the course. I also included suggestions for further study in my final feedback. Students who completed the

final assessment received a certificate of completion. A few students expressed disillusionment and disappointment that the certificate did not qualify them for a corporate job in Canada.

Data Collection

In this study, I incorporated multiple sources of data that include both data I created and student data. Gathering data from myself as well as from my students is an example of data triangulation that helps increase the integrity of this study (Wilson, 2014). My data include my field notes, a report I prepared to the funders of the coding course and learning materials. I also took the Teaching Perspectives Inventory (TPI) survey to better understand how I taught the Introduction to Coding course. Participants' or students' data include their responses to an online survey questionnaire and responses to personalized interviews. While the study relies extensively on documentary data-gathering methods such as field notes, it also incorporates a live method through student interviews (McNiff, 2013). Incorporating different data-gathering methods such as documentary and live methods is an example of triangulation, methodological triangulation, that also helps to increase the depth of the study (Wilson, 2014). In addition, the use of data generated from my field notes, my TPI survey results, and student interviews reflect my intention to gather and analyze personalized data for this action research study.

Field Notes

Throughout this study, I relied extensively on field notes. Field notes are “jottings” of “researchers’ private, personal thoughts, ideas, and queries” taken on-site (Phillippi & Lauderdale, 2018, p. 381). Described as “an essential component of rigorous qualitative research,” field notes were the records of my observations of students in the online sessions (Phillippi & Lauderdale, 2018, p. 381). Making field notes helped me obtain “valuable contextual data” that were critical in supporting deeper analysis (Phillippi & Lauderdale, 2018, p. 381). With my “field” site being the online Zoom classroom, I recorded my field notes shortly

after a session ended which was either in the evening around 8 p.m. or in the afternoon after a 5-hour session. Often, I was tired at the time of recording my notes but saw the value in the academic advice in taking these notes close to the time I was interacting with students (McNiff, 2013). In these notes, I recorded details related to the students' characteristics, my feelings about how the session went and responses from students that surprised me. Some field notes were more reflective than others. The notes also track my worries and feelings about drops in student participation. My notes included the date and the session number out of the 10 sessions. By recording the session number, I could easily identify the lesson plan I used in that session. There were four sets of field notes: two from the 2 classes that ran January, and the another 2 sets of notes from the March classes. The use of field notes from these 4 separate classes is another example of the data triangulation used in this study.

Report to COSTI's Course Funders

Reports can also serve as valuable documentary data-gathering methods to support action research studies (Atkins & Wallace, 2016; McNiff, 2013). While I was teaching my last classes, I created a report for the funders of the coding course. My comments addressed 1) challenges with the course delivery, 2) effective and ineffective teaching practises, 3) a summary of best practices, and 4) opportunities for improvement and expansion. This two-page report serves as a record of my thinking while I was actively teaching my last two groups in March 2023 and before I started reviewing students' interview content. By noting the timing of this report, I was able to see changes in my thinking as they developed through this action research cycle (McNiff, 2013).

Teaching Perspectives Inventory (TPI) Survey

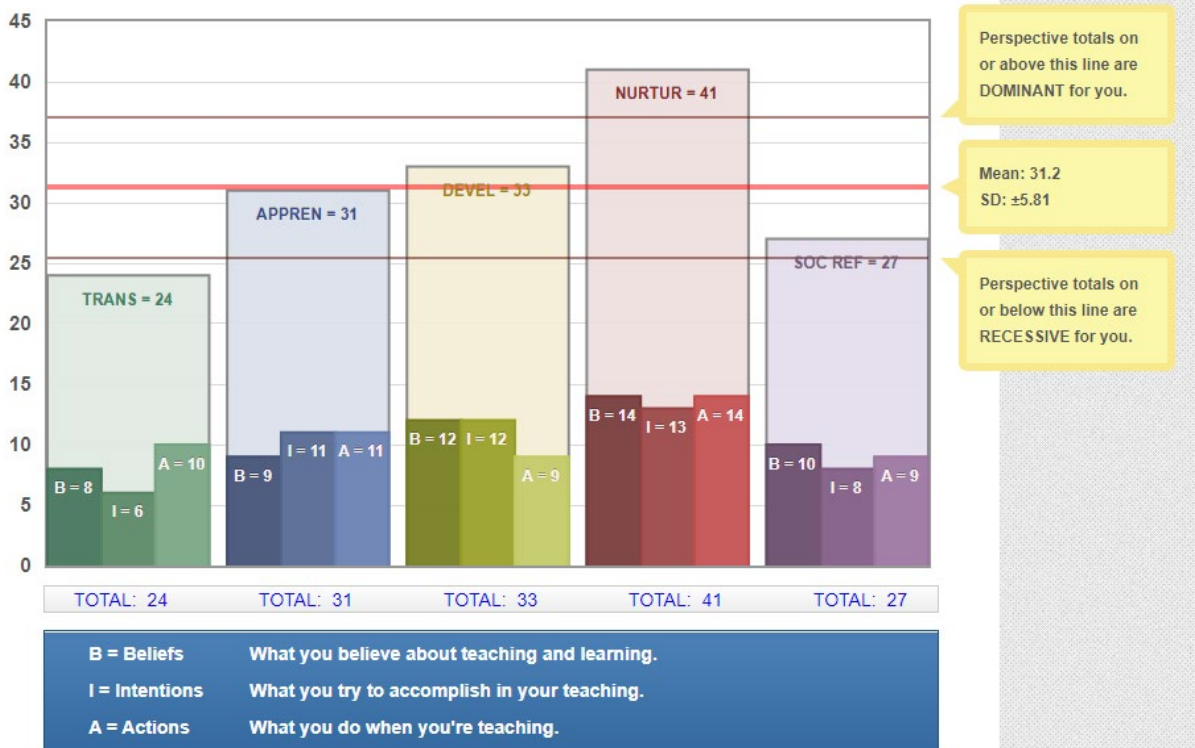
As a way to obtain an external perspective on how I approached my coding instruction in this study, I took the TPI online survey and made notes of my results to examine further. This well-known survey data helps teachers identify their own teaching beliefs, intentions, and actions (Collins & Pratt, 2011). This is another documentary data method used to check my own perceptions as to how I was relating to my students in the study (McNiff, 2013). In addition, Guzdial (2015) identifies taking this survey as being helpful for computer science teachers in

Figure 4

TPI Survey Results

TPI Profile Sheet

Thank you for taking the TPI. Your results are represented on the graph below. For information on how to interpret your results, please see the [Interpretation page](#)



Note: I used my TPI survey results to develop a stronger awareness of my teaching.

improving their teaching. Reflecting on the results of TPI survey can help educators better understand their own assumptions about teaching and also can help increase educators' awareness of other approaches to teaching. The five teaching perspectives are transmission, developmental, apprenticeship, nurturing and social reform (Collins & Pratt, 2011). Based on my experience teaching the Introduction to Coding Course, the survey identified my dominant perspective as the nurturing perspective with the transmission perspective standing out as being recessive (Figure 4). While I was not aware that I would have ranked so highly on the nurturing perspective, I was not surprised that I scored the lowest on the transmission perspective.

Online Survey Questionnaire

In the January and March teaching cycles, I conducted an online survey questionnaire using Microsoft Forms (Appendix A). I conducted the survey at the end of the class session during the second to last week of the course. A benefit of using this method is that it was relatively simple to administer as a class activity. The survey also was a good way that I could obtain data from multiple students (Cohen et al., 2000). Unlike the time-intensive interviews, conducting these questionnaires could be completed within a short time frame of fifteen minutes. McNiff (2013) identifies questionnaires as a documentary data method and highlights questionnaires as a way to obtain “an idea of trends” (p. 109). The online survey questionnaire included multiple-choice questions and several opportunities to provide short-answer comments. To support the study's reliability through data triangulation, I invited former students from the November and October sessions to complete the survey in early March 2023.

The purpose of these questions was to develop a general sense of their reasons for taking the course as well as their opinions on how they learned coding through the different methods provided in the course. Questions were structured around these ideas: 1) how coding skills and

knowledge could help them in their lives 2) what they considered to be effective learning activities 3) their challenges in learning to read and write code and 4) their desire to continue learning coding. In designing the survey, I aimed to examine how students' reasons for learning coding matched with the reasons I cited from my literature review. Another important survey objective was to record whether these students wished to continue their learning of coding. This question reflects an important assumption of this study: namely, that a desire to continue to learn coding is a gauge of learning success (Guzdial, 2015). The final question allowed for comments that required qualitative analysis. Twenty-four survey responses were collected; the majority of responses (67%) were students from the January and March courses who took the survey during class time. The other 33% of responses came from students after they had completed the course months earlier. While less representative, this is significant because these were responses from students who had more time to reflect and think about what they had gained from the course. This was a detail I paid attention to in my analysis.

Semi-structured Student Interviews

Like the data gathered from the field notes, semi-structured student interviews were used to elicit highly personalized data (Appendix B). Adams (2015) characterize the semi-structured interview as a conversation and describe this technique as “a blend of closed- and open-ended questions, often accompanied by follow-up or how questions” (p. 493). Identified as a live data method because it captures the real-time responses of the interviewee, this data also serves to supplement the results from the online survey questionnaire with richer qualitative content (McNiff, 2013). In the spring of 2023, I conducted interviews with a few of these students to more fully bring student voice into the research (Bron & Veugelers, 2014). I invited all students who had been active in the program a chance to interview with me. I had six express

some interest and engaged four of these students. This was a daunting request for most of these students. Three out of the six potential volunteers expressed some apprehension as to whether their English skills would be at the required level for participation. For one particular student who doubted her ability to engage in a conversation with me, I suggested that she could provide her responses in written form, and this student was quite receptive to this option. Two of the women provided written responses to a set of personalized questions I prepared. Unlike the students who spoke with me, they would have had more time to reflect on their responses and would have had opportunities to receive assistance. The other two women participated in an online 1-hour recorded conversation with me. Like the questionnaires, the questions focused on reasons for learning to code, experiences in the course and their desire to continue learning to code. Unlike the questionnaires, these questions contained more personalized questions relating to how well prepared they thought they were at the start of the course, their own professional aspirations, and particular observations I made of them based on their participation in the course. In the findings and discussion provided in Chapter 4, I used pseudonyms to refer to these students for the purpose of protecting their privacy.

Data Analysis

Given that this is my first study performing extensive data analysis, I followed straightforward research advice provided by Atkins and Wallace (2012) and Rosala (2022). While Rosala (2022) provided an accessible guide on performing a thematic analysis of data, Atkins and Wallace (2012) presented a broad set of steps for data analysis that progresses from reading data to presenting data. I began by reviewing my datasets thoroughly to gain a solid sense of the data I collected; this is what Atkins and Wallace (2012) highlight as immersing yourself in the data. To start with my data analysis, I focused in particular on the richer, larger

textual data sets that included field notes, the open-ended responses of the online questionnaire survey and the written interview data. Using Rosala's (2022) guide, I applied open codes, or phrases, to label segments of the textual data. I then created an Excel worksheet for each dataset and recorded each code along with its corresponding data. From the open codes, I assigned a category to begin to group and organize my data. Table 4 shows a few of my entries from my field notes.

Table 4

Coded Data Examples

Category	Open Code	Text
Student characteristics	Students hesitant and cautious about sharing	From the Padlet entry, they seem hesitant and cautious to share details about themselves
Instructor discomfort	Teacher not feeling comfortable	I feel a bit incompetent in pronouncing their names with the right accent
Discovery of a teaching tool	A new way to use online tools for teaching	Realized I could use a Zoom pen which is what I started using
Indications of learning and engagement	Student-led class	We did not get to the planned activity due to individual questions they had

Note: This is a sample of how I organized my field notes content. Text refers to the actual field note content. I then assigned an open code and then a broader category.

After completing this process for my textual data, I looked for patterns and made notes of themes I saw. Emerging themes included 1) my own uncertainty and doubts about my teaching, 2) students' changing attitudes, behaviours and responses in the online classroom, and 3) the need to alter instructional plan to better respond to students.

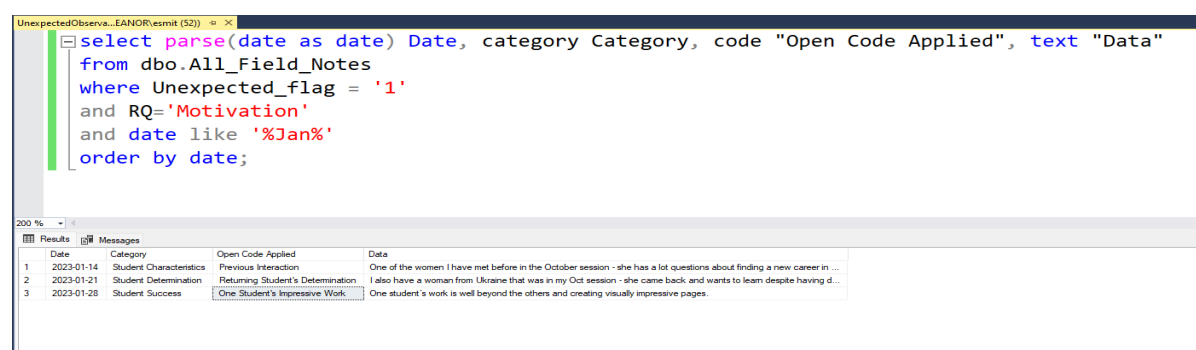
For the audio content of the two semi-structured interviews, I followed a similar process but included more steps as I deemed it important to listen carefully to the audio recordings to make note of what students were communicating through their tone and verbal expressions. As I listened the first time, I recorded notes and summary points. Some interesting notes include those related to students' conceptions of coding. They include "coding as providing more than a survival job," "coding as my pathway into Canadian life", and "coding as a new sphere." As a tool to support my second listening, I had transcriptions created using Cockatoo's online software. As I listened, I highlighted important passages and recorded them in spreadsheets. Next, I followed a similar process that I used for the textual dataset: I applied codes and then assigned the observation to a broader category.

For the supplementary documentary data items such as the management report I wrote and teaching materials, I followed the same process by applying code and categories to text segments that I considered significant. As I progressed through this analysis, I found this process messy and overwhelming to organize. It occurred to me that my analysis would benefit from standard organization that database tools and technology could provide. As part of my own coding literacy that I had developed from my workplace experience, these tools include the Sql Server database, its data management software and the sql query language. In establishing a standard record format for all my data sets, each record had these attributes: date of observation, category, code, text. For each record, I added two additional attributes or tags: 1) which research question the record, or the observation could best support and 2) whether or not the observation was surprising or something I did not expect. I loaded the data from each excel worksheet into a personal SQL Server database table. There was a separate table with the standard format for each data set. Loading these data sets into Sql Server tables gave me the ability to use the sql query

language to perform grouping and filters on the data to support my inquiries while writing about my findings (Figure 5). It also gave me the ability to consolidate datasets such as the field notes tables and the interviews tables. Figure 5 shows how my organization using database software supported one type of inquiry. This query retrieved field note records from January that I had tagged as unexpected and as supporting my research question related to sustaining students' motivation.

