INJECTING COMPUTATIONAL THINKING INTO COMPUTING ACTIVITIES
FOR MIDDLE SCHOOL GIRLS

A Dissertation in
Information Sciences and Technology
by
Heidi Cornelia Webb

@ 2013 Heidi Cornelia Webb

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2013
The dissertation of Heidi Cornelia Webb was reviewed and approved by the following:

**Mary Beth Rosson**  
Professor of Information Sciences and Technology  
Dissertation Advisor  
Chair of Committee

**Heng Xu**  
PNC Technologies Career Development Professorship  
Associate Professor of Information Sciences and Technology

**Lynette Kvasny**  
Associate Professor of Information Sciences and Technology

**Priya Sharma**  
Associate Professor of Education (Learning, Design and Technology)

**Madhu Reddy**  
Associate Professor of Information Sciences and Technology  
Graduate Program Director

*Signatures are on file in the Graduate School*
ABSTRACT

Advances in technology have caused high schools to update their computer science curricula; however there has been little analogous attention to technology-related education in middle schools. With respect to computer-related knowledge and skills, middle school students are at a critical phase in life, exploring individualized education options and starting to wonder "How will computing technology fit into my future?" My research investigates how the merging of computing skills and computational thinking concepts can be integrated into computer learning activities for outreach enrichment programs to provide middle school girls a positive learning experience.

The public school classroom of today provides many opportunities for students to use technology, but it is important to recognize that using does not equate to understanding. A decade ago it was important to ensure that all students acquired basic computer literacy and fluency with general tools like word processors and spreadsheets. However, in today’s world students need a broader set of computer-related concepts and skills to prepare them for their future education and career activities. For example, the ability to access, manipulate and make sense of large amounts of structured data stored in a database can be important in careers as diverse as health care, graphic design, and environment science.

With respect to gender and computer-based skills, research shows that both girls and boys express an interest in many different forms of technology throughout their childhood, but they tend to be attracted to different activities. For example, girls have been shown to enjoy creating stories where they can personalize the environment and work on role-playing scenarios (Kelleher and Pausch 2007). Boys and girls both enjoy creating games, but with boys their role-play tends to involve evil fantasy characters and situations where competition to race or be better than the
next person is important (Kafai 1996). Girls need an engaging learning environment which supports their creativity and interests. When girls are able to work collaboratively on problems which are interesting to them and fun, their confidence increases and they also realize there are other girls who share their same interests in technology activities (Hughes 2005).

Recently, high schools in the United States have shifted their computer-based courses from traditional programming courses to a curriculum inspired by foundational computational concepts (Goode and Margolis 2011). For example, instead of learning to sort bowling scores, students learn about computing and its relationship to the world around them; they do this by working on projects that produce computational artifacts. Programming is still a part of the coursework, but instead of being a focus for skill development, programming is presented as a tool that aids in discovery of new ideas and concepts (CollegeBoard 2010). More specifically, computational thinking (CT) skills and techniques have gained attention in K-12 education; CT includes problem solving, pattern recognition and algorithm design (Wing 2006). Thus far however, middle school education that includes CT concepts is not common; by the time students reach high school their beliefs about how computing fits into the world they live in may already have formed in an inaccurate fashion.

My dissertation examines the potential of teaching CT concepts to girls of middle school age. I chose to focus on girls because it is at this age that girls turn away from computing education, eventually contributing to a significant gender imbalance in the computing workforce. I explored a particular learning paradigm – scaffolded examples – as a technique for engaging young women from the very beginning in realistic programming activities. I discuss the result of this approach, including its impacts on the girls’ CT self-efficacy and career attitudes.
# TABLE OF CONTENTS

LIST OF FIGURES.................................................................................................vii

LIST OF TABLES..................................................................................................x

1 Introduction........................................................................................................1

2 Related Work .....................................................................................................5
   2.1 Computational Thinking .............................................................................5
   2.2 Middle School Student Development ....................................................10
   2.3 Scaffolding ...............................................................................................12
   2.4 Computing Attitudes of Middle School Girls ........................................15
   2.5 Tools and Software ..................................................................................17
   2.6 Approaches to Teaching Computational Concepts and Skills ............22
      2.6.1 The Context of Learning .................................................................22
      2.6.2 Stories ............................................................................................23
      2.6.3 Games ............................................................................................26
      2.6.4 Simulations, Visualizations and Modeling ......................................29
   2.7 Impacts of Computation-Related Education Programs ......................30
   2.8 Summary and Discussion .......................................................................32

3 Research Overview .........................................................................................34
   3.1 General Workshop Structure ................................................................34
   3.2 Refinements to the GCI Concept ............................................................36
      3.2.1 Computing Tools ............................................................................36
      3.2.2 Scaffolded Examples ......................................................................37
      3.2.3 Activity Workbooks .......................................................................40
      3.2.4 Data Collection Methods ................................................................43
      3.2.5 Data Analysis Methods ..................................................................44
      3.2.6 Research Participants .....................................................................47
   3.3 Research Study Synopsis .......................................................................48

4 Formative Research .........................................................................................51
   4.1 Fall 2011 (GCI-1): After-School GirlsCreateIT! Program ...................51
      4.1.1 Participants .....................................................................................52
      4.1.2 Procedures ......................................................................................55
      4.1.3 Materials .........................................................................................57
      4.1.4 Results ............................................................................................60
      4.1.5 Discussion .......................................................................................66
   4.2 Spring 2012 (GCI-2): After-School GirlsCreateIT! Program ................69
      4.2.1 Participants .....................................................................................69
      4.2.2 Procedures ......................................................................................71
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2.3</td>
<td>Materials</td>
<td>73</td>
</tr>
<tr>
<td>4.2.4</td>
<td>Results</td>
<td>76</td>
</tr>
<tr>
<td>4.2.5</td>
<td>Discussion</td>
<td>85</td>
</tr>
<tr>
<td>4.3</td>
<td>Summary of the Formative Research</td>
<td>87</td>
</tr>
<tr>
<td>5</td>
<td>GirlsCreateIT! Summer Camp</td>
<td>89</td>
</tr>
<tr>
<td>5.1</td>
<td>Participants</td>
<td>89</td>
</tr>
<tr>
<td>5.2</td>
<td>Procedures</td>
<td>91</td>
</tr>
<tr>
<td>5.3</td>
<td>Materials</td>
<td>93</td>
</tr>
<tr>
<td>5.4</td>
<td>Results</td>
<td>94</td>
</tr>
<tr>
<td>5.4.1</td>
<td>Pre- and Post-Workshop Surveys</td>
<td>94</td>
</tr>
<tr>
<td>5.4.2</td>
<td>Workbooks</td>
<td>97</td>
</tr>
<tr>
<td>5.4.3</td>
<td>Classroom observations</td>
<td>99</td>
</tr>
<tr>
<td>5.4.4</td>
<td>Interviews</td>
<td>101</td>
</tr>
<tr>
<td>5.4.5</td>
<td>Artifacts</td>
<td>106</td>
</tr>
<tr>
<td>5.5</td>
<td>Discussion</td>
<td>111</td>
</tr>
<tr>
<td>6</td>
<td>Prospects for Computational Thinking by Middle School Girls</td>
<td>113</td>
</tr>
<tr>
<td>6.1</td>
<td>Summary of Findings</td>
<td>114</td>
</tr>
<tr>
<td>6.2</td>
<td>Limitations and Future Work</td>
<td>121</td>
</tr>
<tr>
<td>6.3</td>
<td>Contributions and Conclusion</td>
<td>123</td>
</tr>
<tr>
<td>7</td>
<td>References</td>
<td>125</td>
</tr>
</tbody>
</table>

Appendix A: Comparison of Teaching Tools | 129 |
Appendix B: Scaffolded Example Activities | 130 |
Appendix C: Participant Surveys | 149 |
Appendix D: Interview Questions (GCI-3) | 161 |
Appendix E: Workbook Excerpts | 163 |
Appendix F: GCI-2 End of Day Questions | 172 |
LIST OF FIGURES