Figure 5

Querying Coded Data Observations Using Sql Server Management Studio Software



The screenshot shows the SQL Server Management Studio interface. The top pane displays a SQL query: `select parse(date as date) Date, category Category, code "Open Code Applied", text "Data" from dbo.All_Field_Notes where Unexpected_flag = '1' and RQ='Motivation' and date like '%Jan%' order by date;`. The bottom pane shows the results of the query in a table with four columns: Date, Category, Open Code Applied, and Data. The results are as follows:

	Date	Category	Open Code Applied	Data
1	2023-01-14	Student Characteristics	Previous Interaction	One of the women I have met before in the October session - she has a lot questions about finding a new career in ...
2	2023-01-21	Student Determination	Returning Student's Determination	I also have a woman from Ukraine that was in my Oct session - she came back and wants to learn despite having d...
3	2023-01-28	Student Success	One Student's Impressive Work	One student's work is well beyond the others and creating visually impressive pages.

Note: Here is how I was able to repurpose my workplace coding literacy (the sql language and query software) for academic usage.

Ethical Considerations

I pursued and received an ethics clearance through Brock University, Office of Research Ethics (file 22-134 – COLLIER, Appendix F). I also obtained approval through COSTI, the organization that runs the introductory coding course. I elicited students' responses through a Microsoft Forms survey (Appendix A) and also through one-on-one oral and written interviews (Appendix B). A risk to my group of students involved the fact that I was the authority who evaluated their work; because of my role, they would likely feel obligated to participate and provide pleasing comments. To help address this issue, the survey results were sent to Dr. Collier and were released to me after I completed the evaluation of their work. I spoke

to this important detail in class; this information was also presented in the informed consent section of the online survey (Appendix C).

For the interview activity, I sent an email invitation to all students from the October to January sessions who had completed the program (Appendix E). In this email, I described the research project's benefits as well as emphasizing that their participation was optional. For the March sessions, I informed these students in class of an opportunity to participate in an interview with me. Students who participated in the interview activity all received ten-dollar gift cards as compensation. From those who expressed interest, I provided specific details regarding voluntary participation, benefits and risks, confidentiality, compensation, and the right to withdraw through the Informed Consent document (Appendix D).

I carefully reviewed the language used in the survey, interview prompts, consent forms, and invitation scripts with Joeseeph Padro, manager of the COSTI organization. Padro made helpful suggestions to me regarding modifying my wording to be more accessible to my participants, many of whom did not have a full command of the English language. One suggestion employed on the survey's informed consent area was to use questions, such as "Do I have to participate?" as headings to organize the important details related to participation, confidentiality, risks and benefits (Appendix C). I also took extra care in communicating these aspects of the research process by providing them these important details in both oral and written form. I also informed the four women who provided me with personalized data how I used their information and wording in my findings and discussion chapter.

Limitations of the Study

There are several limitations of my study to be acknowledged. My action research study seeks a more thorough understanding of how teachers might address obstacles students encounter

in developing their coding literacy. While my study does incorporate students' thoughts and opinions, my study captures those thoughts and opinions of my students who did not withdraw from the course. My study incorporates data provided by committed students rather than those who decided to withdraw. A significant number of students who signed up for the introductory coding course were not successful in fulfilling its requirements of 80% attendance and submission of a web page. In addition, it is quite possible that some important thoughts from my students, all of whose first language is not English, were not fully captured because they expressed themselves in a language less familiar to them.

Another limitation of this action research study is that it centres on one narrow application of coding literacy. In this introductory coding course, web development skills through the HTML and CSS languages were the focus. HTML is a markup language used to describe the content of a web page, and CSS is a styling language used to add visual properties to a web page. Through our practice of these two coding languages, there were limited opportunities for the development of decision-making and iterative logic, important aspects of computational thinking. JavaScript is another major web language that could be used to develop these aspects of computational thinking. In these five-week sessions, there were brief mentions of JavaScript and its uses.

Finally, my lack of practical experience in performing extensive qualitative data analysis is a significant limitation of this study. I was learning qualitative data analysis among other important topics, such as action research, literacy theory and computational perspectives for my study. In particular, I learned that conducting oral semi-structured interview requires a nuanced awareness that develops through experience and reflection. Peredaryenko and Krauss (2013) point out that “sensitivity, responsiveness and flexibility” are qualities that researchers require

when interviewing subjects (p. 1). Looking back after having listened to the audio content, I was less able to demonstrate these qualities in my interviews as I was feeling the urge to be in control and anxious to obtain valuable research data. As a result, I spoke too much which could have possibly stifled the interview subject. Unexpected, rich data is more likely to emerge when researchers are able to create a comfortable, open space for their interview subjects to contribute (Peredaryenko & Krauss, 2013). While this action research study has given me a deeper understanding of using interviewing as a qualitative data method, I am still in the early stages of developing these multiple sets of skills. Because I was learning the subtleties of conducting interviews in addition to other skills, my findings and reflections from the interviews are likely not as nuanced and textured as they could have been.

CHAPTER FOUR: PRESENTATION OF THE FINDINGS AND DISCUSSION

This study examined computer coding as a literacy practice in adult learning communities to identify ways to customize coding instruction to make it more effective. The students of this study were female immigrants to Canada, who had enrolled in the COSTI organization's introductory coding course. Through an action research framework, I sought to examine my specific situation involving teaching web coding to a diverse group of adult women. Through an analysis of my field notes, student interviews, and other documentary artifacts, I aimed to explore how these students saw learning coding as a way to help them progress in building new lives for themselves in Canada. In addition, I sought an understanding as to how to best support their development of coding literacy. In this discussion, I will use the term coding literacy, a broad term that refers to the ability to read and write computer code to fulfill a wide range of purposes (Vee, 2017). An important premise of coding literacy is that instruction would vary according to participants' objectives (Guzdial, 2015; Street, 2006; Vee, 2017). The students in the study had their own goals for learning coding that general-purpose or standard academic coding instruction would not necessarily address. This discussion focuses on these research questions:

1. Why do adult learners decide to develop their coding literacy?
2. What are the major obstacles for adult learners in developing their coding literacy?
3. Which tools and pedagogical approaches best sustain adult learners' motivation to develop coding literacy?

This discussion of findings includes 1) students' perceptions of the benefits of being able to read and write computer code, 2) language and communication barriers as distinct aspects of this

course experience, and lastly 3) a review of the effective teaching tools and practices used in the introductory coding course. An examination of these three areas of insight might provide a more complete picture of coding literacy for this study's participants.

Students' Varying Perceptions of the Benefits of Learning Coding

This first section will discuss students' reasons for their enrollment in COSTI's introductory coding course. It will also discuss a few reasons that students gave for their continued engagement in the course. While students' reasons for their initial participation reflected their personal beliefs on the value of coding, students' reasons for their continued engagement highlighted some unanticipated outcomes from their experience in the course. These reasons not only reflect their views on the value of learning coding but also show several real-world examples of some uses for coding.

Participants stated that developing employment skills was the main driver for their participation in this course. From the online questionnaire survey that students completed in the 4th week of the course, the top reasons given for learning coding are in the category of professional pursuits. While 48% cited that their main reason for taking the course was to increase their job opportunities, 44% indicated that increasing their personal and professional productivity was their top reason for participation. While the top-ranking reason does focus exclusively on employment, the second-ranking reason regarding increasing personal and professional productivity is more general.

These survey responses matched my field observations that employment concerns heavily influenced their decisions to participate in this introductory coding course. In my field notes, I recorded occurrences of student-initiated discussions regarding ways to approach obtaining a job in Canada. When students raised job-related questions, I drew from my own experiences in

finding corporate jobs. For example, I advised them to get familiar with job and career websites, such as Indeed to search for jobs and LinkedIn to showcase their professional profile to potential employers. With student-initiated discussions being generally infrequent, I saw that topics around seeking employment were an effective way to engage these students in class conversations. In one of our January sessions, a few students voluntarily stayed online after class to chat with me about their career direction.

The theme of learning coding for employment and socio-economic advancement in Canada also appeared prominently in the students' interview data. Four of COSTI's students agreed to participate in a semi-structured interview activity. These responses provided more textured details regarding their hopes for employment and also included other benefits that coding literacy could provide them. While the promise of more favourable type of employment initially attracted all of them to COSTI's program, two of the students revealed that they recognized other benefits beyond obtaining employment that sustained their commitment to learn coding.

Favourable Employment Seen as a Main Objective of Developing Coding Literacy

There are some intriguing variations in students' perceptions of how coding might help them make a living in Canada. For example, *Evita*¹ saw learning coding as an opportunity for flexible, self-employment to cope with rapidly rising costs of living. Evita commented that learning coding would help her obtain "freelance opportunities" and generate "a side income which is really needed with how the economy is now." In the classes, there were several mothers with young children who saw coding knowledge as allowing flexible work arrangements. For them, coding had the potential to help them obtain jobs that they could perform at home while

¹ Pseudonyms are used for all participants in this study.

staying close to their children. These students including Evita recognized coding literacy as helping them adapt more favourably to life's demands and circumstances (Scribner, 1985).

Another student, Mahvash, saw learning coding as a way to move beyond the need to take precarious employment. Mahvash with her family had fled the Taliban takeover in 2021 and had lived temporarily in the Middle East, Europe, and the US before settling in Canada. After she and her spouse were settled in housing in Canada, her next major life challenge involved finding employment that would match her level of education and one-year work experience as a civil engineer. Mahvash said she was able to obtain “a survival job,” a contemporary term that refers to low skill work that people accept while working towards obtaining employment more conducive to their talents. Taking a survival job, a common experience for many immigrants, might be described as an anti-climatic experience for highly educated women like Mahvash, who managed to safely escape a harrowing situation in her native country (Keung, 2023; Kusumajuda, 2022). Because Mahvash regarded web software development as a highly esteemed occupation in North America, learning web coding was her approach to her problem of not being able to find suitable employment. Throughout our conversation, Mahvash iterated her belief that seeking more education in coding and technology will improve her job prospects. At the time of the interview, she had secured an online part-time teaching position as a math teacher for Grade 7 and 8 students. This position is one that does not quite classify as a survival job in that it makes use of her education but is a part-time position that would not quite be considered the gainful employment she hoped to obtain. More importantly, Evita's and Mahvash's views that coding will help increase their socio-economic prospects are their beliefs that have yet to be realized at the time of writing this paper. Their opinions reflect a common view that coding literacy will lead to more favourable employment (diSessa, 2018; Guzdial, 2015; Vee, 2017). The widely

held notion of coding knowledge and skills leading to a good job, particularly in a specialized IT field, is one that diSessa has questioned. Critical of the jobs' argument for learning coding, diSessa (2018) mentions that "the vocational value of programming, per se, might be overblown and more symbolic than functional" (p. 23). Statistics Canada figures from May 2021 may not show an overwhelmingly positive picture for immigrants seeking entry into coding and technology jobs, but these figures do suggest that there are reasons for Mahvash and Evita to be hopeful. While information and communications technology (ICT) sector jobs make up a small and growing percentage of total employment at 3.9% in Canada, 40% of those employed in the ICT sector were immigrants (Statistics Canada, 2022). In addition, the coding languages they were learning, such as HTML and CSS, are foundational coding languages required for web page development. Given that the vast majority of businesses and organizations have an online presence, learning these languages (as opposed to more obscure or specific languages such as Assembly or Fortran) would at least help increase their marketability to a wide range of businesses and industries.

Beyond the Employment Argument for Coding Literacy

More to the point in diSessa's (2018) criticism of the jobs' argument for learning coding is that there are other important reasons to develop coding literacy. For instance, learning coding can transform the way we think and learn (diSessa, 2018, Guzdial, 2015; Vee, 2017). Accounts from two other students, Mira and Michi, include other reasons beyond employment for learning coding. More importantly, their accounts reflect actual learning outcomes that include greater self-confidence, self-awareness, and stronger English language skills. These next student examples help show coding literacy as a way to develop other types of knowledge including increased communication skills and self-knowledge

Self-efficacy as One Woman's Realized Outcome of Developing Coding Literacy

As a model student, Mira increased her self-awareness of her abilities and talents through her participation in COSTI's coding course. Her interview with me occurred six months after she had completed the course. As the instructor who introduced her to web coding languages, I was impressed by her use of advanced software developer terminology such as Flexbox and Grid. After taking the course, Mira taught herself how to use highly specialized coding tools such as Visual Studio Code and advanced web layout techniques. What she was learning on her own were standard topics that would be covered in a semester-length Canadian college web coding course. At the time of starting the course, Mira lived in Canada for two years, was finding learning conversational English difficult, and had been feeling "a little lost." She had not sought coding skills or knowledge previously before moving to Canada from Belarus. A psychology major in university, she had the perception that coding was an activity limited to people who excelled in math. She made a comment that, "you need to be pretty smart for this stuff" that might suggest she had some doubts about her own capabilities to acquire coding knowledge. She decided to take the coding course at COSTI because it was a free course, and she had the time to devote to the course; it was an accessible opportunity to learn new, current knowledge that might lead to a new occupation for herself. While the jobs argument may have influenced her decision to join the coding class, she recognized several other benefits that kept her motivated to learn more about web coding.

Being able to write code that resulted in the creation of a web page was something she described as "really inspiring." Like other COSTI students in the introductory coding course, Mira was also concurrently enrolled in COSTI's English classes to develop her language skills. In the coding class, Mira created a personal portfolio page that required her to list her skills,

education, and experiences. This assignment that required her to communicate details about herself gave her practice not only in coding but in communicating in the English language. Reinforcing the language skills she was developing in her English course, the assignment also required Mira to use code that included multimodal content such as photos, pictures, graphical effects, and sounds to support the text content on the web page. This example connects with another theoretical argument for learning coding that involves access to new ways of communication and expression (Guzdial, 2015; Vee, 2017). Reflecting on the value of the course, she described the learning experience as “a good opportunity for women to start something new in IT sphere.” Her opinion reflects the social progress/broadening participation argument for learning coding (Guzdial, 2015; Vee, 2017). COSTI’s introductory coding course was specifically intended for female newcomers to help them learn to read and write computer code. Through her participation in the course, Mira not only developed web coding skills and her English communication skills but also her self-efficacy.

English Language Skills as an Unanticipated Outcome of Developing Coding Literacy

Another student, Michi, adjusted her expectations as she progressed through the introductory code course and realized an outcome that she did not fully anticipate. Her initial motivation for joining involved employment, but she did become disillusioned with this objective. She said :

My thought was that I may have a chance to get a job with coding skills.
However, I found out that it is difficult to get a job only learning introductions
and honestly coding doesn't suit me.

Michi revealed to me that she was on the verge on withdrawing from the course but persevered because she recognized the course content along with the methods I used would help her improve her English skills. English like coding was knowledge that would help her establish

herself in Canada. In her reflection on the course experience, she felt strongly that her English had improved “by learning coding including the ways you taught us.” Michi clarified that the coding practice along with class discussions helped her improve her English conversational skills. Her engagement in the course also helped increase her vocabulary. In our class, we experimented with adding different fonts and colours to our web pages by using the CSS styling language. Through writing CSS code, Michi learned the English names for fonts and colours.

Michi’s experience illustrates that another possible reason for learning web coding is to learn English. It could be said that Michi coded to better learn how to communicate in the English language (Guzdial, 2015). This idea of coding as a pathway to develop other types of knowledge is present in academic literature. For instance, scholars have conceptualized coding as a way to understand advanced mathematical concepts (diSessa, 2018; Guzdial, 2015). There are examples of how coding languages such as Logo, Boxer, and Bootstrap support the introduction of mathematical concepts to children (diSessa, 2018; Guzdial, 2015). This is the essence of computational literacy, a term referring to the use of coding and computing to support transformative learning (diSessa, 2018). Michi’s account helps support the idea that coding to learn other types of knowledge is not limited to math and science disciplines. While Stevens and Verschoor (2017) remark on the potential of the HTML language, a markup language, to help foreign adult students learn English, the idea of coding to learn English is more obscure (Portnoff, 2018). As a markup language, HTML is used to organize content on a web page (MDN, n.d.). There are established standards for writing semantic HTML that support search optimization and accessibility (Juviler, 2022; W3C, n.d.). Learning HTML and its associated best practices are particularly helpful for these students because it gives them a powerful option for creative online expression.