Figure 1 - Alice Programming Environment .........................................................24
Figure 2 - Scratch Programming Environment .......................................................25
Figure 3 - Scratch Fish Game ..................................................................................28
Figure 4 - GCI-1 Scaffolded Example in Alice 2.2 ...................................................38
Figure 5 - Activity Workbook Page ........................................................................42
Figure 6 - Research Study Overview .......................................................................49
Figure 7 - Evolution and Refinement of Research Methods .....................................50
Figure 8 - Frequency of Participants by Grade ..........................................................53
Figure 9 - Frequency of Participants by Grade ..........................................................53
Figure 10 - GCI-1 Animal Park Display ...................................................................57
Figure 11 - GCI-2 Grade Frequency .........................................................................70
Figure 12 - GCI-2 Age Frequency ............................................................................70
Figure 13 - Artistic changes to sprites and background .............................................81
Figure 14 - GCI-2 Storyline changes with new characters ........................................82
Figure 15 - GC-2 Alternate story ending ...................................................................82
Figure 16 - Custom sprite with added code ..............................................................83
Figure 17 - GirlsChoice Week Two player game ......................................................85
Figure 18 - GCI-3 Grade Frequency .........................................................................90
Figure 19 - GCI-3 Age Frequency ............................................................................90
Figure 20 - On your own project with custom sprites and backgrounds ....................107
Figure 21 - GCI-3 on your own time project with broadcasts ....................................108
Figure 22 - GCI-3 Corresponding broadcasting receives ..........................................109
Figure 23 - Second on your own project created on 3rd day of GCI-3 .......................110
Figure 24 - GCI-1 Alice Vet Story ..........................................................................131
Figure 25 - GCI-1 Vet Story in Alice 2.2 ................................................................. 131
Figure 26 - GCI-1 Vet Story with problem-solving prompts ........................................ 132
Figure 27 - Fish Food Fun Game - only bad food count showing .................................. 133
Figure 28 - Fish Food Fun- good variable count and code needed ................................. 133
Figure 29 - Drag and Drop interface to create queries .................................................. 134
Figure 30 - Webpage creation ......................................................................................... 135
Figure 31 - RCX Programmable Brick with sensors and motor ........................................ 136
Figure 32 - Robolab program to download to brick .......................................................... 136
Figure 33 - Olivia Story dialog problems ........................................................................ 137
Figure 34 - Broadcasts used in friends’ scene ................................................................. 138
Figure 35 - Ocean Fun with missing variable .................................................................. 139
Figure 36 - Earth Day Messages Screen .......................................................................... 140
Figure 37 - Earth Day Code Problems ........................................................................... 141
Figure 38 - Scratch Animation missing code ................................................................... 142
Figure 39 - Lego’s WeDo Tilt Sensor, motor and USB hub ............................................... 142
Figure 40 - Vacation Plans story with missing character ................................................ 143
Figure 41 - Billy code problems – Hide used when show was needed ............................. 144
Figure 42 - Vacation Maze with missing code ................................................................. 145
Figure 43 - Vacation Maze User Input Code ................................................................... 145
Figure 44 - Working with Data Activity - Language Translator ........................................ 146
Figure 45 - Language Translator Code Complexity ....................................................... 147
Figure 46 - Vacation Packing Tilt Sensor Code Errors ...................................................... 148
Figure 47 - GCI-1 Interactive Story Workbook Problem Solving Page ............................ 163
Figure 48 - GCI-1 Interactive Story Designing solutions Page ......................................... 164
Figure 49 - GCI-2 Working with Data Minimalist Instruction Example Page ..................... 165
Figure 50 - GCI-2 Sensors and Motor Workbook Analysis Page.............................166

Figure 51 - GCI-3 Introduction to Scratch in the Vacation Story Activity........................167

Figure 52 - GCI-3 Workbook Broadcasting Questions Page........................................168

Figure 53 - GCI-3 Game Activity Test and Debug Workbook Page...............................169

Figure 54 - GCI-1 Game Activity CT Probes.................................................................170

Figure 55 - GCI-1 Game Activity End of Day CT Probe .............................................171
LIST OF TABLES

Table 1 - Children living in households with computers and internet access ....................... 1
Table 2 - CT Capabilities and Dispositions (Barr and Stephenson 2011) ............................. 8
Table 3 - CT Concept/Computer Activity Mapping ............................................................ 9
Table 4 - Piaget’s Cognitive Functions .................................................................................. 11
Table 5 - Conceptual Category Descriptions (Webb 2012) .................................................. 18
Table 6 - Computing Tools under Consideration ................................................................. 21
Table 7 - Research Study Overview ...................................................................................... 35
Table 8 – Scaffolded Examples with Associated CT Concepts and Skills ......................... 39
Table 9 – Overview of Data Collection Methods .................................................................. 44
Table 10 - Questions for analysis of Scratch Projects .......................................................... 46
Table 11 - Activities and Objectives for GCI-1 .................................................................... 59
Table 12 - Career Inventory Ratings ..................................................................................... 60
Table 13 - Activities and Objectives for GCI-2 .................................................................... 73
Table 14 - New Computational Thinking Self-Efficacy Scale ............................................... 75
Table 15 - Attitudes Assessed via the Pre-Workshop Survey in GCI-2 ................................. 77
Table 16 - Attitudes Assessed via the Post-Workshop Survey in GCI-2 ............................... 77
Table 17 - Attitudes Assessed via the Pre-Workshop Survey in GCI-3 ............................... 95
Table 18 - Pre- and Post-Workshop CT Self-Efficacy Ratings, for combined sample from GCI-2 and GCI-3 ................................................................................................. 96
Table 19 - Attitudes Assessed via the Post-Workshop Survey in GCI-3 ............................... 97
Table 20 - CT Objectives and Evidence of CT Summary ....................................................... 114
Table 21 - Genres of CT Scaffolded Example Activities ..................................................... 116
Table 22 - Process to Create a Learning Activity ................................................................. 120
Table 23 - Application Categorization (from Webb 2012) ................................................... 129
1 Introduction

The number of students who have not seen or used a computer prior to the first day of school is decreasing each year. U.S Census data on the percentage of children living in households with computers and have access to internet usage is rising as shown in Table 1. (Cheeseman Day, Janus et al. 2005; Bureau 2010). However, mere use of computer technology does not equate to understanding how applications work, or recognizing the role that technology plays in everyday scenarios. Depending on the grade level computer education ranges from literacy to fluency. Though many school children of today are surrounded by technology in both school and their everyday lives, there is still a need to find applications that engage them as learners and help educators to meet their learning objectives. For the educator, finding the most effective tool or application for learning can be a challenge. Students at all levels of the K-12 pipeline need additional skills and understanding of technology applications to prepare them for the future.

Table 1 - Children living in households with computers and internet access

<table>
<thead>
<tr>
<th>Census Year</th>
<th>Age range</th>
<th>Lives in household with computers (%)</th>
<th>Lives in household with internet usage (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010</td>
<td>3 - 17</td>
<td>85.2</td>
<td>79.2</td>
</tr>
<tr>
<td>2003</td>
<td>3 - 17</td>
<td>75</td>
<td>49.8</td>
</tr>
</tbody>
</table>

In 2006 Jeannette Wing began a discussion on the need to expose students of all grade levels to computational thinking (CT) skills and techniques, which include problem solving, pattern recognition and algorithm design (Wing 2006). Others have reinforced this message and there is now a shift occurring in US high schools, converting computer science (CS) courses that used to focus on computer programming to a curriculum inspired by CT concepts, as well as integrating CT into other courses in the high school curriculum (e.g., journalism, general science) (Freudenthal, Roy et al. 2010; Ruthmann, Heines et al. 2010).

One illustrative program that has been favorably received by high schools is "Exploring Computer Science" (www.exploringcs.org). This curriculum consists of modules that can be interchanged and arranged to accommodate the many variations found in schools with computing curricula. Students learn about computing and its relationship to the world around them through projects that emphasize creativity in the building of computational artifacts. For example, one
module has the students learn about computing components through interviewing a family member for details in purchasing a computer, from the specifications discovered during the interview they are then able to further learn about how computing components meet functional requirements. Other modules have the students perform scavenger hunts on the internet to learn search tools and resources available for future school activities; through the use of games, stories and webpage creation students can also learn some basic concepts of software design and programming. Programming is still a part of the coursework, but is introduced and practiced as a tool that can be used to aid in discovery of new ideas and concepts (CollegeBoard 2010).

These changes occurring in the high school computing curricula are exciting, in that they point to an entirely different and more conceptual view of computing as a pervasive supporting technology in the world. To date however, little attention has been devoted to analogous options suitable for middle school students. This is surprising given the middle school child is considered a 'digital native, which implies knowledge and comfort with technology' (Bennett, Maton et al. 2008). That is, one might think that they are already exposed to a range of technologies and should be able to leverage that general use to explore more conceptual issues. Middle school is a pivotal time for both boy and girls; they are generally in a new school, can begin to select elective courses and are interested in exploring different types of extra-curricular activities. This general situation suggests that there might be room in their studies to introduce CT activities and projects.

Elementary and middle school students are often exposed to computer activities in summer camps (Adams 2007; Doerschuk, Liu et al. 2009; Lau, Ngai et al. 2009). However, girls and boys do not have the same interest levels when it comes to coursework and extra-curricular activities. Programs which focus on one type of activity may not provide a young woman at the middle school age level with the motivation to learn more about computing technologies. Many of the successful programs have guided participants to create a game, interactive story or solve a problem with a Lego robot; these activities are fun and engaging for the students (Kelleher and Pausch 2006; Werner, Denner et al. 2009). For this reason it is important to consider computing tools and related activities that are of interest to middle school girls. There may also be options for after-school programs that include computer-based activities, but not all school districts are able to offer such programs. Even in schools that do offer such programs, rarely are the programs aimed at evoking an interest in how computers connect with the world around them.

With computer technology having such a broad impact on our lives now and in the future, it is important for children at this age level to engage in activities that help them see how computer concepts and skills can help to solve a wide variety of problems. Also, many students
in middle school may not realize that CT may play a role in many real world domains, well beyond activities for which they have already used computers like music or photography. In my dissertation I am investigating whether and how computer technology and scaffolded CT learning activities can be integrated into an after-school learning environment, with a particular focus on middle school girls. This dissertation project is addressing these general research questions:

**R1:** "What CT concepts and skills are comprehensible and usable by girls of middle school age?"

CT concepts and skills are often discussed in the literature using broad definitions and at levels appropriate for a university level theory course. Often CT concepts overlap one another and the skills needed to understand an abstraction needed in a problem solution may be the same as those needed to decompose and understand the problem which needs to be solved. In fact, even before entering middle school students have been exposed to some of the conceptual methods and skill sets that underlie CT. For example, elementary school aged children begin using problem-solving and abstraction in Mathematics, English and Geography when they study word problems, learn about sentence structures and grammar rules, and work with maps. However, prior to middle school the problem sets tend to be concrete and scaffolded to aid the student in learning the foundations needed to work with complex problems. As children develop their cognitive capacities, they should be able to explore and learn about CT in a more abstract and generalizable fashion. I propose to investigate which and how CT concepts and skills can be injected into activities that are engaging and comprehensible by middle school girls. This question will be answered qualitatively with data gathered during the learning activities as well as more quantifiable data gathered at points through the process.

**R2:** “How can we use the educational strategy of scaffolded examples to engage middle school girls in computer activities, while also introducing and building an understanding of CT concepts and skills?”

In a problem-based learning environment scaffolding can provide students who are new or have limited experience with computing tools an opportunity to receive the level of assistance needed to complete complex tasks. Even with a computing tool which has never been used by a novice user, an appropriately organized introductory sequence can provide a minimal amount of instruction that engages the user to produce an artifact; success at this should elicit a desire to do more (Carroll 1990; Carroll and van der Meij 1998). I propose to investigate a particular form of
the minimalist approach to education – scaffolded examples that have been carefully constructed to convey important concepts and where the learning activity is a combination of example exploration and modification (Rosson and Carroll 1996). The result is an activity that introduces concepts and skills but at the same time can result in an artifact complex enough to be meaningful and engaging to the learner. In related work, covering only what is necessary and still relevant to the completion of an activities task has been successfully used to introduce high school girls to learning website development from interactive databases (Rosson, Ioujanina et al. 2009).

Because CT concepts and skills inherently build on one another, scaffolding provides a means to create activities for middle school girls, provide a supportive environment for learning and motivate girls to become active members of their learning process. The research question concerning the design and effectiveness of the scaffolded examples for CT learning will be answered using qualitative data gathered as part of the learning process.

R3: "Will learning activities that introduce CT concepts skills lead to enhanced feelings of computer self-efficacy and improved attitudes about careers that involve computing? If so why?"

The increase of computing technology in schools has enabled students of all ages to learn new skills and develop more confidence in considering computer applications as a needed resource for their classwork. However, a student’s computer self-efficacy is also determined by the receiving of cues from within their environment, from peers and teachers (Schunk 1991). Though these cues are important to the social learning context of the student, they do not provide information on the impact of the activity or the student’s attitude on their own sense of accomplishment. Past experiences when a task could not be completed may have left a student feeling she is inadequate in her ability to use a particular technology. For students who have set high goals for what they should be able to accomplish, but who then find the computing tool working against them in finishing their task, their self-perceptions of abilities and interest may diminish. For young women who are working with computing tools, any self-doubts may translate into beliefs that they are not well suited for work with computers in a future career. In contrast, when a person’s perceived self-efficacy is high there tends to be aspirations to learn a new skill to accomplish one’s goal (Bandura, Barbaranelli et al. 1996). The assessment of self-efficacy and computing attitudes will be done using standard pre/post self-report data; the exploration of why such changes may occur will involve drawing connections between aspects of the girls’ learning process and the extent to which these indicators change.


2 Related Work

Research in the K-12 pipeline and in colleges has included revisions to curricula aiming to integrate computer applications into more courses, recruiting and retention of students who are interested in majoring in Information Technology, and conducting summer outreach programs for younger students. Many of the after-school programs and summer camps involve using a variety of software application tools for creating webpages and performing science experiments, all the while engaging the student in learning new technology skills (Clegg, Gardner et al. 2006; Hardnett 2008; Maloney, Peppler et al. 2008; Rodger, Hayes et al. 2009) However the primary goals of these programs have been in outreach and awareness-raising among students and their families. For example after school and summer programs for young people are often used by a school to make connections with the students and generate interest in technology, engineering and mathematics. Despite the reported success of many such outreach programs, the majority of students – particularly females and minorities – still avoid computing-related coursework in high schools and college. Unless students have success with fundamental concepts and skills in computing, and build from these successes to practice and extend these skills as their education continues, their computer-related self-efficacy will remain low and they are unlikely to see advanced education or careers that rely on CT.

2.1 Computational Thinking

Some aspects of CT, for instance problem decomposition, algorithm design and problem abstraction (Wing 2006) are not new concepts in education. Similar learning objectives are addressed by many science curricula. However, given the increased role of computer-based tasks
in many every day and professional activities, it is important to more fully understand the role of
CT across the curriculum, including at younger ages like middle and high school. In an effort to
move in this direction, the following broad definition provides a starting point for considering
how CT might be integrated into the K-12 academic environment.

Computational thinking is the thought processes involved in formulating problems and
their solutions so that the solutions are represented in a form that can be effectively
carried out by an information-processing agent. [CunySnyderWing10] (Wing 2011)

Beginning from this definition, we can begin to consider how CT may be relevant within
various disciplines. The key element in this definition is that the problem formulation and
problem solving is done in support of an information-processing solution – in other words some
way for computers to implement the problem solution. In mathematics, engineering and many of
the sciences, it would be difficult to find course work that does not involve some form of problem
decomposition and abstraction. The role of algorithm is less pervasive, but certainly the general
concept of formulating a logically ordered and complete set of steps is common across
disciplines. However CT emphasizes that the problem analysis and solution should be of the sort
that can be carried out by a computer. Some of the simplest and accessible examples could
involve development of numerical models in spreadsheets; more complex examples might be the
use of more specialized simulation or data analysis tools or languages.

In computer science of course, the role of CT is more obvious. Algorithm design is not
only essential to application or system programming, but also to more general computer-based
concerns such as usability, privacy and security. Other disciplines such as art, music or literature
may not depend on CT concepts, but yet many are discovering that computers can be used to
enable CT concepts they have been doing along (Ruthmann, Heines et al. 2010; Wolz, Stone et al.
2011). For instance, designing and implementing a hypermedia document depends on basic CT
concepts, as does the manipulation or conversion of different forms of media (Guzdial 2004).
While programs and disciplines are beginning to embrace CT concepts the computer science education research community still has no clear definition of how to approach and operationalize CT as a research construct. Even if a basic definition could be provided, it is not at all clear how it might be modified or elaborated to consider the developmental capacities of students at different age levels. Added to that is the large community of K-12 teachers who are offered at best a vague definition with little or no structure for curricula while they struggle to keep up with the latest technology landscape. One conceptual objective of the proposed research is to contribute to the need for better and more precise operational definitions of CT.