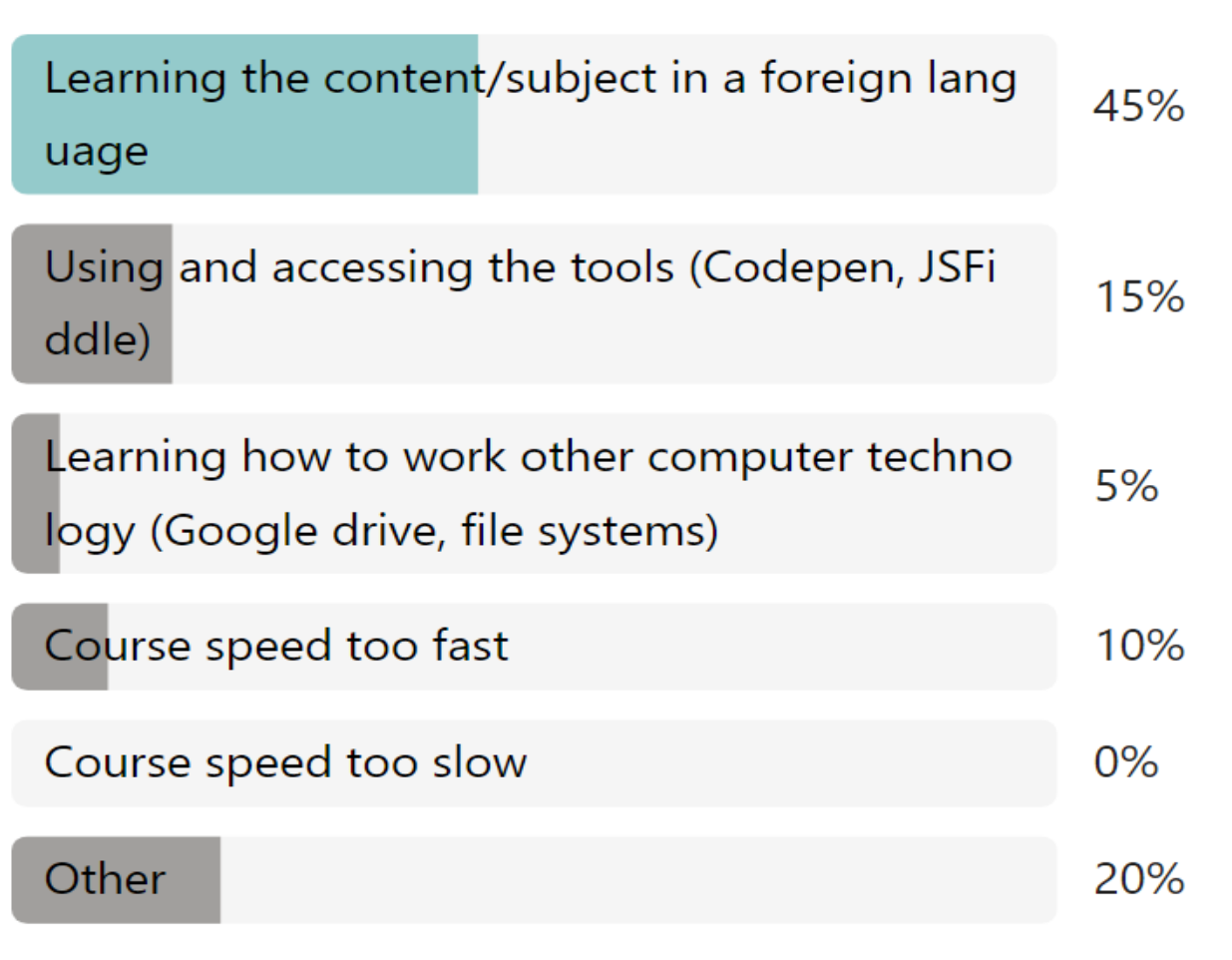
Language and Communication Barrier as a Distinct Aspect of the Learning Environment

While at least one student continued in COSTI's introductory course because she recognized a valuable opportunity to improve her English skills, there were others who withdrew from the course because learning coding in their non-native language was too much of a challenge. In this section, I will describe the language and communication challenge that adversely affected student engagement. This section will also highlight a few teaching adjustments made as well as this barrier's impact on the productivity of group work.

Data from the online survey questionnaire support the interpretation that the English-language delivery was a significant obstacle for them. In the online survey questionnaire, students first selected their greatest difficulty. From the online survey questionnaire, 45% responded that learning the content in a foreign language was their biggest difficulty in the course. This was the highest-ranking response with the next one accounting for only 15% of students' replies (Figure 6). In the following question, they had to select any other difficulties that they might have experienced. Another 25% indicated that receiving instruction in their non-native language may have not been their greatest difficulty but was an aspect of the course that made their learning of coding harder (Figure 7). A few students did provide duplicate responses to these sets of questions, and this might indicate that these students did not fully understand the English instructions on the survey. In addition, for both questions, no one selected 'course speed too slow' as a learning issue, but a few did indicate the opposite that the course speed was too fast. The responses involving the pace are likely related to the English-language delivery.

Figure 6

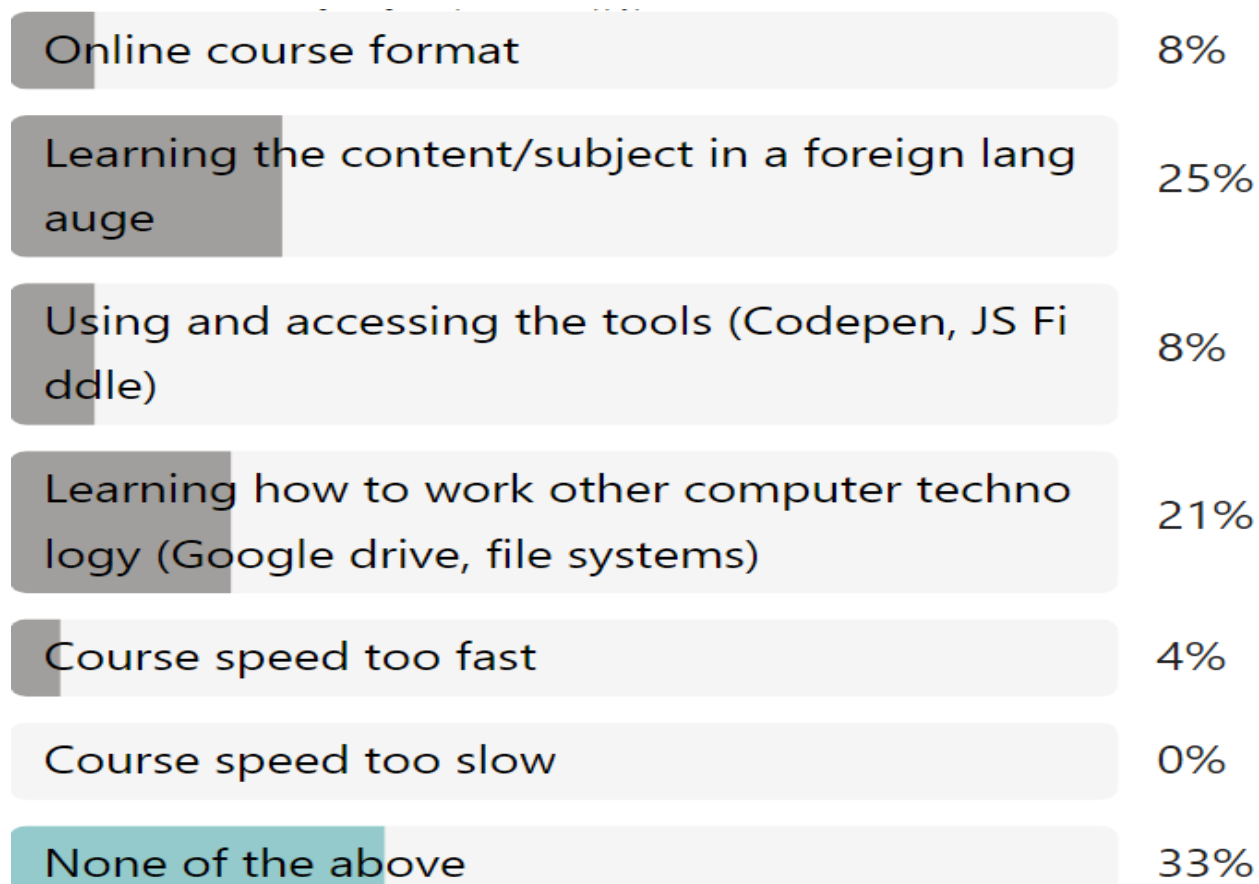
Question 9 from the Online Survey Questionnaire



Note. In this question, I asked students to select their “biggest difficulty to learning coding in this course.” This question allowed for only one selection.

Figure 7

Question 11 from the Online Survey Questionnaire



Note. In this question after asking students to identify their greatest difficulty, I asked students to identify any other difficulties they experienced in the course. There were a few instances in which students selected the same answer for both this question and question #9. This may indicate students' misunderstandings of the questions' wording.

In my personal story, I recalled that my first experiences learning code in my native language were challenging for me and tested my patience immensely. Many of these students quite likely felt even more frustrated than I did because they were dealing with the added complexity of learning coding in their non-native language. Michi recalled from her memory of the first day of class:

I can't forget what I felt on the first day of a course. I was thinking, "Today could be the first day and last day." Because I couldn't understand anything about coding and English.

For these students, the cognitive load of learning coding was especially high because they were struggling to understand English, both written and spoken, while learning to work with the HTML and CSS computer languages. To complete COSTI's introductory coding course, students needed high levels of determination and patience.

This language and communication barrier difficulty was an unremovable, defining characteristic of the course experience. English was not the native language for many of the women enrolled in this introductory coding course. Throughout this engagement, I wondered about possible approaches to better support these students. My field observations make mention of a complicating factor. In every session, there was a diversity of cultures. In the January 2023 weekday session, there were five women who regularly attended, each of them from a different country, including China, Albania, and Iran. In such a group, there was less of an opportunity to create a supportive community by grouping students according to their native country and/or language. In getting to know these students, I learned that employment topics were a good way to stimulate the conversation required for bonds among us to develop.

Ad hoc Ways to Better Respond to Student's Needs

To facilitate better communication between me and the students, I made some adjustments as I continued in this teaching engagement. For example, I realized during class that I could use the Zoom pen to show students what part of their code needed changes. While this was an ad hoc discovery I made while teaching, I made other changes based on more substantial reflection. I recorded in my field notes that I had puzzled these students by advising them that the corrective action for their code was to "add a semicolon." I had overlooked that my students as non-native English speakers would not necessarily be familiar with the names for common

punctuation marks such as the semicolon. In addition to semicolons, colons, double quotes, angle brackets, and curly braces are other important characters required to write functioning HTML and CSS code. Assuming prior knowledge of the terms for these punctuation marks was one of my "expert blind spots" (Nathan et al., 2001, p. 1). An expert blind spot refers to fundamental knowledge that students require to learn a topic and that teachers as subject matter experts assume students know (Nathan, 2001; Guzdial. 2015). According to Guzdial(2015), computing teachers are particularly prone to these oversights as he believes they tend to privilege content knowledge over pedagogical content knowledge. This was an instance in which I was able to gain important pedagogical content knowledge of teaching web coding to non-native English speakers. In the final session I facilitated in March 2023, I did provide this chart (Table 5) so that students had a visual aid with labels to support them in matching the keyboard characters with their English names. This example is one that shows how coding content needed to be adjusted for this group of students.

Table 5

Important Coding Characters and Terms in the HTML and CSS languages

Character	Name	Example
;	Semicolon	colour: white;
:	Colon	colour: white;
“”	double quotes	document.getElementsByTagName("h1")[0].style.fontSize = "84px";
<>	Angle brackets	<h1> Hello World!</h1>
{ }	Curly braces	body { font-family: system-ui; }

Impact on Collaborative Learning Activities

The language and communication difficulties also contributed to some students' frustration when I assigned students to work with each other in smaller group sessions through the Zoom break-out room feature. My intention was that group work made of up 3 or 4 students would help stimulate more interaction and help create a sense of community in the class. If I had at that time viewed my function as providing English language instruction, I would also have recognized this activity as a route to provide them with an interactive experience that would help develop their English communication skills.

What was influencing this decision more was my academic study of computational participation which emphasizes learning coding through social exchanges. Moving away from "individualistic view of programming," computational participation is promoted as a way to make learning coding more productive and relevant for broader audiences (Kafai & Burke, 2017, p. 394). In addition, I assumed that smaller groups would help these students feel less self-conscious as I would not be able to see or hear them. This judgment as I look back, was perhaps faulty, as I now understand through their survey responses and comments, that many of these students looked to me to provide reassurance. This was one example in which my well-meaning teaching intention did not lead to productive results (Brookfield, 1995).

The break-out group session activities had an opposite effect on some students than I intended. Mira, who impressed me as an active learner of web development, provided an account of a stressful group work situation. In conversation with me, Mira responded that the smaller break-out sessions were the least enjoyable and least effective activities of the course. She felt a lack of direction, discomfort, and confusion with these sessions that she was part of. Very little or no student conversation occurred in the group Mira was part of. Phrases Mira used to describe

the break-out room experience were “we just waste our time,” “a lot of people just were silent,” and “nobody can help us.” Being sent to a virtual room without much interaction was likely a strange, alienating experience to already tense students, many of whom like Michi struggled to understand course material delivered in a foreign language. While I would drop in on these sessions and check in with smaller groups, I found that I ended up spending more time than anticipated during my check-in with the first group. As a result, some groups did not get my equal attention which quite likely added to their sense of alienation. Student responses on the survey suggest that many of them may have felt as Mira did about the ineffectiveness of the online group work. A small percentage, 4% of students responded that they learned best through break-out group sessions; twelve percent of students said that the break-out group sessions was an activity that helped them learn. Instructor demonstrations and individual practice ranked much higher as useful learning activities among this group of students.

Given the language barrier and diversity among these students, it was naïve for me to expect that these students would work with ease in virtual smaller group sessions. Even with a more homogenous group of native English speakers, collaborative activities including group work are sometimes problematic due to interpersonal dynamics and tensions (Kafai & Burke, 2017; Popov et al., 2012). In their study of multicultural student group work, Popov et al. (2012) found many factors including “insufficient English language skills,” “diverse disciplinary backgrounds of members in one group,” “culturally different ways of interacting” and “culturally different styles of complying with supervisor's guidelines” adversely impact the effectiveness of group work (p. 311). While these are indeed significant and relevant challenges that make the facilitation of collaborative activities more difficult, these students, who have the

objective of establishing their lives in a multicultural country such as Canada, would benefit immensely from developing their collaborative abilities (Popov et al., 2012).