There is an emerging body of work done investigating CT concepts and computer science education for the K-12 community. Repenning (Repenning, Webb et al. 2010) has developed curricula emphasizing CT tools that have been focused on teacher training and authoring. In addition, a group of researchers and practitioners have initiated discussions about CT as part of the K-12 curriculum objectives and teaching approaches. In the summer of 2009, the Computer Science Teachers Association (CSTA) and the International Society for Technology in Education (ISTE) began a multiphase project aimed at developing an operational definition of computational thinking for K-12 (Barr and Stephenson 2011). Table 2 lists the capabilities and dispositions that have been proposed as a way to add CT to K-12 teaching. Because it is the result of thoughtful discussions by stakeholders intimately involved in K-12 teaching standards, I have elected to rely on it as a starting point for my own research questions, methods, and study materials.
Table 2 - CT Capabilities and Dispositions (Barr and Stephenson 2011)

<table>
<thead>
<tr>
<th>Capabilities (What students would actually do)</th>
<th>Dispositions (areas of values, motivations, feelings, stereotypes and attitudes applicable to CT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Design solutions to problems (using abstraction, automation, creating algorithms, data collection and analysis)</td>
<td>• Confidence in dealing with complexity</td>
</tr>
<tr>
<td>• Implement designs (programming as appropriate)</td>
<td>• Persistence in working with difficult problems</td>
</tr>
<tr>
<td>• Test and debug</td>
<td>• The ability to handle ambiguity</td>
</tr>
<tr>
<td>• Model, run simulations, do systems analysis</td>
<td>• The ability to deal with open-ended problems</td>
</tr>
<tr>
<td>• Reflection practice and communicating</td>
<td>• Setting aside differences to work with others to achieve a common goal or solution</td>
</tr>
<tr>
<td>• Use the vocabulary</td>
<td>• Knowing one's strengths and weaknesses when working with others</td>
</tr>
<tr>
<td>• Recognize abstractions and move between levels of abstractions</td>
<td></td>
</tr>
<tr>
<td>• Innovation, exploration, and creativity across disciplines</td>
<td></td>
</tr>
<tr>
<td>• Group problem solving; and Employ diverse learning strategies</td>
<td></td>
</tr>
</tbody>
</table>

In my research work, I am most interested in the CT capabilities from Table 2 that relate to problem solving and abstraction as it applies to computer programming. The learning activities used in my research, which include a mix of interactive stories and games, have operationalized the general concepts of design and implementation of software solutions (i.e. algorithmic thinking)(Cooper, Dann et al. 2000), the use of software-related abstractions in problem solving, and the general understanding of computing vocabulary. This concentration of concepts all have a foundation in computer programming, the standard mechanism to implement an information-processing solution, to use algorithmic thinking. Table 3 maps the general CT concepts to these operational computer programming concepts.
### Table 3 - CT Concept/Computer Activity Mapping

<table>
<thead>
<tr>
<th>CT Concepts</th>
<th>Computing Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem Solving</td>
<td>Identifying important pieces of problem</td>
</tr>
<tr>
<td>o Analyze</td>
<td>Brainstorming about computer solutions</td>
</tr>
<tr>
<td>o Design solutions</td>
<td>Organizing solution steps as an algorithm</td>
</tr>
<tr>
<td>Algorithmic thinking</td>
<td>Use of specific software concepts</td>
</tr>
<tr>
<td></td>
<td>Expressions</td>
</tr>
<tr>
<td></td>
<td>Control Structures</td>
</tr>
<tr>
<td></td>
<td>I/O</td>
</tr>
<tr>
<td></td>
<td>Data Types and Structures</td>
</tr>
<tr>
<td></td>
<td>Searching</td>
</tr>
<tr>
<td></td>
<td>Sorting</td>
</tr>
<tr>
<td></td>
<td>Test and Debug</td>
</tr>
<tr>
<td>Abstractions</td>
<td>Use of variables and alternate solutions</td>
</tr>
<tr>
<td>Understand the Vocabulary</td>
<td>Communicate through written documentation the problem solving process of problem decomposition</td>
</tr>
<tr>
<td></td>
<td>Using programming vocabulary (coding)</td>
</tr>
</tbody>
</table>

Computing activities invariably involve some form of problem solving with the design of solutions and algorithmic thinking to implement them. The activities developed for this research work have concentrated on problem solving and abstraction as it operates within the design and algorithmic thinking needed for information-processing solutions. The level of abstraction needed for the solving of specific problems can be tailored from simple learning about the storage of information using a single variable to the complexity of list processing using searching and sorting. As algorithmic thinking is used as to facilitate the design of solutions in computing environments there will always be some form of testing and debugging, which is why I have included these activities within the learning process. For there to be a dialogue between an instructor and student an understanding of the vocabulary used for the subject matter is needed in situations of requests for help and further clarification of the original problem. Vocabulary is always important for learners, and particularly when new concepts are being covered and the learner needs to communicate with a teacher to complete an activity. There is an opportunity to
developing learning activities with CT concepts and computing activities integrated to provide a foundation for middle school students.

2.2 Middle School Student Development

Working with computers can raise unexpected side-effects or complexities. The forms of abstraction that are inherent to computational problem solving can become overwhelming to younger children. For example, the concept of a variable in mathematic equations is often introduced to create generic formulas; however in computing the data type of a variable becomes an important concern. For instance a variable that is Boolean can support very different logic and operations than one that is a string. If a data structure such as a list is created as a variable, the complexity increases even more, as now the problem solver must consider how and when different elements need to be processed for an algorithm to work correctly.

Given these complexities, it is important to consider from the start what is known about middle school children’s intellectual development and learning capacities. A common source for such discussions is the work of Piaget on stages of cognitive functional development (Table 4). These stages of development have been shown to remain fairly sequential for all children, although it is also likely that rate of development is not the uniform, and culture may have important effects on the development of the child (Piaget 2008).
When designing CT learning activities for middle school students, it is important to align them for the capacities of children who in general are 11 to 14 years of age. According to Piaget’s stage theory, the capacity for formal operations (abstract and logical thought) is just beginning to emerge as a child reaches 11 to 12 years of age, and this capacity continues to develop until the age of 14 to 15 years. Thus care needs to be taken to structure activities appropriately (Piaget 1975). For students who are still processing concrete operations a story problem should present the necessary facts needed for a solution, and the possible solutions will most likely be just one. Once a child is able to work a story problem with abstract reasoning they can see more solutions and may be able to find all the possible outcomes for a solution. However, anticipating the general reasoning level of a middle school child is just part of the picture, it is also important to consider how one might assess learning.

How does one know if a child has learned? Piaget suggests that the child needs to be able to re-invent the subject that is being taught (Piaget 1975). While repeating the material back to the teacher is a good start, repeating the material verbatim is not a re-invention of the subject matter. The student needs to have learned the material well enough that she can repackage it and create a new process or application from the subject. This requires the student to discover for herself the true meaning of the material that has been taught (Piaget 1975). The problem is that

<table>
<thead>
<tr>
<th>Periods</th>
<th>Piaget’s Four periods of Cognitive Functions</th>
<th>Ages</th>
<th>Additional note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sensorimotor</td>
<td>Pre language skills</td>
<td>1 ½ to 6 -7 yrs.</td>
<td></td>
</tr>
<tr>
<td>Preoperatory</td>
<td>Preoperatory</td>
<td>1 ½ to 6 -7 yrs.</td>
<td></td>
</tr>
<tr>
<td>Concrete Operations</td>
<td>Concrete Operations</td>
<td>7-8 to 11- 12 yrs.</td>
<td>Abstract reasoning starts at about 11 yrs. of age</td>
</tr>
<tr>
<td>Formal Operations</td>
<td>Formally</td>
<td>11-12 to 14 -15 yrs.</td>
<td></td>
</tr>
</tbody>
</table>
when a child encounters unfamiliar material, they may be unable to make sense of it, become “stuck” and simply experience frustration rather than discovery.

Seymour Papert, known for constructionism, also provides insight into how children learn from their environment, especially from sources other than the formal classroom. He was influenced by Piaget by using children’s as builders of their own intellectual structures and coined the term called ‘Piagetian learning’ for the vast quantity of information children learn before they enter school (Papert 1993). With constructionism the thinking is that children will do best by finding for themselves the specific knowledge they need and what children need is knowledge to acquire more knowledge (Papert 1993). Papert does part with Piaget when it comes to the belief that some children intellectually develop slower than others and considered other issues as contributing to the differences in how children learn, such as the subject matter coming with a stigma which can affect learning, regardless of child’s development level. For example, Papert uses mathematics as an example as having cultural toxins associated with learning the subject, some students have come to believe they are not mathematical (Papert 1993). The subject of mathematics may have been one they enjoyed in elementary school but later in middle and high school some students have been influenced into thinking they are not able to handle the complexity of the subject material. No matter how much one memorizes or studies there are always those areas of certain subject that tend to constantly challenge some students.

### 2.3 Scaffolding

The concept of scaffolding in learning materials is often used to guide students through the complexity of computing technologies while still allowing a teacher, peer, or computer-based tool to assist the student as needed in the learning process. A student may not be conscious of how she is learning a subject and many times her ‘awareness’ occurs long after she has acted
upon a situation, not even realizing she is learning in the process (Piaget 1975). It is important for a teacher to know when to step in to help a student while she is in this unaware learning mode. Understanding what can be achieved by the student and knowing how to help can be challenging.

Vygotsky offers a complementary view to Piaget’s theoretical position. He described a framework known as the Zone of Proximal Development (ZPD) – the distance between a child’s actual development level as determined by what a child can do today with assistance and what they will be able to do tomorrow by themselves (Vygotsky 1978). A child’s learning begins when they are born and as they develop they are in a learning mode of developing new skills while they are still developing themselves. For learning to be effective or ‘good learning,’ it should occur at a level of development that is just in advance of a child’s current level of development (Vygotsky 1978).

Learning activities can be organized in a way such that the learner is provided the appropriate amount of information to complete tasks without adding additional complexity. One approach to this is the minimalist model of instruction; in this framework learning activities are designed to be realistic and involve the learner as an active participant in their learning, so that they will be able to recognize and make inferences when needed information is missing (Carroll and van der Meij 1998). When a learner becomes an active participant of their learning the task they work on becomes important and depending on the stage of their development may need to have an extra layer of support when working with computing technologies.

In a complex skill-based problem domain, scaffolded examples can provide an opportunity for learners to gradually encounter a set of concepts and skills. Because novices can become overwhelmed when faced with a blank screen when using a computing technology for the first time, the availability of working examples offer an opportunity to explore and build confidence as new skills are acquired. In prior work, we have shown that high school girls can benefit from several types of scaffolding – including examples – to assist in the learning of
databases and webpages (Rosson, Ioujanina et al. 2009). Not only is it important to provide the example for the students, but also the content provided in the exercise can be successful to helping those students who are between the stages of concrete and abstract development stages.

Children are constantly gathering information from the people they encounter in their environment. They use the materials they have obtained in the form of communications or actions to gain access to new information either formally or informally. The children can imitate others in collective activities and using help from adults, however in order for the children to learn from those imitations they need to be at the appropriate development level for the activity (Vygotsky 1978). Learning occurs when children interact with people in their environment and in cooperation with their peers. In my dissertation research, I leverage this by organizing the learning activities into pairs of girls working together.

When students are able to use materials within their learning environments to help make connections to the new material they can enjoy learning the new information. Providing the appropriate level of learning in a classroom can be done in a variety of ways, through the content which is being presented in an activity, the structure of the activity itself and the assistance provided to the learner during the learning process. By placing scaffolded examples in a peer-collaboration setting, I hope to leverage the ZPD concept in two ways: the examples are used to convey more advanced skills and concepts than the student can enact at the present; and the peer interaction may help them to co-construct knowledge that neither could build alone. I also hope to offer activities which present learning material with just enough information to provide a challenge to a student without deterring their desire to learn new computing technologies.

The approach of using multiple supports, also called distributed scaffolding, provides a classroom environment that encompasses the entire learning experience for a student (Tabak 2004). Support can be used to help students learn what needs to be done and also assist them in completing a task. When students work in pairs a social scaffold can form between the two that
not only assists each of them in the learning of new skills but also in conjunction with a teacher provides an additional layer of support in their learning environment. When distributed scaffolds work to augment each other, this synergy forms what is known as synergistic scaffolds that provide the guidance needed to complete a task by interacting and working together (Tabak 2004). A workbook to assist a student in working through multiple tasks can serve as a procedural scaffold and when paired with an actual example provides a conceptual scaffold to help convey concepts covered in the example and workbook. The challenges that are faced by students in a classroom need scaffolds that work together to provide support in learning new skills, constructing solutions to problems and also interacting with others in a team environment. My research draws upon the use of distributed scaffolds, using social, procedural and conceptual scaffolds working together to create a learning environment to support the learner in acquiring new skill sets to problems that can be challenging in computing activities.

2.4 Computing Attitudes of Middle School Girls

When girls are not comfortable in a learning context, their attitudes of inadequacy can transfer to their perceptions of the subject matter, especially if it involves computing technologies. This becomes a particular concern for young women at the middle school age, as many important changes are occurring as girls go from elementary to middle school. They find new course subjects in their class schedules and a greater variety of teachers who will serve as role models. Girls begin to notice which friends, family and teachers who enjoy working with and learning about new technologies versus those who avoid such activities. Girls at this age are very concerned about feeling comfortable in their new environment; they often choose to participate in activities that not only are feminine but also meet the approval of their peers (Barker, McDowell
et al. 2009). It is important to a young woman entering the new learning environment to fit in and be successful.

Researchers have observed that when students see their peers achieve similar performance goals in the classroom, they too are motivated and encouraged to handle the challenge. In fact it is better for the student to see a peer perform a task than a teacher (Schunk 1991). Working in isolation may not be the best motivator for a young girl who is working on fitting in and also trying to build her confidence in her new learning environment. For some young girls, self-confidence becomes a personal barrier that inhibits them from seeing the computer as nothing more than a tool they are unable to conquer or master (Denner, Werner et al. 2005). In contrast, working in small groups or pairs has been successful in creating an environment which is positive and motivating for young women. For projects in entry level programming courses, pair programming has been shown to increase the confidence level and enjoyment of all students in the class with their assignments (McDowell, Werner et al. 2006). At the same time, additional barriers can emerge if girls are paired with boys who are exhibiting more confidence in using the technology and come off as 'experts'. When middle school aged girls were polled in a 1992 AAUW survey they were more likely to say they were ‘not smart enough’ or ‘not good enough’ (Margolis and Fisher 2003). Young girls enjoy working with computers but they often expect there to be a purpose or task goal that motivates the use of computing technology.

To create an environment that is motivating to girls, educators have explored the role of stories and games. For example, research conducted with Storytelling Alice found that middle school girls not only enjoyed “programming” a story that was relevant to them but they also enjoyed turning it into a movie (Kelleher, Pausch et al. 2007). By presenting programming as a “means to an end to storytelling” (Kelleher and Pausch 2006), Kelleher found that girls can be motivated to learn programming concepts and find a purpose in the work they accomplish. The
“Girls creating Games” program also found girls became confident in becoming producers of computing artifacts and not just users of the technology (Denner, Werner et al. 2005). It is important that girls are provided learning activities in computing environments which build their self-confidence and provide opportunities for a positive experience. I hope to draw from the previous programs and learning activities using stories and games to build upon them with CT concepts. My goal is to create motivating learning activities that will build confidence and encourage girls to continue as producers of computing artifacts.

### 2.5 Tools and Software

K-12 educators are no longer asking the question "Should I use technology to enhance or deliver my lessons?" Instead they ask "Which technology or application should I consider using?" Unfortunately, with the constant improvements and advancements in computing and information technology staying current and up-to-date can be challenging for educators, so even if they are asking which rather than whether, the necessary expertise may simply be unavailable. Teachers can become overwhelmed by the new computing tools available for the K-12 classroom; this problem is magnified if they perceive that their students are quite adept at staying current. In today’s rapidly evolving world of technology many students have integrated mobile devices, the internet and social networks as a central component of their daily lives. In the past, technology tools and applications were seen as having more of a secondary role in learning (Bransford, Brophy et al. 2000). Now, students as young as elementary school are using social media and are very aware of the many technologies around them.