Despite student feedback regarding the futility of the online group work, I did see that some of these students recognized some value in learning coding through collaborative activities. Even Mira mentioned in her interview with me that communication skills are important for coders because “you don’t work alone.” In this virtual class, students were paying attention to what their peers were doing. In their conversations with me, both Mahvash and Mira expressed admiration for their peers’ work. Students also took some initiative in organizing groups outside of the class. In two separate classes, there were two cases in which students did initiate the creation of a WhatsApp group, one called Web Designing course and Web Design for Women. In both cases, they invited me to join their online group. While there were not a lot of exchanges in these groups, it does at least show that some of these students considered online social exchanges as being valuable to developing their coding literacy.

Effective Teaching Tools and Approaches

With the group work activity in the course being limited in helping students learn, there were some approaches that include software tooling that really helped mitigate the language and communication barrier challenge. In this section, I will describe the use of Codepen, a coding playground, and discuss how this tool increases students’ access to coding and effectively supports principles of computational participation (Kafai & Burke, 2017). While this part of the discussion reflects my view as the instructor of what effectively supported students’ learning, the next part of the discussion will focus on the student perspective on what worked for them. It will also include an analysis of the instructional approaches that students indicated best supported

their learning. In my analysis of instructional approaches, I used the Teaching Perspective Inventory as an external guide (Collins & Pratt, 2011).

Coding Playgrounds as Accessible Instructional Coding Tools

Coding playgrounds played a major role in increasing the accessibility of learning coding for COSTI's students. Incorporating coding playgrounds, such as Codepen, into the online classroom was much more than an ad hoc response to my students' needs. Identified in the early phases of my action research cycle, teaching coding using such a tool helped streamline the learning of coding for this audience of students, many of whom were undertaking the challenge of learning English. As browser software, coding playgrounds are self-contained code editors that eliminate the need to install and configure software on one's computer (Sonmez, n.d.). Unlike developer-grade code editors, the code created within a coding playground requires no creation or storage of files on one's computer. During the time of this engagement, I was also teaching web coding to a distinctly different group of students, who were enrolled in a semester-length HTML & CSS course at a Canadian college. With many of these students having specific goals of finding corporate employment as software engineers, these students gained relevant, authentic experience from working with developer-grade code editors, such as Visual Studio Code (Guzdial, 2015). The objectives for these women learning coding in this study were more exploratory. Rather their introduction to coding course aimed to provide a general learning experience to help them make an informed decision regarding pursuing further computing education. The choice of coding software is one way that web coding instruction might differ according to participants' objectives and interests (Guzdial, 2015; Street, 2006; Vee, 2017). Coding playgrounds as instructional tools were suitable for the students in COSTI's introductory coding courses. They supported the objectives of providing students with a general learning

experience that would enable them to make an informed decision regarding pursuing more computing education.

Guzdial (2015) theorizes that to broaden participation in computing among diverse audiences, there needs to be a thoughtful consideration of reducing superfluous content for people who might want to learn coding but don't necessarily aspire to become software engineers. In previous sessions, COSTI's students used tools such as Sublime and Caret, code-editing software that is equivalent to Visual Studio Code. Using such standard developer tooling significantly increases the computing skills these students would need to succeed in completing the introductory coding course. This specialized software requires computer updates as well as knowledge of accessing file systems and text-editing software. Another complicating characteristic of these types of specialized tools includes the fact that they are device-dependent. While Windows and Apple systems would work with Sublime, chrome books required Caret software. Providing and supporting separate sets of software installation instructions added setup demands that increased the possibility of overwhelming, fatiguing and confusing these students before they had the chance to read or write any computer code. I did attempt to use the standard code editors in the first sessions that I taught, and the results were disastrous. In the October 2022 session, I witnessed the steepest drops in attendance after facilitating a group work activity session devoted to helping students to familiarize themselves with these specialized tools.

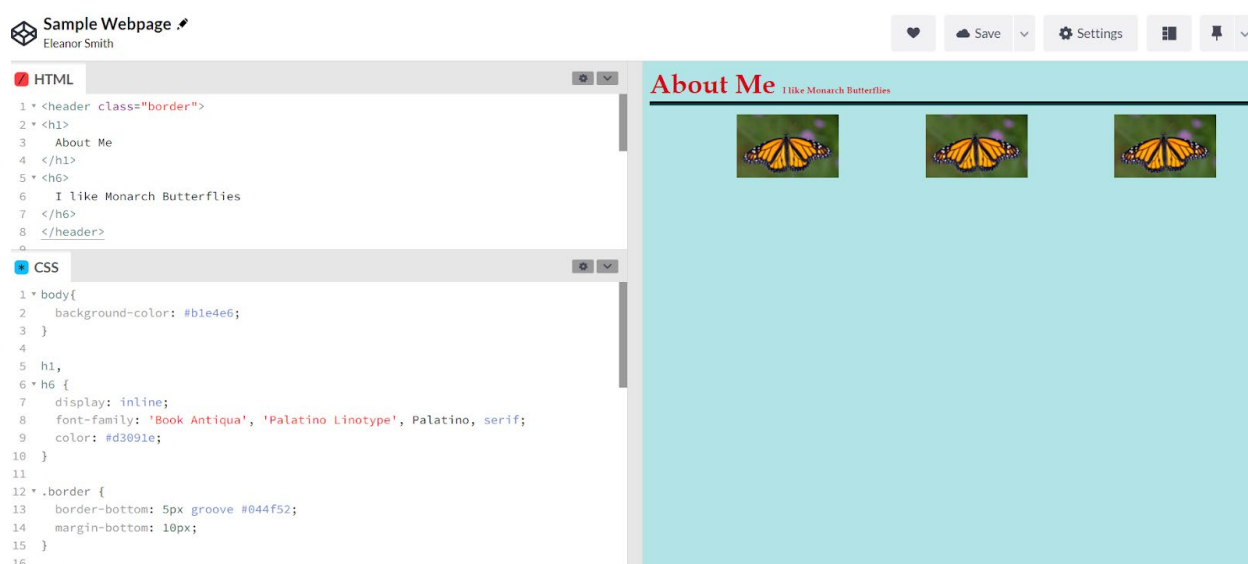
Coding Playgrounds as Better Tools for Novices

Reducing the software knowledge setup requirements, Codepen also included the capability of making code results immediately visible. Evita, who left the course with a new impression that coding was fun, described this capability as a helpful way to work with code "in real time." Once students wrote some instructions, they could see their results in the Codepen

environment (Figure 8). In contrast, working with standard code editors requires knowledge of switching from source code to web program view. While this might be something that would strike a software engineer as a trivial task, acquiring the practical knowledge of switching from the code view to the program view was something that would require a significant amount of class time in our short five-week course.

Figure 8

Codepen Development Environment: Real-time view



Note. Codepen made it easier for students to see the results of their coding. The left side is the code view while the right side is the web page view. As students wrote code, they would see their results immediately on the right.

Students, including Mira, Mahvash, and Michi, agreed that working with Codepen was productive for their learning. While students indicated that they saw some value in the use of Codepen, I recognized its value to a much greater extent. As a simpler alternative that significantly reduced student effort in working with code, Codepen proved to be a more appropriate software development tool for my group of students. Its features helped immensely in streamlining the beginning activities of the course. Through this tool adoption, I was likely able to retain a few more students for a few more sessions in subsequent cycles. For the next

three cycles with students using coding playgrounds exclusively, I never witnessed such a steep drop in attendance as the one that had occurred in the October session. From my data, there is a striking example of Codepen serving as a scaffold for engaging in more rigorous coding self-study. Months after our session ended, Mira embarked on her own self-study. She had advanced to using Visual Studio Code and learning Flexbox, a modern website layout method (MDN, n.d.). Using Codepen as a gateway into working with code, Mira progressed to being capable of teaching herself the tools and programming concepts that are standard in a rigorous semester-length college-level course.

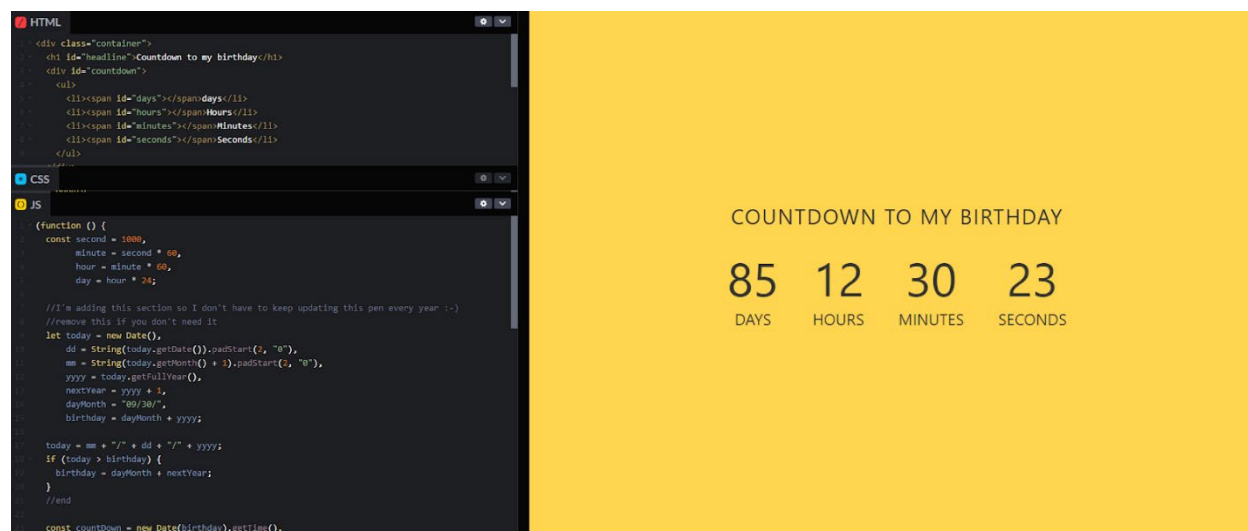
Coding Playgrounds as Supporting Collaboration

Another important, defining characteristic of coding playgrounds include their capability to allow users to “build, test and share code with colleagues and the community” (Sonmez, n.d., para 2). As a coding playground, Codepen is also a social development environment that provides public access to a collection of web programs from developers around the world. Not only can Codepen’s users browse code written by others, but its users can also make and save copies of public programs. Through this feature, users can modify and repurpose Codepen programs. Not only to be viewed as a scaffolding tool that streamlines web coding instruction, Codepen, lends itself to showing students the benefits of computational participation. Kafai and Burke (2017) conceptualize computational participation as a social practice that both broadens and deepens students’ engagement in learning to read and write code. Using Codepen as an instructional tool allowed me to illustrate these two important dimensions of computational participation: 1) “programming in a community” and 2) “programming as remixing code” (Kafai & Burke, 2017, p. 400). While the community dimension of computational participation sounds vague as being described as a way to learn coding, “with and from others,” the remixing

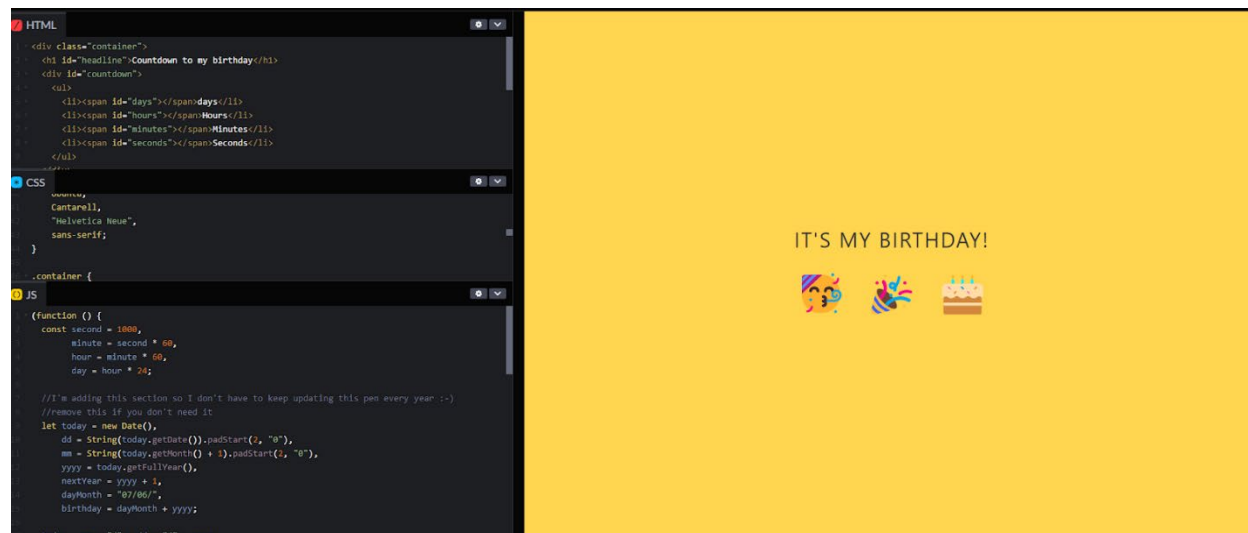
dimension is more specific and involves learning coding by modifying the coding of others (Kafai & Burke, 2017, p. 397). In the next part of this discussion, I will describe how I used Codepen to illustrate these two principles of computational participation.

Two Examples of Codepen Activities

In the first few weeks of the introductory coding course, students experimented with web programs that were publicly available on Codepen's website. One such program was a Countdown timer program (Figure 9, Figure 10). This program would display a message indicating how many days until a certain date based on the value of a JavaScript variable. In the class activity that used this program, students changed the value of the date and observed the effects on the display. An easy and fun activity, it not only illustrated how the three languages HTML, CSS, and JavaScript work together to form a web application but also introduced students to the online community code accessible to them through the Codepen website. Through this activity, students were increasing their coding literacy through computational participation. This activity helped illustrate how the community principle of computational participation might engage novice coders by providing an example of what coding could do. While there were difficulties in the course encouraging students to collaborate with each other, this activity was a more effective way to show them the value of learning through others' examples.

Figure 9*Codepen Countdown Timer Program*

Note. This is the result when the date variable does not match the current date.

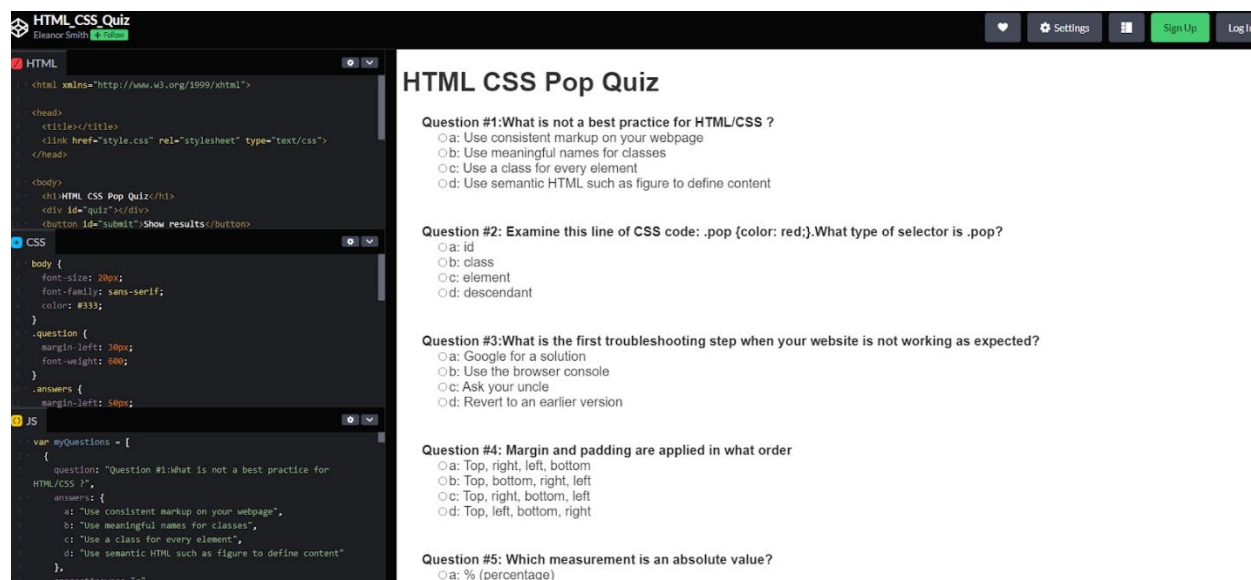
Figure 10*Codepen Countdown Timer Program*

Note. When the date matches the current day, then the birthday emojis appear.

Later in the course, students modified or remixed a quiz program I repurposed from Codepen (Figure 11, Figure 12). For this activity, students made their own copy of this program, known as “forking” in Codepen, and saved the program to their own work area. Then using the CSS language, students added visual effects, such as font styles and colours, frames, and background colours to their saved program. This was a timed, fun activity that challenged students to make as many styling changes as they possibly could without worrying about how garish their page appeared. Bringing the remixing principle into the classroom through this activity was fun and productive. It allowed students to get messy in making the web page appear as bold and bright as possible using their knowledge of CSS. As the instructor, this was an activity that I always enjoyed facilitating because students showed indications of their engagement by voluntarily sharing their work and asking questions about how to create advanced visual effects. There were a few times when my students asked me questions that required my own research.

Figure 11

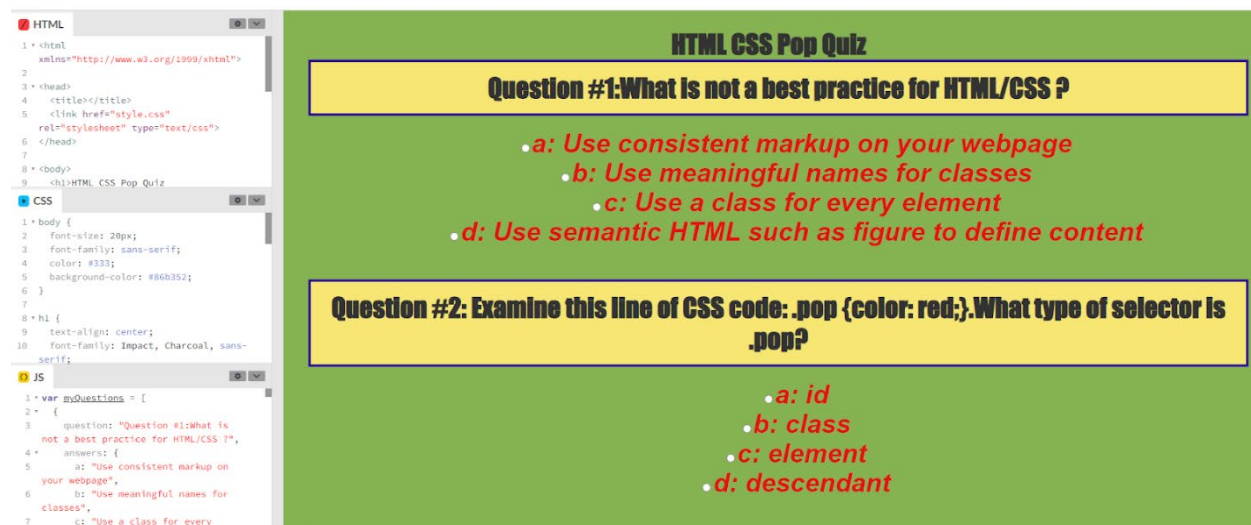
Codepen Program with Basic Styling



Note. This starting program is basic html code with a few styling flourishes.

Figure 12

Codepen Program with Styling



Note. This shows the original program with a lot more styling properties. Students were required to add as many CSS styling properties as they could.

Issues With Coding Playgrounds

There were some drawbacks in making use of the features of a coding playground that relate to computational participation. While not mandatory to start coding, an online account was required to save work and to make copies of other programs. To fully participate in these activities, students created a free online Codepen account that made all their work visible and available to the public. This drawback is a noted issue with computational participation (Kafai, 2016; Kafai & Burke, 2017). There will likely be students who will not feel quite comfortable with the idea of having their code fully visible for others' review (Kafai & Burke, 2017). As well students may not want others to be able to copy their work. The ability to copy and modify programs interestingly conflicts with traditional academic assumptions that coding is an individual endeavour and should be written from scratch to show knowledge (Benda et al., 2012, Kafai & Burke, 2017). In the college-level course that I was teaching at the same time, allowing students to submit work based on borrowed code was not a generally accepted way for students to work. In addition, this practice would raise questions regarding academic dishonesty protocols. This quiz program exercise that worked from borrowed code is an example of how this introductory program differed dramatically from the typical instruction provided in a standard academic program. In this way, there is a potentially disruptive aspect of teaching and learning with coding playgrounds.

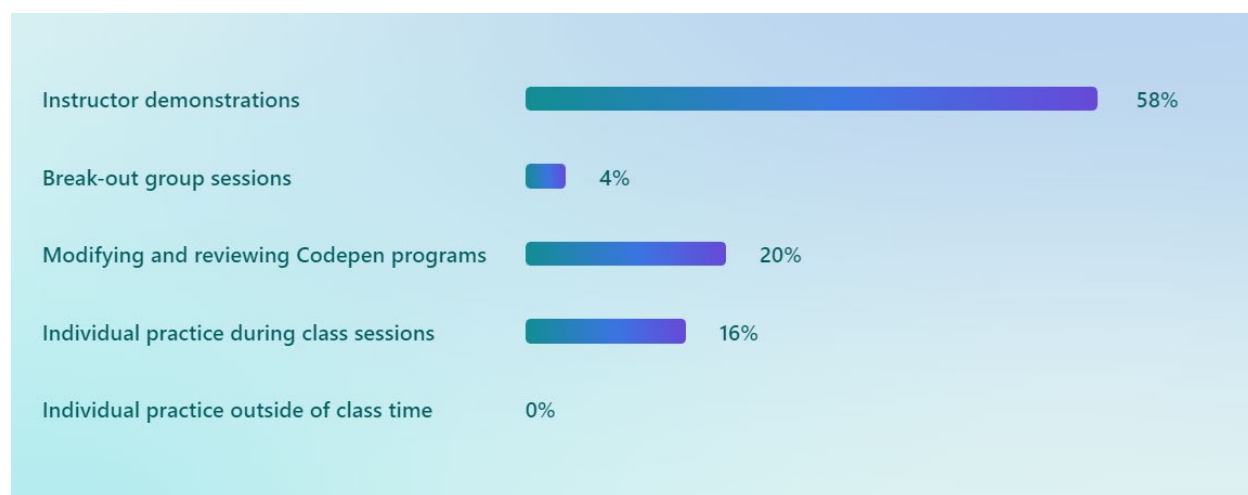
While using Codepen instead of standard developer software certainly reduced some instructional demands, it did add some requirements that included my monitoring of what students were sharing on their public work areas. Although there was an option to pay for a private Codepen workspace, this was one that none of us used. With my group of students, many of whom had difficulties understanding me, I was not sure that they fully comprehended that the

work they created on Codepen would be visible to anyone with internet access. Throughout the sessions, I kept an eye on their work to ensure that the content they were creating was appropriate. For instance, posting details such as their phone number and email address are data that could put them at some risk. With their work being shared with me through an internet link, I at least, was able to provide reasonable oversight on the content students were creating in the class.

Role of the Instructor

Student feedback on the most supportive course activities contrasts with my own assessments regarding the course experience. For much of my time teaching in COSTI's program, I was not certain that I was providing effective instruction, but I did think that I at least made a sound, astute decision in incorporating coding playgrounds. While student feedback was quite positive regarding my instruction, it is important to note that this feedback came from students who were successful in completing the course. The student feedback that I have access to represents one perspective that may not fully reflect reality.

The online survey questionnaire and interview comments show that students regarded me as their instructor as being critical to their learning. Figure 13 shows that 58% of these students (14 out of 24) rated instructor demonstrations as the course activity that they learned the most from.

Figure 13*Question 6 from the Online Survey Questionnaire*

Note. For this question, I asked students to indicate which activity they learned best from. This question was notable in being the question with the most agreement among students.

Interview comments suggest that these students conceptualized coding knowledge as requiring a responsive person to contextualize this subject matter rather than pre-recorded instruction or an AI tool such as ChatGPT. Some of these comments include “definitely need a teacher to understand and learn coding,” “for me, learning code through self-study is impossible,” and “I think everyone needs an instructor or a guide.”

The qualitative content from the online survey comments also reflects the importance that students placed on me as their teacher to guide their learning. Over 60 percent of the comments were personalized ones that included my first name. None of the comments included negative criticism that I have, on occasion, seen on other course evaluations. What is noteworthy is that in my survey design, I did not intend to solicit their opinions on the quality of my teaching. The survey prompt was quite generic in its instruction: “Write any more comments about the Introduction to Coding course experience below.” In phrasing the survey question this way, I

think a part of me was nervous that their assessments of my teaching would be negative as I was sensitive to the situation of high drop-out rates. Some of the comments that illustrate their regard for me include:

when I heard first time about coding I thought I would not be able to do this but Eleanor made it happen.

It was a good luck to have a teacher as Eleanor, she always was very attentive, patient and helping.

I would like to continue studying with Eleanor even if it is paid.

These are quite personally flattering comments with one student giving me the credit for her learning. These pleasing comments are not critical assessments nor are they reflective of all students who had enrolled in COSTI's introductory coding course. There were students who did drop out who likely did not think that fate served them well by having me as their teacher and who would not spend any of their own money on other courses I teach. Still, these positive assessments of me provide some insights. For instance, their comments convey respect for me as their teacher and might also indicate that they wished to please and impress me. These responses coming from the successful students also show a correlation between student success and respect for me and approval of my authority as the teacher. Their demonstration of respect towards me is reflective of Street's ideological view of literacy. The ideological perspective acknowledges that the opinions and feelings that their students have towards their teachers have a major influence on their learning (Street, 2006).

Using the Teaching Perspectives Inventory as a Starting Guide

Student feedback that indicated approval of my instruction surprised me somewhat and made me consider that I likely lack some awareness as to how I am relating to my students. In her comments to me, Michi used this vague but curious phrase "the ways that you taught us"

which made me wonder what was distinctive or characteristic of the instruction that I provided them. As a start to develop a better understanding, I took the Teaching Perspectives Inventory (TPI) survey, an online tool that helps teachers identify their own teaching beliefs, intentions, and actions (Collins & Pratt, 2011). The five teaching perspectives are transmission, developmental, apprenticeship, nurturing, and social reform (Collins & Pratt, 2011). Each perspective represents distinct approaches to teaching. For example, the transmission perspective privileges content knowledge above other values such as authentic application (apprenticeship perspective) or social impact (social reform perspective). The TPI is a well-known survey that Guzdial (2015) mentions as being particularly informative to computing teachers, whom he believes as a group need to work on increasing their pedagogical content knowledge of coding.

During this action research study, I took the TPI survey twice with my experience of teaching COSTI's students in mind. In both sets of survey results, I scored the highest on the nurturing perspective with the development approach as a close second. There are two things I did quite regularly in this teaching engagement that the TPI survey questions helped me see more clearly. The actions include 1) pointing out students' achievements to the entire class and 2) asking lots of questions to provoke students' thinking. While remarking on excellent student work matches with the nurturing perspective, asking questions is characteristic of the development teaching model. The nurturing teaching approach privileges building student confidence; the development approach values active learning that promotes student involvement through a variety of open-ended activities that might include discussions and group work (Kane, 2004). The development approach to teaching can possibly intimidate and overwhelm students who are not accustomed to this type of teaching and learning (Kane, 2004). I wondered what the reason for students was being unresponsive when I used the student feedback tool,

AnswerGarden, to solicit anonymous responses. Perhaps for some of these students, who withdrew from the course, my habit of trying to elicit their responses was seen as unproductive. Even one of the program's most successful students, Mira, considered the group work sessions, a developmental learning method, a waste of time. Given that instructor demonstrations ranked the highest as the most helpful learning activity, I believe that many of these students were not comfortable with developmental ways of learning that depend on their active participation.

One thing I would have liked to have done better in my teaching would have been to leverage the students' background and existing knowledge in this introductory coding course. There were a lot of students who were reluctant to share details about themselves in the course's introductory activities. It is quite likely that nervousness about communicating in English was part of this reluctance. At the start of the course, I asked each student to complete a document that requested them to briefly describe themselves including their computing background and expectations for the course. This document was a private one between me and each student. Completing this document was an activity that was voluntary as I strongly believe that it is not respectful to force students to share details about themselves. In each class session, only a few students completed the document. Given their reluctance in sharing details about themselves, I was limited in my capacity to make use of students' prior knowledge, a characteristic activity of developmental teaching. Finding more effective ways of supporting developmental learning for this group of students is something that required a greater awareness on my part (Kane, 2004).

My nurturing approach in the online classroom might be related to the quantity of personalized positive responses noted on the online questionnaire survey. Students shared comments on the survey that show their need to develop their confidence. Despite being educated and accomplished in their own native countries, some students admitted that they were

feeling insecure at the beginning of the course. Here are some comments that show their doubts and fears:

I was intimidated, shy and nervous in the beginning of the course. I thought I will not be able to understand any of it.

I cannot lie, it was scary at first to come to a new country and learn to code.

In addition, students indicated that they recognized my nurturing approach to teaching and were responsive to this approach. One student from the January session observed that I consistently motivated students. The language that Evita used in her written response emphasizes not how the class changed her thinking but how it had a positive impact on her morale. She said of my role in her learning “You were encouraging and made feel like I can do it and understand the lesson.” She also left the class with the view that becoming an effective coder is something that requires not only the mind but also in her words “the heart.” The idea that learning is connected to our feelings is central to the nurturing perspective (Collins & Pratt, 2011).

The nurturing and developmental way of teaching that I tried to deliver is compatible with the objectives of a community program designed to help its participants adapt to living in a new country. The nurturing approach can support developmental methods of teaching that require students to take risks. As my dominant perspectives, the nurturing and development model might serve as complementary approaches because the developmental approach, dependent on student risk taking, requires confidence that the nurturing approach helps develop. In addition, my ways of teaching coding literacy align with the ideological view of literacy that recognizes the importance of social dynamics in teaching and learning (Street, 2006; Vee, 2017). This type of approach to teaching would be less suited for more narrow instructional situations in which participants wanted to prepare for a specific certification within a short time frame.