Many of the tools and applications found in today’s classrooms began with what have been termed constructional design kits (Resnick 1996); they had the express goal of aligning the students content activities to knowledge acquisition, with the students being active participants in
their learning. For example, students of all ages designing a model with a Lego programmable brick can provide a representation of a concept they have learned, such as a park exhibit which activates when a light sensor is triggered. Being a participant in their learning can be motivating and engaging for students of all ages. These early construction-centered environments were not only fun and social; they provided learning communities. Children as young as 8 years old could learn reading and writing skills and basic computer programming concepts in these early constructivist learning environments (Lehrer, Lee et al. 1999; Bruckman, Biggers et al. 2009).

Computing tools for the classroom have continued to evolve and understanding more about what is currently available can help in the selection process of what to use when developing a computer-related activity.

As part of my preparation for this proposal, I reviewed a broad range of current technology used in K-12 settings for educational purposes. My analysis suggested four conceptual categories that help to convey which computing tools may be more appropriate for certain types of activities (Webb 2012). Table 5 contains a summary of these categories and their respective descriptions, which are not intended to be inclusive or exclusive. The CT concepts and dispositions discussed in section 2.1 can be found in each of the conceptual categories and provide an additional framework when working with the computing tools.

<table>
<thead>
<tr>
<th>Table 5 - Conceptual Category Descriptions (Webb 2012)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social Networking</td>
</tr>
</tbody>
</table>

There are many interesting computing tools available that can be mapped to CT concepts and hence provide learning opportunities. For example, the Scratch 2D and Alice 3D programming environment (Figure 1, p. 32 and Figure 2, p. 33) both provide users the opportunity to work with simple data types, but also the list data structure. Not only can students work basic list processing of insertion and deletion, searching and sorting can also be done. Since both of these computing tools are programming environments there are opportunities to create learning activities using expressions, control structures and other programming constructs. Lego Mindstorms programmable bricks are still popular to introduce students to the programming concepts of input and output. Activities involving light or touch sensors gather input for programs to generate output needed for motors to move a model.

With the variety of computing tools available another consideration when using computing tools in K-12 classrooms is the technology infrastructure available to educators in school settings. With internet access becoming more pervasive in public schools (Gray, Thomas et al. 2008), concerns related to computer operating systems used within school classrooms may not seem to be as significant as in past years. However because public school systems may not have the resources to track and update their infrastructure, many educators still find themselves stymied by platform requirements. Schools may also have restrictions placed on classroom computers in regards to sites students may access on the internet.
A large list of computer applications and tools that are updated regularly and often used in K-12 computing activities, as well as supported computer platforms has been included in Appendix A. Because I conducted my research in middle school classrooms I limited my options to current technology possible in public school classrooms; for example, I used tools that did not require the school systems to obtain site licenses which may not have been in their budget but still offered features needed for my research. Table 6 provides information on a list of the computing technologies I considered for my research that would work with the existing technology in the school classrooms and mapped them to the CT concepts.
Table 6 - Computing Tools under Consideration

<table>
<thead>
<tr>
<th>Computer Tools</th>
<th>Supported OS</th>
<th>Cost</th>
<th>CT Concepts Highlights</th>
<th>Supported Computing Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PC</td>
<td>MAC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alice 2.0, 2.2,</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
<td>Stories, Games, Movies</td>
</tr>
<tr>
<td>StoryTelling Alice</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scratch</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Stories, Games, Movies</td>
</tr>
<tr>
<td>Agent Sheets</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Stories, Games, Simulations</td>
</tr>
<tr>
<td>Kudo</td>
<td>Y</td>
<td>N</td>
<td>N</td>
<td>Games</td>
</tr>
<tr>
<td>Lego Mindstorm</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Models, Simulations</td>
</tr>
<tr>
<td>RCX</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lego Mindstorm</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Models, Simulations</td>
</tr>
<tr>
<td>NXT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lego WeDo</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Models, Simulations, Games</td>
</tr>
</tbody>
</table>

In the analysis conducted on the computing tools it was important not only to consider the tool based on its cost and compatibility with existing classroom technology but also to consider how versatile the tool is when it comes to the building CT concepts in computing activities. When computing tools support story and game activities, such as Alice and Scratch, CT concepts can be introduced in an engaging and motivating learning environment. With the sequential nature of stories, basic programming concepts can be kept introduced gradually leading up to more complex abstract concepts which go with games. I used the computing tools which best leveraged the computing activities and CT concepts addressed in my research. In addition I worked with the local school system to use their Lego Mindstorm systems for some of the more complex computing activities involving CT concepts.
2.6 Approaches to Teaching Computational Concepts and Skills

As discussed in Section 2.1, CT skills are applicable to a variety of subjects and Table 6 provides a list of the computing tools which mapped best to the CT concepts and dispositions from Table 2 (p. 17). Though not all the computing tools in this category are free of charge for educators, they offer a diverse set of approaches to learning and are appropriate for all ages. Of course, use of any of these tools presupposes expertise on the part of the educator, but gaining such expertise is often left to the individual who will be leading a workshop or class activity. This can be a problem when the planned learning activities have time constraints and/or the staff is less experienced than the participants (Webb 2012). Not all of the computing tools described above were intended for elementary or middle school aged children, though many have been recruited for summer camps and afterschool programs. The question then becomes, "Which is the best to use if I want to teach computing concepts of a particular sort?" Given the current variety of applications available, there is often more than one answer.

2.6.1 The Context of Learning

Summer camps and after-school programs have been popular mechanisms for providing new experiences with computer-based concepts and skills for middle and high school boys and girls (Clegg, Gardner et al. 2006; Adams 2007; Kelleher, Pausch et al. 2007; Nugent, Barker et al. 2009; Repenning, Webb et al. 2010). These options are attractive to educators and researchers they offer them opportunities to conduct research on new computing tools, attitudes towards technologies and methodologies in informal learning environments. For example, students in K-12 can be introduced to computer programming without worrying about homework assignments, grades and tests.
Summer camps and afterschool programs each use different approaches to introduce new technology concepts. Many summer camps have created programs which engage and motivate their participants by providing a variety of activities using different computing technologies (Fox, Newell et al. 2004; Doerschuk, Liu et al. 2009), while others use a single technology, such as Alice 2.0/2.2 or StoryTelling Alice (Adams 2007; Werner, Denner et al. 2009). Robotics has also found its way into many programs either as the focus for a week-long summer camp or as a separate component of a more general experience (Goldman, Eguchi et al. 2004; Cannon, Panciera et al. 2007). A popular format for afterschool programs has been as a “club”. The Kitchen Science Institute is one such informal environment where science was presented to students through everyday experiences (Clegg, Gardner et al. 2006) or the Computer Clubhouse which provided access to technology for students who may have had limited opportunities to use computing technologies.

My research study was conducted as both an after-school program and summer camp within the local school district’s outreach program. While other program had operated as “clubs” I used the setting of a workshop series as individual units of learning modules, within the local middle schools for a given timeframe, to cover specific CT concepts in both after-school and summer camp settings.

2.6.2 Stories

Traditionally, stories have often been used to teach young children lessons on morality, safety skills and other important life lessons. These stories are typically “told” by adults, teachers, or older children who want to pass on one or more lessons. In contrast, children in today’s classrooms are now often being encouraged to create their own stories. In research conducted with StoryTelling Alice, storytelling was a successful motivator to learning programming
concepts (Kelleher, Pausch et al. 2007). Stories encourage children to brainstorm ideas and then transfer those ideas to a working solution. Since stories are sequential in nature they can be used as foundations for basic CT concepts which can then be built towards additional complexity. Stories are sequential in nature which can be built upon with more complex concepts.

Figure 1 - Alice Programming Environment

With respect to digital storytelling, both Scratch and Alice (Figure 1) provide considerable support for creating characters, backgrounds, and character behaviors in service of a story line. Scratch supports media manipulation and has a fixed background (Maloney, Peppler et al. 2008) and students can share their projects on the community website. While the pre-defined gallery of objects in Scratch is not as extensive as the one found in Alice, it provides a pixel editor that can be used to import images or create original artwork (Figure 2), a feature not included in the Alice
computing tools. With Alice 2.2 students can now export their stories to video which makes them easier to share with friends. Storytelling Alice was created when it was found that girls were more receptive to learning how to program when they could build a project that was based on a story they wanted to tell (Kelleher, Pausch et al. 2007). The components in Alice are centered around common 3D objects and in Storytelling Alice they are centered around stories (Kelleher, Pausch et al. 2007). Storytelling Alice with its focus on story scenes, custom animations for characters and a tutorial based on a story provides a supportive learning environment for young women of middle school age.
I considered the current status of computing tools as I made my decisions on the tools and types of activities I used in my study. The first activity in the workshop series was involved an interactive story, which created a need for a tool that supported programming foundations for novices while still enabling me to create activities that would evolve in complexity as the confidence levels of the girls increased. It was also important to ensure the tools were developmentally stable and maintained by the developers. This led to decisions of not using obsolete tools or those still in beta development.

2.6.3 Games

One popular activity used to interest middle school children in attending summer camps or after-school programs is design and construction of computer games. Although computer games are often associated with boys more than girls, girls do enjoy some video games and many summer programs advertise for them to create a computer games. Studies show that boys and girls are attracted to different game genres (Jenson, Castell et al. 2007). For example girls are not likely to choose characters and activities that are combat ready and firing bullets; boys first choice are not puzzle games. Thus, when using games as a vehicle for teaching CT concepts and skills, the genre preferences of the learners should be carefully considered.

The design and construction of a computer game involves much more than knowing how to play a game. Not only do students need to know the general concepts of game play but they also need to understand general computational concepts as well as the authoring platform that is being used to create the games. For example, if building a game requires a score and a timer, the learner would need to have some understanding of concepts like incrementing and decrementing of counters, decision logic to determine if the winning score has been met in the required amount of time and which message (win or lose) to display to the user so the game ends.
Alice 2.0 was originally created as a programming environment for introductory college students; K-12 teachers have used this tool to help students create animations and simulations for school projects (Rodger, Hayes et al. 2009). These activities can also be converted into games (e.g., a simulation of a historical event can be played as a game if characters quiz the users and points can be earned for answering correctly and feedback, but no points for answering incorrectly). In courses held at San Antonio Community College, similar games were created by college aged students to teach the alphabet and even Spanish to younger children. Students as young as middle school age can also work with Alice 2.0, to create games while learning to use a Java-like programming language within a 3D simulation setting. For example, in a summer camp for middle school girls there were games created involving characters in a storyline where the user needed to answer questions to collect items which moved them to the next challenge (Webb and Rosson 2011).

The 3D simulation environment of Alice introduces a number of complexities related to viewing and animation in three dimensions (Figure 1). As an alternative, Scratch uses a 2D environment with a fixed stage background and interlocking code puzzle pieces, which may make it a better choice for younger students who struggle with the freedom of movement a character has in the 3D environment of Alice with drag and drop code tiles (Adams 2010). Scratch also includes a community website of work created by peers which can be downloaded and extended or modified as a starting activity, or to use as models for new ideas (Kafai, Fields et al. 2009). A typical game in Scratch might be of a fish game involving the consequences between good and bad food (Figure 3), that would rely on CT concepts such as problem solving and abstractions in getting the two counters to register correctly and determining an abstraction which will work in an animation for the eating of food by fish.
A benefit associated with both Alice and Scratch is that learners can test their games as they drag and drop their objects and program code, without seeing complex compiler errors. Even though they provide no specialized game engine (e.g., like GameMaker or Kodu), Alice and Scratch have been successfully used in summer camps and afterschool programs for students to create their own games to play (Adams 2007; Werner, Denner et al. 2009). Of course these games tend to be incomplete and informal, far from commercial game quality, but nonetheless children in these camps present their games with pride to parents and peers during showcases. When examined in detail, researchers were able to identify a number of familiar computational elements, rules, goals and interactive behavior (Werner, Denner et al. 2009).

By using a game as a workshop activity I could introduce more complex CT concepts such as abstraction and also build on the CT concepts introduced in the activities involving stories while still providing an engaging and motivating learning environment. The scaffolded examples
for the computing activities based on a game were carefully constructed to place emphasis on the complexity of the CT concepts. This was needed since the game play of games can be a distraction.

2.6.4 Simulations, Visualizations and Modeling

When children are able to create visualizations to demonstrate concepts in science and mathematics it not only helps them learn the content of the subject it also engages them in the technology while they are learning (Repenning and Ioannidou 2004; Rieksts and Blank 2008). Simulations and models are also useful in helping younger children in grasping complex problems. By working with sound, images, and other elements students can have fun and be creative as they add their personality to their projects at the same time they are learning new school subjects.

One computer application that is designed to support simulations and modeling is Lego Robotics. Projects using this tool can be both diverse and complex. For example a simple model might involve a car which is activated when a light sensor is activated by a beam of light, but a more complex model might be creating a proximity sensor which activates an alarm system. In each example understanding of programming is needed as well as input and output; however the more complex exercise would also involve understanding variables, decision logic and also interacting with another programmable device via Bluetooth technology. Another interesting feature of this platform is that robot projects give students the opportunity to work with both the hardware and software to create visual models which are interactive. The hands-on interaction with physical objects like the sensors or the motors along with the Lego bricks themselves may be particularly engaging to children, who often have an experience of building a variety of structures to support their play at home or with friends. The digital environment of The Sims offers similar
engagement, with considerable attention spent on organizing and managing “pretend” homes, cities, and other structures.

The AgentSheets environment has been used for a number of years to enable young students to create games, stories and simulations aimed at exploration of science and mathematics concepts. It is one of the few applications which has research work related to computational thinking (Repenning, Webb et al. 2010). Students can research information about a problem and then create an interactive visualization to illustrate their understanding of the information gathered. A key aspect of this tool is the underlying grid framework that is used for placing objects (agents) that can detect the presence of other objects, behave accordingly, and so on. As they learn AgentSheets programming, the students are not only creating simulations but also bringing an element of creativity to their work. For example, children enjoy changing backgrounds, changing fonts and creating their own sprites as is possible in the Scratch environment. AgentSheets operates in a 2D visual environment, simplifying the viewing and manipulation of visual objects and activities.

Models and simulations can provide student’s opportunities to explore more than one computing technology in a single activity; however these additions also come with increased complexity in problem solving concepts. It was important the computing activity to construct a model was engaging for those children who had little to no experience with the technology. By focusing on the CT concepts it was possible to develop an activity to create interactive structures with Lego robotic sensors and motors to provide a positive learning environment.

2.7 Impacts of Computation-Related Education Programs

To date there has been no longitudinal study of students who have attended computer summer camps and then gone on to completing Computer Science degrees. Instead data
collection has focused on student’s self-perceived interest in computer science topics, and what they predict they may do in the future. As an example, one project that sponsored a summer camp for middle school students using Alice reported that 79% of the students reported that they were inspired to continue learning computer science in high school and 3.5% were interested in considering a career in computer technology (Hardnett 2008). Though the percentage considering a career in CIS is low, it is still important that middle school students are beginning to think about their futures and courses they take in high school will impact choices they make for college.

There are many reports of success in using the Alice programming environment to increase student understanding and engagement at the college and high school level (Cooper, Dann et al. 2003; Edwards, Gersting et al. 2007; Powers, Ecott et al. 2007).

One outreach program used Storytelling Alice with middle school students to determine if it could be used for game programming; the researchers found that students using it were engaged in the CT skills of algorithmic thinking, programming, modeling and abstraction (Werner, Denner et al. 2009). In this program, the researchers found that even when the students were not prompted to focus on computational concepts, they incorporated them into their computing activities. For example, understanding of modeling and abstraction were exhibited in the development of additional methods which required new variables and parameters to be passed between methods. Algorithmic thinking and programming was demonstrated in the students projects with the use of ifthenelse logic, loops and parallelism using the do together condition (Werner, Denner et al. 2009). In addition the Computer Clubhouse is an ongoing program for introducing urban youth to computing activities in a regular after school activity center. Researchers studying this outreach program found that when project choices were left to the young participants, 53.6 % of them worked on projects that included user input and output, and 51% used loops. Both of these are programming concepts that are fundamental to game design (Maloney, Peppler et al. 2008).
The after school environment has been used by school districts and external organizations to set up supplemental science, engineering and computing programs (Nugent, Barker et al. 2009). The opportunities to try out new technologies may appear to be the same for all children at the middle school age level; however that is not always the case for all school districts or for all children. Students do find ways to use and learn new technologies in both formal and informal settings. With boys and girls still discovering what they may enjoy studying in the future, it is important to provide those opportunities. The research conducted on afterschool programs and summer camps has shown students can be motivated to try new computing technologies and have fun with the experience. My research on computing tools and related CT integrated computer technology activities that engage middle school girls in an afterschool environment adds to the research conducted by these earlier computer-related education programs.