Conclusion

This study sought to develop a more comprehensive understanding of what coding literacy is for my students who were female newcomers to Canada. Important factors to consider in this analysis include their views on the value of learning coding, particular challenges they faced, and teaching practices and tools that supported their learning. Examining their reasons for pursuing coding knowledge exposed a noted contrast between their initial reasons for participation and the realized outcomes of their participation in the course. While employment concerns attracted many of these students to this introductory course, some students realized other broader benefits of learning web coding that included strengthening their English language skills as well as their confidence. The language and communication barrier as a distinctive aspect of this course had a major impact on the course experience that required some specific adjustments for this audience of learners. For example, working with code through a coding playground tool such as Codepen was one major successful adjustment made for this group of students. As an effective way to illustrate the benefits of computational participation, using Codepen reduced tedious communications and setup tasks that threatened to demotivate students before they had a chance to work with any code. Finally, analyzing students' perspectives showed that these students held a fair amount of trust in me as the instructor to personalize and contextualize their learning experience. Together, these findings support the view that the successful development of coding literacy is highly dependent on social factors, such as student characteristics, and requires customized instruction.

CHAPTER FIVE: SUMMARY, DISCUSSION, AND IMPLICATIONS

This action research study examined computer coding as a literacy practice in adult learning communities. My study involved an analysis of my engagement teaching an online introductory web coding course to female newcomers to Canada. Participants of this study enrolled in the COSTI organization's introductory coding course mainly for the objective of increasing their job prospects. In looking at their particular situation, I hoped to gain meaningful insights that would contribute to a less limited and limiting coding pedagogy. This final chapter includes a summary and discussion that offer some theories for the results of the study. This final chapter also provides implications for practice and future research as well as a few concluding remarks.

Summary

The findings and discussion of Chapter 4 exposed a noted contrast between their initial reasons for participation and the realized outcomes of their participation in the course. Looking at students' reasons for continued participation highlight how COSTI's introductory course actually helped a few of its participants. One unanticipated reason for one student's continued participation was the development of her English language skills. This example suggests that coding literacy has the potential to suit broader and unexpected purposes. Recognizing that employment concerns were the main driver for initial participation offers possible explanations as to why some students chose to withdraw from the course. These explanations also consider broader social factors such as the particular situation of these women who sought access to the Canadian job market. Some of these women were already highly educated in their native

countries. Applying Street's (2006, 2009) ideological model of literacy adds possible insights that help explain reasons for some students' withdrawals and other students' engagement.

Discussion

In this discussion, I will expand upon the findings and discussion from Chapter 4 to develop a more complete picture of what coding literacy entailed for the women of this study. The first part of this section examines students' beliefs on the value of pursuing coding literacy and contrasts their beliefs with the benefits noted from this study. The next section will apply insights from literacy and social theories to offer possible explanations for the high student drop-out rates of COSTI's introductory coding course program. The third section will highlight how academic theory such as Street's model of literacy would explain the experience of COSTI's most engaged students.

The Value of Coding Literacy: Examining Students' Beliefs and Benefits

The results from this study showed that most students enrolled in COSTI's introductory coding course to provide them with the knowledge to obtain favourable employment in Canada. According to these students, coding literacy held the promise of providing them with a lucrative, rewarding and/or highly esteemed occupation. Mahvash, trained as a civil engineer, perceived that a coding occupation was one that was more highly respected in Canada than in her native country of Afghanistan. Mahvash's expectations for developing her coding literacy best epitomize Scribner's (1985) general conception of "ideal literacy" as being "simultaneously adaptive, socially empowering, and self-enhancing" (p. 18). Her choice to pursue coding skills and knowledge through COSTI's community program was an adaptive response to adjusting to a new life in a foreign country. In addition, her decision to learn coding holds the potential to enhance her own life by increasing her socio-economic situation. She might also become a role model for other women in her situation. If coding literacy does eventually provide Mahvash with

upward mobility, then she will become an inspiring example to other striving immigrant women in similar circumstances. Mahvash's perspective, in particular, helps us see the potentially "adaptive, socially empowering and self-enhancing" aspects of coding literacy (Scribner, 1985, p. 18).

Another student Evita saw value in the adaptive benefits of learning coding. For her, coding literacy was a way to obtain a flexible, part-time work to help her maintain a good standard of living during these times of high economic inflation. Mira, a university graduate who majored in social sciences, shared that she was engaging in her own self-study to become a front-end web developer because she realized that creating web pages was fulfilling to her. Their beliefs also show how notions of coding literacy influence people's decisions to learn coding and reflect the rhetorical power of literacy (Vee, 2017). While these students hoped that coding literacy would help them flourish in the Canadian economy, these hopes did lead to disappointment and disillusionment for some students such as Michi who realized that participation in the five-week introductory course by itself would not be enough to obtain a desirable job.

Michi was astute in recognizing that taking this course by itself would not qualify her for Canadian jobs that involved coding. Additional training and significant investment in time and possibly money for tuition would be required to obtain a lucrative job that involved coding. This realization may have played a role in the high drop-out rates I observed during my seven months of teaching in this program. While such disappointment is not surprising, there were some unexpected successes. A few students realized considerable benefits from participation. Close to withdrawing, Michi saw that the introductory course I was teaching was valuable in helping her increase her English language skills. Mira's experience is another example. Through her

participation in the course, Mira realized her own capabilities to become a front-end web developer and so taking this course contributed to her increased self-efficacy. While this study has no evidence that it directly helped any of these students secure good employment, the results of the study do show that the course did equip a few of these students, such as Michi and Mira, with language skills and self-confidence that would certainly help them advocate for themselves in job interviews.

Possible Theories for the High-Drop Rates

I have theorized that some students in COSTI's program left the course because they realized that obtaining a coding job would require significantly more study, time and money. I know as well from multiple sources of data that some women withdrew because they could not understand much of the content because of the English language delivery. Certainly, the language barrier was a significant obstacle for this group of students. I also believe there were other cultural challenges that may have affected some students' motivation for learning coding. The employment objectives of COSTI's students were similar to Byrd's (2020) example of African Americans who learned the same web development coding languages in a nonprofit boot camp for the purpose of boosting their job prospects. Byrd's discussion curiously but vaguely mentioned that the coding bootcamp had to do quite a bit more than teach students the technical aspects of coding. The program also prepared its students to adapt to standard workplace expectations. Byrd (2020) tells us that Clearwater's program taught their racialized students how to fit in with white culture so that they could gain entry to better jobs. While this helps support my argument that educational programs should not neglect social aspects of coding literacy, this also reflects a harsh reality that acquiring certain jobs often requires people to adopt and to conform to a dominant culture's exclusionary practices and customs (Cazden et al., 1996).

With regards to my study involving female immigrants, they may have come to this course already quite frustrated with their situation as newcomers to Canada. Canadian immigrants do report that they find the Canadian job market to not be particularly inclusive of them (Keung, 2023; Kusumajuda, 2022; Makkar, 2023). Some, such as Makkar (2023) who is an experienced architect, describes her frustrations of having to undersell her impressive educational and professional qualifications to get entry into the Canadian job market. Looking at my notes from my October class, I remember that there were several accomplished women from a range of professions. They included a medical doctor, a photojournalist, police officer, and a microbiologist. None of these women completed COSTI's introductory coding course. While I do not claim to know why they withdrew, I might say they were likely highly frustrated by the Canadian economy that is not overly hospitable to educated immigrants. Mahvash's account provides some support for this theory through her story. Despite her engineering training, work experience designing bridges in Europe as well as having a good command of the English language, she found that she only had access to "survival jobs" shortly after settling in Canada.

My realization of these women's situation was something that I was not fully aware of until I had analyzed the content of student interviews, particularly Mahvash's. Before interviewing Mahvash, I was not familiar with the term "survival job." In Chapter 1, I mentioned that I identified with my students at Mohawk College who were quite annoyed that they had to take a coding class. Having remembered being in a similar situation a few short years before, these students' complaints made a lot of sense to me, and I knew how to address their concerns by sharing my own difficulties learning to code. In retrospect, they were the easiest group of students I taught because I could readily empathize with their situation. I believe these students saw that I genuinely understood how they felt and would accept my advice. Few dropped out,

and a good many left my class not only with their coding credit and also an ‘A’ mark. In contrast, teaching the women in COSTI’s program was a significantly more challenging and confusing teaching assignment in part because I had not experienced their particular situation.

The academic literature on literacy and adult education adds further insight as to why dropout rates may have been high. Street (2009) mentions that well-meaning educational efforts, such as COSTI’s introductory coding program, are prone to fail if they use teaching and learning practices that are foreign to its audiences. For example, separating students into break-out rooms to work together on a coding example was not an effective learning practice for these students. Mira shared with me that this class activity was unproductive and strange: her group sessions were marked with silence. Another practice that did not work was having students provide anonymous input on online whiteboards. While I was trying to incorporate modern online tools to encourage students to actively participate, these attempts at engaging this group of students fell flat. As noted with mass literacy efforts, mainstream classroom practices can be alienating to diverse groups of students (Heath, 1982; Street, 2006, 2009; Vee, 2017). Through reflecting on this experience, I have realized that I need to be more cognizant of how my ways of teaching might seem strange to some students from other cultural backgrounds.

I also wondered why only a few of COSTI’s students completed an introductory document to briefly describe themselves, their computing background, and their expectations for the course. One possible explanation is that some may have not had the language skills to complete the document. Another explanation is that some of these students did not trust me enough to share these details about themselves. While I think I am showing respect to my students by wishing to learn more about them, some students may have thought I was being intrusive and not maintaining appropriate boundaries. Brookfield’s (1995) words reflect my

thinking on this situation when he says that “one of the hardest things teachers have to learn is that the sincerity of their intentions does not guarantee the purity of their practice” (p. 1).

Having provided some theories involving the high drop-out rates, I offer another perspective, the concept of “cultural competence” that provides additional insight. A term first used in social work in 1989, cultural competence is “a set of congruent behaviours, attitudes, policies, and structures that come together in a system or agency or among professionals and enables the system, agency, or professionals to work effectively in cross-cultural situations” (Flaskerud, 2007, p. 121). While this definition recognizes cultural competence as an organizational objective, it can also be an individual practitioner’s ongoing and long-term goal” (Flaskerud, 2007). Demonstrating cultural competence as a teacher means progressing beyond a pedagogy of cultural blindness that is naively unaware of the impact of cultural diversity on learning (Goode et al., 2020; Washington, 2020). A thorough development of cultural competence requires substantial “cross-cultural knowledge” that is an ongoing, long-term pursuit (Washington, 2020, p. 215). In addition to acquiring extensive cultural knowledge, cultural competence includes a deep analysis and awareness of one’s own biases and advanced skill to apply cultural knowledge to minimize misunderstandings that are prone to occur in a diverse classroom (Washington, 2020). My field notes and the student data from the interviews revealed multiple instances of confusion and misunderstandings in the group work and class activities. Reflecting on this teaching experience, I would say that my developmental level of cultural competence was a factor in the high drop-out rates. While I approached this teaching engagement with a curious mindset open to learning more about my students and their challenges, this openness and willingness, while certainly a helpful start, are not enough to address the needs of culturally diverse students. Further increasing my cultural competence

might help me more thoroughly answer my questions as to why some students did not respond to my teaching.

Reasons for Engaging Some Students

In wrapping up this part of the discussion, I will point out how insights from academic theory would explain the success I had engaging some of the women who participated in the introductory coding course. There were some women that I was able to connect with because they were able to identify with me as a mother and technology professional. In my small January session of five women, I did feel I connected with this group in a meaningful way when one of the women commented on my patience, I replied that my spouse and my children do not find me patient at all. I went on to mention my daily trials cleaning up my family's messes. This response got most of them laughing genuinely. In many online class sessions that were marked by awkward silences, it was no small feat to create some laughter. Later that week, a few of them sought my advice about marketing themselves to employers and chatted with me well after our online class had ended. These examples of engagement reflect the ideological view of literacy that sees meaningful, social interactions between students and the teacher as requirements for learning and literacy development (Street, 2006, 2009). Larabee(2003) also reflects Street's perspective and communicates this idea more directly. He says that if teachers manage to get their students to like them, then it is much easier to get students "to go along with the kind of learning they are working to foster" (p. 19).

It was the most engaged students who shared with me their opinions about the course. Their comments from the survey do show that I did manage to get some of these students to like me. While their overwhelmingly positive comments about me are not objective assessments of my teaching, they do suggest that student trust and respect for their teachers is closely tied to

learning and literacy development (Street, 2006). Mira stands out as the student who achieved the most by building on her experience in the introductory course to engage in an advanced study of web layout techniques. In her comments, she gave me some credit for her own impressive achievement by saying that luck had served her well by having me as her teacher. The fact that these students found instructor demonstrations the most helpful part of the course indicates the high levels trust and respect that they had in me to guide them through the course content. Through my nurturing approaches to teaching, I believe I was able to maintain their trust and respect. Students such as Evita admitted to feeling very intimidated at the start of the course. My nurturing approach to teaching helped her feel more energized and positive about her capabilities to learn coding. Learning engagement depended heavily on students receiving me positively as the teacher.

Implications for Practice

The results of this study provide some insights that would inform other educators in similar situations that involve teaching coding to diverse audiences. One key insight is that success in teaching coding technology to one group of students does not ensure effectiveness teaching another distinct group of students, especially when they are culturally diverse. In my experience, I noted that my first teaching job at Mohawk College went much more smoothly than teaching immigrant women through COSTI's program. I had thought that my teaching experience, professional experience collaborating with East Indian teams, and academic research on computing and literacy would be adequate preparation for teaching female immigrants in COSTI's program. This teaching assignment did cause me to question my ability to deliver instruction. Adopting a critically reflective perspective of my teaching was imperative to have the focused energy to continue teaching the women in this study (Brookfield, 1995). There were

times when I felt quite incompetent working with these students. Had I let my negative feelings overwhelm me and blame myself fully for what was not working, I would have not been able to help students such as Mira and Michi. Each distinct group of students brings their own set of dynamics and can introduce challenges that cause even the most experienced and knowledgeable of teachers to question and examine different approaches to teaching (Brookfield, 1995; Palmer, 1997). In my case, new approaches to teaching involve the long-term commitment to developing the cultural competence to support a culturally responsive computing pedagogy (Goode et al., 2020; Flaskerud, 2007; Washington, 2020).

There was a combination of practices, theories, and tools that supported the critical reflection of my teaching through COSTI's program. Recording my observations using field notes and taking the Teaching Perspective Inventory (TPI) survey were ways I increased my self-awareness of my teaching (Collins & Pratt, 2011). To support my inquiry and my own development as a teacher I took field notes to keep a record about what was happening in these online sessions. Another tool such as the TPI survey helped me more clearly see how I might be coming across to students. This survey is one that Guzdial(2015) sees as being helpful to computer science teachers who really need to develop their pedagogical content knowledge. While I knew from other teaching evaluations that my developmental approach to teaching sometimes made my students feel uncomfortable and annoyed, I did realize that my nurturing approach may have helped my students, such as Mira, forgive me for forcing them into uncomfortable situations such as the Zoom break-out rooms.

Together, Street's (2006) ideological model of literacy and Kafai and Burke's (2017) computing-specific perspective of computational participation were informative in helping me teach these students more effectively. Appreciating that learning coding is heavily dependent on

social factors, including collaborative work and students' background knowledge, is critical in teaching coding effectively. Both the ideological model of literacy and the computational participation perspective would support my use of coding playgrounds, accessible and interactive coding editors that make it easy for users to share code. Using an online coding playground allowed me and the students in the study to easily exchange coding examples by sharing web links. Like the high school teachers in Ericson et al.'s (2015) study, most of COSTI's students did not have a computing background. Interactive code editors allowed both the high school teachers and COSTI's students to build their confidence with coding more quickly. For these groups of novices installing specialized coding editors that professional software developers use was an unnecessary task. Using such specialized tools threatened to dampen their motivation before they had the opportunity to read and write computer code.

Implications for Future Research

One unanticipated and non-trivial result of this study involved the observation that writing HTML and CSS code helped a few students, such as Michi and Mira, develop stronger English language skills. Michi, in particular, indicated that learning English became her main reason for continuing in the course. While she thought my teaching techniques, class discussions, videos, and group work, helped her learn English, working with the web coding languages also contributed to her language skills development. While developing stronger language skills was not part of the academic arguments for learning coding listed in Chapter 2 Table 1, this particular result connects to diSessa's (2018) conception of computational literacy that sees coding as a vehicle of transformative learning. Pioneering scholars such as Papert and diSessa (2018) have made strong cases for the idea that learning coding is an effective and fun way for children to grasp advanced mathematical concepts such as vectors (Guzdial, 2015; Vee, 2017). Through this

study, there are suggestions that coding is not limited to aiding the learning of mathematical concepts.

This study did stumble on the idea that learning web coding overlaps with the acquisition of a second language. Learning HTML might be an effective way to help non-native English speakers develop English language proficiency. In this study, the markup language, HTML, and the styling language, CSS, were the coding languages that supported the development of a few students' English language skills. Stevens and Verschoor (2017) describe their practical experience working with Middle Eastern engineering students and recognize the potential of writing HTML as an ESL approach. Certainly, this possible approach is worthy of further research. Portnoff (2018) also notes foreign language instructional principles may inform more effective coding pedagogy. In his argument, leveraging foreign language principles in coding instruction has the potential to substantially improve student outcomes in learning to read and write computer code. Incorporating foreign language principles into coding instruction is another potentially powerful insight worthy of further examination (Portnoff, 2018).

Concluding Remarks

Through this action research study, I sought to develop my understanding of coding literacy, the major computing perspectives such as computational participation and my own pedagogical content knowledge of coding (Kafai & Burke, 2017; Shulman, 2013). At the end of this study, there are some words from others that are occupying my thoughts. Michi's phrase "the ways that you taught us" is suggestive of the highly personal and ambiguous qualities of teaching that can make a meaningful difference to students. Another thought-provoking statement for me is from Guzdial (2015): "We don't know the limits of good teaching" (p. 159). Both sets of words reflect the idea that the results from our teaching while potentially powerful

are not necessarily predictable. With regard to my study, I do not claim to have gotten close to reaching the limits of good teaching. Still, I do feel a real sense of fulfillment in witnessing how a few students in this study, such as Mira and Michi, developed their own coding literacies that better prepared them for their own life challenges.

References

- Adams, W. C. (2015). Conducting semi-structured interviews. In K. E. Newcomer, H. P. Hatry, & J. S. Wholey (Eds.), *Handbook of practical program evaluation* (pp. 492-505). Wiley Online Library. <https://doi.org/10.1002/9781119171386.ch19>
- Atkins, L. & Wallace, S. (2012). *Qualitative research in education*. SAGE.
- Benda, K., Bruckman, A., & Guzdial, M. (2012). When life and learning do not fit: Challenges of workload and communication in introductory computer science online. *ACM Transactions on Computing Education (TOCE)*, 12(4), 1-38. <https://doi.org/10.1145/2382564.2382567>
- Berry, M. & Kölling, M. (2014). The state of play: A notional machine for learning programming. *The 2014 Conference on Innovation and Technology in Computer Science Education* (21-26). <https://doi.org/10.1145/2591708.2591721>
- Bron, J. & Veugelers, W. (2014). Why we need to involve our students in curriculum design: five arguments for student voice. *Curriculum and Teaching Dialogue*, 16(1-2), 125-140.
- Brookfield, S. D. (1995). *Becoming a critically reflective teacher*. (1st ed.). Jossey-Bass Publishers.
- Burke, Q., O'Byrne, W. I. & Kafai, Y. B. (2016). Computational participation: Understanding coding as an extension of literacy instruction. *Journal of Adolescent & Adult Literacy*, 59(4), 371-375. <https://doi.org/10.1002/jaal.496>
- Byrd, A. (2020). Like coming home: African Americans tinkering and playing toward a computer code bootcamp. *College Composition and Communication*, 71(3), 426–452.
- Carter, S. (2006). Redefining literacy as a social practice. *Journal of Basic Writing*, 25(2), 94–125. <https://www.jstor.org/stable/43443829>

Cazden, C., Cope, B., Fairclough, N., Gee, J., Kalantzis, M., Kress, G., ... & Nakata, M. (1996).

A pedagogy of multiliteracies: Designing social futures. *Harvard Educational Review*,

66(1), 60-92. <https://doi.org/10.17763/haer.66.1.17370n67v22j160u>

Clark, J. S.; Porath, S.; Thiele, J.; and Jobe, M. (2020). *Action Research*. NPP eBooks.

<https://newprairiepress.org/ebooks/34>

CodePen. (n.d.). *About codepen*. <https://codepen.io/about>

Collins, J. B. & Pratt, D. D. (2011). The teaching perspectives inventory at 10 years and 100,000

respondents: Reliability and validity of a teacher self-report inventory. *Adult Education*

Quarterly, 61(4), 358-375. <https://doi.org/10.1177/0741713610392763>

Columbia. (n.d.). *It's a computing revolution in the liberal arts*.

<https://www.cs.columbia.edu/2016/its-a-computing-revolution-in-the-liberal-arts>

Coiro, J., Knobel, M., Lankshear, C., & Leu, D. J. (2014). Central issues in new literacies and

new literacies research. In J. Coiro, M. Knobel, C. Lankshear, & D. J. Leu (Eds.).

Handbook of research on new literacies (pp. 1–22). Routledge.

<https://doi.org/10.4324/9781410618894>

diSessa, A. A. (2018). Computational literacy and “the big picture” concerning computers in

mathematics education. *Mathematical Thinking and Learning*, 20(1), 3–31.

<https://doi.org/10.1080/10986065.2018.1403544>

Du Boulay, B., O'Shea, T., & Monk, J. (1981). The black box inside the glass box: presenting

computing concepts to novices. *International Journal of Man-machine Studies*, 14(3), 237-

249. [https://doi.org/10.1016/S0020-7373\(81\)80056-9](https://doi.org/10.1016/S0020-7373(81)80056-9)

- Ericson, B., Guzdial, M., Morrison, B., Parker, M., Moldavan, M., & Surasani, L. (2015). An eBook for teachers learning CS principles. *ACM Inroads*, 6(4), 84–86.
<https://doi.org/10.1145/2829976>
- Fincher, S., Jeuring, J., Miller, C. S., Donaldson, P., Du Boulay, B., Hauswirth, M., ... & Petersen, A. (2020). Notional machines in computing education: The education of attention. *The Working Group Reports on Innovation and Technology in Computer Science Education* (pp. 21-50).
- Flaskerud, J. H. (2007). Cultural competence: what is it? *Issues in Mental Health Nursing*, 28(1), 121-123. <https://doi.org/10.1080/01612840600998154>
- Floyd, S. (2022). *The past, present, and future direction of computer science curriculum in K-12 education* (Publication No. 29238984) [Doctoral dissertation, The University of Western Ontario(Canada)].ProQuest Dissertations Publishing. <https://ir.lib.uwo.ca/etd/8463>
- Franklin, A. (2019, August 6). *Coding vs. programming for beginners: What is the difference?* Goodcore. <https://www.goodcore.co.uk/blog/coding-vs-programming/>
- GitHub. (n.d.). *Let's build from here*. <https://github.com/about>
- Goode, J., Johnson, S. R. & Sundstrom, K. (2020). Disrupting colorblind teacher education in computer science. *Professional Development in Education*, 46(2), 354-367.<https://doi.org/10.1080/19415257.2018.1550102>
- Guo, P. J. (2017). Older adults learning computer programming: Motivations, frustrations, and design opportunities. *The 2017 CHI Conference on Human Factors in Computing Systems*, 7070–7083. <https://doi.org/10.1145/3025453.3025945>
- Guzdial, M. (2015). *Learner-centered design of computing education: Research on computing for everyone*. Morgan & Claypool Publishers.

- Guzdial, M. (2019, September 9). *Where a notional machine doesn't help: Javascript and the dom*. Computing Ed. <https://computinged.wordpress.com/2019/09/09/where-a-notional-machine-doesnt-help-javascript-and-the-dom>
- Heath, S. B. (1982). What no bedtime story means: Narrative skills at home and school. *Language in Society*, 11(1), 49–76. <https://doi.org/10.1017/S0047404500009039>
- Jacob, S. R. & Warschauer, M. (2018). Computational thinking and literacy. *Journal of Computer Science Integration*, 1(1). 1-21. <https://par.nsf.gov/servlets/purl/10073487>
- Jenkins, H. (2007, December 4). *Reconsidering digital immigrants*. Henryjenkins. http://henryjenkins.org/blog/2007/12/reconsidering_digital_immigran.html
- Juviler, J. (2022, June 17). *Semantic html: What it is and how it improves your site*. Blog.hubspot. <https://blog.hubspot.com/website/semantic-html>
- Kafai, Y. (2016). From computational thinking to computational participation in K--12 education. *Communications of the ACM*, 59(8), 26–27. <https://doi.org/10.1145/2955114>
- Kafai, Y.B., Burke, Q. (2017). Computational participation: Teaching kids to create and connect through code. In P. Rich, & C. Hodges (Eds.) *Emerging research, practice, and policy on computational thinking* (pp. 393-405). Springer, Cham. <https://doi.org/10.1007/978-3-319-52691-1>
- Kane, L. (2004). Educators, learners and active learning methodologies. *International Journal of Lifelong Education*, 23(3), 275–286. <https://doi.org/10.1080/0260/37042000229237>
- Keung, N. (2023, June 11). *I respect myself too much to stay in Canada: Why so many new immigrants are leaving*. Toronto Star. https://www.thestar.com/news/canada/i-respect-myself-too-much-to-stay-in-canada-why-so-many-new-immigrants-are/article_a9940db2-a98a-5c27-9b80-d0e95bdca02d.html

- Kusumajuda, W. (2022, October 25). *Is a survival job a good first step in Canada?* Canadian Immigrant. <https://canadianimmigrant.ca/careers-and-education/careers/is-a-survival-job-a-good-first-step-in-canada>
- Labaree, D. F. (2003). The peculiar problems of preparing educational researchers. *Educational Researcher*, 32(4), 13-22. <https://doi.org/10.3102/0013189X032004013>
- Macintyre, C. (2012). *The art of action research in the classroom*. David Fulton Publishers.
- Makkar, K. (2023, March 18). *I wanted to make Canada my home. Then I realized my degree was worthless here*. CBC. <https://www.cbc.ca/news/canada/first-person-degree-was-worthless-in-canada-1.6772923>
- MDN. (n.d.). *Flexbox*. https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Flexbox
- MDN. (n.d.). *HTML: HyperText Markup Language*. <https://developer.mozilla.org/en-US/docs/Web/HTML>
- Nathan, M. J., Koedinger, K. R., Alibali, M. W. & others. (2001). Expert blind spot: When content knowledge eclipses pedagogical content knowledge. *The Third International Conference on Cognitive Science*, 644-648. https://website.education.wisc.edu/mnathan/Publications_files/2001_NathanEtAl_ICCS_EBS.pdf
- Palmer, P. J. (1997). The heart of a teacher identity and integrity in teaching. *Change: The Magazine of Higher Learning*, 29(6), 14-21. <https://doi.org/10.1080/00091389709602343>
- Peredaryenko, M. S. & Krauss, S. E. (2013). Calibrating the human instrument: Understanding the interviewing experience of novice qualitative researchers. *The Qualitative Report*, 18(43), 1-17.

- Phillippi, J., & Lauderdale, J. (2018). A guide to field notes for qualitative research: Context and conversation. *Qualitative Health Research*, 28(3), 381–388.
<https://doi.org/10.1177/1049732317697102>
- Popov, V., Brinkman, D., Biemans, H. J., Mulder, M., Kuznetsov, A. & Noroozi, O. (2012). Multicultural student group work in higher education: An explorative case study on challenges as perceived by students. *International Journal of Intercultural Relations*, 36(2), 302–317. <https://doi.org/10.1016/j.ijintrel.2011.09.004>
- Portnoff, S. (2018). The introductory computer programming course is first and foremost a language course. *ACM Inroads*, 9(2), 34–52. <https://doi.org/10.1145/3152433>
- Ranjen, N. (2015, March 18). *Girls for Change*. [Video]. YouTube.
<https://www.youtube.com/watch?v=WI0tPLjt8N4>
- Rao, K., & Meo, G. (2016). Using universal design for learning to design standards-based lessons. *Sage Open*, 6(4). Article 2158244016680680.
<https://doi.org/10.1177/2158244016680688>
- Rosala, M. (2022, August 17). *How to Analyze Qualitative Data from UX Research: Thematic Analysis*. Nielsen Norman Group. <https://www.nngroup.com/articles/thematic-analysis>
- Royal, C. (2017). Coding the curriculum: Journalism education for the digital age. In R. Goodman & E. Steyn (Eds.), *Global journalism education in the 21st century: Challenges and innovations* (pp. 383-408). Knight Center for Journalism in the Americas. <https://live-journalismcourses.pantheonsite.io/wp-content/uploads/2020/06/GlobalJournalism.pdf>
- Scribner, S. (1984). Literacy in three metaphors. *American Journal of Education*, 93(1), 6–21.
<https://doi.org/10.1086/443783>

- Shephard, T. (2023, March 20). *URBAN HERO: Andrew Young founded Vaccine Hunters Canada, which helped millions get COVID-19 doses*. Toronto.
https://www.toronto.com/news/urban-hero-andrew-young-founded-vaccine-hunters-canada-which-helped-millions-get-covid-19-doses/article_020bd84b-7420-5657-97d8-b04d37f33e0e.html?
- Shulman, L. S. (2013). Those who understand: Knowledge growth in teaching. *Journal of Education*, 193(3), 1-11. <https://doi.org/10.1177/002205741319300302>
- Sonmez, John. (n.d.). *The 7 Best Code Playgrounds to Learn, Share and Experiment*. Retrieved July 4, 2023, from <https://simpleprogrammer.com/best-code-playgrounds>
- Statistics Canada. (2022, November 30). *Jobs in Canada: Navigating changing local labour markets*. <https://www150.statcan.gc.ca/n1/daily-quotidien/221130/dq221130b-eng.htm>
- Street, B. (2006). Autonomous and ideological models of literacy: Approaches from new literacy studies. *Media Anthropology Network*, 17, 1–15.
- Street, B. (2009). The future of ‘social literacies.’ In M. Baynham, & M. Prinsloo (Eds.), *The future of literacy studies* (pp. 21–37). Palgrave Macmillan UK.
https://doi.org/10.1057/9780230245693_2
- Stevens, V. & Verschoor, J. (2017). Coding and English language teaching. *TESL-EJ*, 21(2), n2
- Tedre, M., Simon, & Malmi, L. (2018). Changing aims of computing education: A historical survey. *Computer Science Education*, 28(2), 158–186.
<https://doi.org/10.1080/08993408.2018.1486624>
- The Logical Indian. (2016, April 7). *Dharavi teenage girls build apps to solve critical living problems in their slum*. <https://thelogicalindian.com/story-feed/get-inspired/dharavi-teenage-girls-build-apps-to-solve-critical-living-problems-in-their-slum>

- Tissenbaum, M., Sheldon, J., & Abelson, H. (2019). From computational thinking to computational action. *Communications of the ACM*, 62(3), 34–36.
<https://doi.org/10.1145/3265747>
- Vee, A. (2017). *Coding literacy: How computer programming is changing writing*. The MIT Press.
- Vincent, D. (2000). *The rise of mass literacy: Reading and writing in modern Europe*. Polity Press & Blackwell Publishers Ltd.
- Vista, A. (2020). Teaching coding as a literacy: Issues, challenges, and limitations. *Academia Letters*, 2.
- Voogt, J., Fisser, P., Good, J., Mishra, P. & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies*, 20(4), 715–728. <https://doi.org/10.1007/s10639-015-9412-6>
- W3C. (n.d.). *Making the web work*. <https://www.w3.org>
- Washington, A. N. (2020). When twice as good isn't enough: The case for cultural competence in computing. *The 51st ACM technical symposium on computer science education*, 213–219.
<https://doi.org/10.1145/3328778.3366792>
- Wilson, V. (2014). Research methods: Triangulation. *Evidence Based Library and Information Practice*, 9(1), 74–75. <https://doi.org/10.18438/B8WW3X>
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35.
<https://doi.org/10.1145/1118178.1118215>
- Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1881), 3717–3725. <https://doi.org/10.1098/rsta.2008.0118>


Wing, J. (2011). Research notebook: Computational thinking—What and why. *The Link Magazine*, 6, 20-23.

Wing, J. M. (2014). Computational thinking benefits society. *40th Anniversary Blog of Social Issues in Computing*, 2014, 26. <https://doi.org/10.1098/rsta.2008.0118>


Appendix A

Online Survey Questionnaire

Survey Questions

3. My main reason for taking this Introduction to Coding Course is to : (Select one option.) 

- ☐ Develop problem-solving and thinking skills
- ☐ Develop a new way to communicate
- ☐ Increase personal and professional productivity
- ☐ Increase my job opportunities
- ☐ Support social purposes and my community
- ☐ Other

4. If other was selected, write more detail below. 

Enter your answer

5. I also enrolled in this introduction to coding course for these reasons. (Select all that apply.) 

- ☐ Develop problem-solving and thinking skills
- ☐ Develop a new way to communicate
- ☐ Increase personal and professional productivity
- ☐ Increase my job opportunities
- ☐ Support social purposes and my community
- ☐ None of the above

6. I learned best through this learning activity. (Select one.) ☐

- ☐ Instructor demonstrations
- ☐ Break-out group sessions
- ☐ Modifying and reviewing Codepen programs
- ☐ Individual practice during class sessions
- ☐ Individual practice outside of class time
- ☐ Other

7. If other was selected, write more detail below. ☐

Enter your answer

8. I also learned through these learning activities: (Select all that apply.) ☐


- ☐ Instructor demonstrations
- ☐ Break-out group sessions
- ☐ Modifying and reviewing shared web programs
- ☐ Individual practice during class sessions
- ☐ Individual practice outside of class time
- ☐ None of the above

9. My biggest difficulty to learning coding in this course was 


- ☐ Online course format
- ☐ Learning the content/subject in a foreign language
- ☐ Using and accessing the tools (Codepen, JSFiddle)
- ☐ Learning how to work other computer technology (Google drive, file systems)
- ☐ Course speed too fast
- ☐ Course speed too slow
- ☐ Other

10. If you selected other above, write your answer below. 


Enter your answer

11. Other difficulties I had in learning coding were: (Select all that apply.) 


- ☐ Online course format
- ☐ Learning the content/subject in a foreign language
- ☐ Using and accessing the tools (Codepen, JS Fiddle)
- ☐ Learning how to work other computer technology (Google drive, file systems)
- ☐ Course speed too fast
- ☐ Course speed too slow
- ☐ None of the above

12. After taking this course, I would like to continue to learn more about computer coding. 


- ☐ Yes
- ☐ No
- ☐ Maybe

13. If No was selected, I no longer wish to learn computer coding because 

Enter your answer

14. If Yes was selected, I will continue to practice and learn computer coding through these ways 

Enter your answer

15. Write any more comments about the Introduction to Coding course experience below. 

Enter your answer

You can print a copy of your answer after you submit

Back

Submit

Appendix B

Sample Semi-Structured Interview Prompts

- 1) What were some reasons you took the Introduction to Coding at COSTI? Was this your first coding course?
- 2) Did you feel ready and well prepared for the Introduction to Coding course you took with me? Had any previous courses, jobs you held or other experiences helped you or prepared you for learning coding?
- 3) What are your professional goals and do you see having coding skills as a way to help you achieve them?
- 4) Do you see coding knowledge and skills as a way to help you in your personal/family life?
- 5) Can you remember how you felt at the beginning of our coding sessions in March? What did you expect at the beginning of the course?
- 6) How did you feel at the end of course after 5 weeks of coding sessions? Did any of your opinions or perceptions about coding change?
- 7) What activities and materials (instructor demonstrations, Lab notes, Codepen) in the Introduction to Coding course helped you learn best?
- 8) What activities in the Introduction to Coding course were least effective in helping you?
- 9) What connection do you see between knowing the English language and knowing how to read and write code? Do you think someone who knows no or little English can code in html and css?
- 10) How did I as an instructor help you learn? How important is the instructor in learning coding? Do you think you could have learned coding through self-study?
- 11) Did you think learning coding through an online Zoom session made learning easier or more difficult for you?

- 12) From your final assignment submission (your favourite coding meme), it appears you found some joy in learning how to read and write computer code. I loved what you did with this part of the assignment. Am I right in my perception that you found satisfaction in learning coding? If I am correct, could you describe in more detail as to how coding helped provide you with positive feelings?
- 13) Now that our course is over, are you continuing your study of coding? If you are not at the moment, are you planning to take other coding courses? If so, what types of courses are you planning on taking (online, college/university program)?

Appendix C

Informed Consent Online Survey

Date: Winter 2023

Project Title: Coding as a Literacy Practice in Adult Learning Communities

INVITATION

You are invited to participate in a study that involves research. The purpose of this study is about the teaching and learning of reading and writing computer code.

How does this affect me?

You will be asked to complete a survey that includes multiple choice questions and short answers.

Time to take the survey will be approximately 15 minutes.

What are the benefits to me?

Possible benefits of participation include having the opportunity to reflect on your learning and experience in this course. The research is to identify insights regarding improved ways to help students such as yourself read and write coding languages.

What are the risks to me?

There are minimal risks. You may feel uneasy about answering some of the questions. It is possible you will feel nervous to comment on your experience as a student in my course. I assure you that you will be allowed to leave any question blank. I will not see the results of survey until I have completed your evaluations for the certificate. The forms will be set up so that they are only sent to Dr. Collier upon submission. Your responses will not impact any services that you receive through the Costi organization.

How will my responses be shared?

Your responses you provide will be kept private. Your name will not appear in any report resulting from this study. With your permission, quotations may be used.

Data collected during this study will be stored on a private computer in my home. Data will be kept for one year after my paper has been submitted in case there are any questions about the data or the analysis, or if I choose to write a publication, after which time all electronic files will be deleted and any paper files will be shredded.

Access to this data will be restricted to the researcher (Eleanor Smith) and the academic supervisor (Dr. Collier)

Here is the link to privacy policy of Microsoft Forms :

<https://myaccount.microsoft.com/settingsandprivacy/privacy>

Do I have to participate?

No, participation in this survey is voluntary. If you wish, you may decline to answer any questions or participate in any component of the study. The Costi administration will not have knowledge of who engages in the research. The choice to participate or not participate will have no bearing on other services you receive through the Costi organization.

PUBLICATION OF RESULTS

Results of this study may be published in professional journals and presented at conferences.

Results will also be shared with the Costi organization.

Feedback about this study will be available from the researcher directly upon completion of the project in June 2023. For more information, please email me at es18cf@brocku.ca.

CONTACT INFORMATION AND ETHICS CLEARANCE

If you have any questions about this study or require further information, please contact Eleanor Smith at es18cf@brocku.ca or

Dr. Collier at dcollier@brocku.ca using the contact information.

This study has been reviewed and received ethics clearance through the Office of Research Ethics at Brock University.

If you have any comments or concerns about your rights as a research participant, please contact the Office of Research Ethics at (905) 688-5550 Ext. 3035, reb@brocku.ca.

Thank you for your assistance in this project. Please keep a copy of this form for your records.

CONSENT FORM

I agree to participate in this study described above. I have made this decision based on the information I have read in the Information-Consent Letter. I have had the opportunity to receive any additional details I wanted about the study and understand that I may ask questions in the future.

Appendix D

Informed Consent Interview

Date: March/April 2023

Project Title: Coding as a Literacy Practice in Adult Learning Communities

Principal Investigator (PI): Dr. Diane Collier, Associate Professor

Department of Education

Brock University

Faculty Supervisor Dr. Diane Collier

Student Principal Investigator (SPI): Eleanor Smith

Department of Graduate Studies in Education

Brock University

dcollier@brocku.ca

es18cf@brocku.ca

INVITATION

You are invited to participate in a study that involves research. The purpose of this study is about the teaching and learning of computer code and specifically how to read and write computer code.

WHAT'S INVOLVED

You can expect questions pertaining to your background (country of origin, education, profession in home country) and how you view coding as helping you progress professionally in your life. Other questions will involve past experiences in learning to code and your opinions in how coding knowledge and skills are best acquired.

POTENTIAL BENEFITS AND RISKS

Possible benefits of participation include providing you with the opportunities to reflect on skills developed in Canada. Another benefit involves practising your interviewing and conversational skills. The research is aimed to identify insights regarding improved ways to help students read

and write coding languages. There may also be risks associated with participation in this study that are minimal. You may feel uneasy about answering some of the questions. It is possible you will feel nervous to comment on your experience as a student in my course. I assure you that you will be free to not answer any questions that you prefer not to answer. I also assure you that you are able to withdraw from the study at any time without consequence. Your responses will not impact other services that you receive through the Costi organization.

CONFIDENTIALITY

The reflective question responses you provide will be kept confidential. Your name will not appear in any thesis or report resulting from this study; however, with your permission quotations may be used. If you withdraw during the study, you will be given a choice as to whether your data may be kept or destroyed.

Data collected during this study will be stored on a private computer in my home. Data will be kept for one year after my paper has been submitted in case there are any questions about the data or the analysis, or I choose to write a publication, after which time all electronic files will be deleted and any paper files will be shredded.

Access to this data will be restricted to the researcher and the academic supervisor. The program supervisor will not be aware of who participated in the study, but will receive the final report with anonymous, de-identified data.

VOLUNTARY PARTICIPATION

Participation in this study is voluntary. If you wish, you may decline to answer any questions or participate in any component of the study. Given the program supervisory will not have knowledge of who engages in the research, the choice to participate, not participate or withdraw will have no bearing on other services you receive through the Costi organization. Withdrawal will not impact the provision of compensation, a \$10 Starbucks gift card. You may withdraw until the advanced stages of analysis which would be roughly 2 months after the interview activity.

COMPENSATION

For your participation, you will receive a \$10 gift certificate from Starbucks.

PUBLICATION OF RESULTS

Results of this study may be published in professional journals and presented at conferences.

Results will also be shared with the Costi organization.

Feedback about this study will be available from the researcher directly upon completion of the project in June 2023. For more information, please email me at es18cf@brocku.ca.

CONTACT INFORMATION AND ETHICS CLEARANCE

If you have any questions about this study or require further information, please contact Eleanor Smith at es18cf@brocku.ca or Dr. Collier at dcollier@brocku.ca using the contact information provided above. This study has been reviewed and received ethics clearance through the Office of Research Ethics at Brock University. If you have any comments or concerns about your rights as a research participant, please contact the Office of Research Ethics at (905) 688-5550 Ext. 3035, reb@brocku.ca.

Thank you for your assistance in this project. Please keep a copy of this form for your records.

CONSENT FORM

I agree to participate in this study described above. I have made this decision based on the information I have read in the Information-Consent Letter. I have had the opportunity to receive any additional details I wanted about the study and understand that I may ask questions in the future. I understand that I may withdraw this consent until the final stages of analysis.

Appendix E

Email Invitation for Interviews

Everyone,

Hope you are all doing well. As part of my graduate research, I am hoping that some of you would like to share your opinions about your experiences learning to code. My research project is examining effective ways to help people learn to read and write computer code.

I have summarized the views of leading academics and developed my own views on this topic; what I hope is to also bring your voices into this research.

Adding your voices will really add to this project.

What I am looking for are volunteers to interview with me. Approximate time of the interview session will be 1 hour. I will ask you questions about your professional goals and how you view learning coding as helping you.

I am able to offer you a \$10 gift card (Starbucks or Tim Hortons) for your participation. Only the audio content of the interview will be recorded.

My research plan has been approved by Brock University and the COSTI organization. Your participation is voluntary and will not affect any other services you receive through COSTI.

Possible benefits to yourself are the opportunity to reflect on your own learning experience and also to practise your communication and interviewing skills.

If you would be interested in participating or have more questions, do let me know by replying to this email. I am hoping to arrange these interviews sometime in March at a time that is convenient to both of us.

Hope to hear from you

Thank-you,

Eleanor

Appendix F

Ethics Clearance Letter



Brock University

Office of Research Ethics
Tel: 905-688-5550 ext. 3035

Email: reb@brocku.ca

Social Science Research Ethics Board

Certificate of Ethics Clearance for Human Participant Research

DATE: 12/16/2022

PRINCIPAL INVESTIGATOR: COLLIER, Diane - Educational

Studies FILE: 22-134 - COLLIER

TYPE: Masters Thesis/Project

TITLE: Coding as a literacy practice in adult learning communities

ETHICS CLEARANCE GRANTED

Type of Clearance: NEW

Expiry Date: 12/1/2023

The Brock University Social Science Research Ethics Board has reviewed the above-named research proposal and considers the procedures, as described by the applicant, to conform to the University's ethical standards and the Tri-Council Policy Statement. Clearance granted from **12/16/2022 to 12/1/2023**.

The Tri-Council Policy Statement requires that ongoing research be monitored by, at a minimum, an annual report. Should your project extend beyond the expiry date, you are required to submit a Renewal form before 12/1/2023. Continued clearance is contingent on timely submission of reports.

To comply with the Tri-Council Policy Statement, you must also submit a final report upon completion of your project. All report forms can be found on the Office of Research Ethics web page at:
<https://brocku.ca/research-at-brock/office-of-research-services/research-ethics-office/#application-forms>.

In addition, throughout your research, you must report promptly to the REB:

- a) Changes increasing the risk to the participant(s) and/or affecting significantly the conduct of the study;
- b) All adverse and/or unanticipated experiences or events that may have real or potential unfavourable implications for participants;
- c) New information that may adversely affect the safety of the participants or the conduct of the study;
- d) Any changes in your source of funding or new funding to a previously unfunded project.

We wish you success with your research.

Approved:

Nicole Luke, Chair

Social Science Research Ethics Board

Note: Brock University is accountable for the research carried out in its own jurisdiction or under its auspices and may refuse certain research even though the REB has found it ethically acceptable.

If research participants are in the care of a health facility, at a school, or other institution or community organization, it is the responsibility of the Principal Investigator to ensure that the ethical guidelines and clearance of those facilities or institutions are obtained and filed with the REB prior to the initiation of research at that site.