2.8 Summary and Discussion

As discussed in this chapter, K-12 curricula are incorporating CT concepts and skills in a variety of courses. Computing activities with problem solving, abstraction and algorithmic thinking provide opportunities to use CT concepts and skills in both school classrooms and extra-curricular enrichment programs. Most children at the middle school age level have the intellectual development and learning capacities to work on concrete and abstract problems that are often found in computing activities (Table 4, p. 20). In addition, from Vgotsky’s ZPD framework we know that learning can be effective for children if it is just in advanced to what the child currently understands. Scaffolding provides a means to guide learners through learning concepts that are just in advanced of their current level of understanding, especially with activities where the computing environment adds additional abstractions.
Many technology related afterschool programs and summer camps originated as outreach programs, researchers have found that students have been motivated to learn more than what is required of them when they work on computing activities in these programs. Students may not have been explicitly taught to use loops, decision logic or other forms of algorithmic thinking and abstraction, yet examples of such concepts can be found in many of the students’ projects. That is, we know students are using CT concepts in computing activities but do they understand what they are doing? Are there additional CT concepts they have not considered and comprehensible for the middle school age level student?

An exercise involving problem solving can vary in complexity, when a computer is added to the problem and solution panic can occur in novice users of computing technology. To engage the middle school participants in my study I used computing activities patterned after stories, games and simulations using scaffolded examples. It is important the computing tools selected for each of the activities did not increase the complexity of the computing activity and subsequently overshadow the CT concepts being studied. By providing computing activities which built on each other in complexity of CT concepts and skills I wanted to determine if attitudes towards computing increased and what motivates them to learn more about the role of computing in their future.
3  Research Overview

My research objectives are to examine the nature of CT concepts and skills that middle school girls can comprehend and apply; to design and evaluate a set of scaffolded computer-based activities that can convey such constructs; and to investigate the consequences that such activities have on girls’ more general attitudes and feelings of computer self-efficacy. I pursued these objectives by designing and conducting two after-school workshop series and a four-day summer camp. In all of these cases I was able to leverage existing school district programs aimed at learning enrichment for this age group, while directing the advertising and recruiting specifically to my target population of middle school girls.

3.1  General Workshop Structure

Table 7 provides an overview of the workshop sessions held in Fall 2011, Spring 2012 and Summer 2012. Early in the planning process, I coined the name “GirlsCreateIT!” as a name to use in advertising and recruitment. I continued to use this name in all the materials and communications, so in this document I will use the convenient acronym “GCI” to refer to each workshop series, with a number to indicate its position within the sequence.

Within a GCI instance, each session consists of an activity that provides hands-on experience with a particular computer tool or programming language, as well as a learning activity that has been designed to convey CT concepts and skills. Thus the goal of each session was to enable the middle school girls to complete an activity that implicitly required the application of CT concepts; if successful the result would be a computer artifact built with a computing tool selected specifically to support the activity. To support this goal, the workshop
activities were centered on scaffolded examples (learning examples built in the tool to be used that day, and illustrating the CT concepts of interest) that are explored under the guidance of activity workbooks; the workbook structure offers students with less experience a gentler introduction to the technology, including introduction of key CT concepts.

Table 7 - Research Study Overview

<table>
<thead>
<tr>
<th></th>
<th>GCI-1 (Fall 2011)</th>
<th>GCI-2 (Spring 2012)</th>
<th>GCI-3 (Summer 2012)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time frame</td>
<td>6-week afterschool program</td>
<td>6-week afterschool program</td>
<td>4-day summer camp</td>
</tr>
<tr>
<td>Session length</td>
<td>90 minutes</td>
<td>90 minutes</td>
<td>90 minutes scaffolded example activity; 90 minutes on your own</td>
</tr>
<tr>
<td>Computing tools</td>
<td>Alice 2.2, Scratch, GirlsCreateIT! DB, Lego RCX</td>
<td>Scratch, Lego WeDo</td>
<td>Scratch, Lego WeDo</td>
</tr>
<tr>
<td>Scaffolded examples</td>
<td>Story, game, web page, sensors and motors</td>
<td>Story, game, lists, sensors and motors</td>
<td>Story, game, lists, sensors and motors</td>
</tr>
<tr>
<td>Supporting materials</td>
<td>Workbook with journal probes</td>
<td>Workbook with end of day questions</td>
<td>Workbook (no questions)</td>
</tr>
<tr>
<td>Data collection</td>
<td>Workbook comments, computing artifacts, classroom observations, pre/post surveys</td>
<td>Workbook comments, computing artifacts, classroom observations, pre/post surveys</td>
<td>Workbook comments, computing artifacts, classroom observations, interviews, pre/post surveys</td>
</tr>
</tbody>
</table>

GCI-1 and GCI-2 were conducted in an afterschool setting. Each session lasted of 90 minutes, during which I was able to focus on specific CT concepts, including practice on scaffolded example that I hoped would help built CT confidence in these young students. The summer camp also allowed me to focus on specific concepts (e.g., in the first half of each day). However the camp was structured as four morning sessions of three hours each, rather than an add-on to an already busy school day. This longer time period (i.e., the extra 90 minutes) meant I could offer an extended on your time session each day. I used this to further engage the participants in exploring the technology and using the newly introduced CT concepts. The week-
by-week format of GCI-1 and GCI-2 allowed me to build on concepts covered in previous sessions, so that I could introduce concepts with increasing complexity. This was even more effective in the condensed day-by-day format of the summer camp.

Other major changes that took place over the three research studies involved the number of computing tools that the girls were exposed to, the specific content of the scaffolded examples, the nature of the workbook used as guidance and them data collection methods. Each of these is described in the following sections.

### 3.2 Refinements to the GCI Concept

As an exploratory research project, I began my work with many questions about how to implement the research concepts in focus (i.e., the CT concepts and skills) as well as how to assess impacts of the workshop activities. As a result several important changes occurred during the three studies.

#### 3.2.1 Computing Tools

I began my project with the goal of introducing the girls to a variety of computing platforms (i.e., as previewed in Chapter 2), with the general design goal of matching the tool to the type of activity being used. For instance, given the success of Kelleher and others in using Alice for story projects with girls of this age, I elected to start with an Alice-based story editing project. The work by Resnick and others using Scratch motivated me to include this tool with a project involving interactive games. Our own research lab’s success with young women’s end-user development of simple data-centric web sites encouraged me to include a data-focused activity of that sort; development of a custom database query tool to support such an activity for
younger girls was an additional incentive. Finally, the general enthusiasm by some girls of this age for hands-on projects that involve “building” things like robots and crickets motivated me to include a physical construction activity that used a specialized programming system to set up and interact with sensors and motors. All of these were built and piloted in GCI-1.

However, as detailed in the first half of Chapter 4, the richness of the resulting workshop was simply overwhelming to the girls. They did have some success with each activity but needed considerable coaching and assistance. Our classroom observations, their workbook comments and their post-survey reports suggested that their natural interests and cognitive skills were well suited to Scratch; thus the two subsequent GCI series focused more narrowly on Scratch as a programming platform, thereby removing the need to learn languages, tool user interfaces and implementation details for several different platforms.

3.2.2 Scaffolded Examples

All of the activities are grounded in a scaffolded example – a working artifact designed to illustrate a learning activity’s target concepts via exploration and subsequent extension or modification (Rosson and Carroll 1996). For instance the first activity for each GCI instance presents what I refer to informally as a “wacky story” - a story that plays out in a way that does not make sense, so that the learners must encounter and practice CT concepts such as analyzing the problematic story features, followed by the design and algorithmic thinking to produce solutions to fix the storyline. The second activity involves a simple interactive game that is “broken” in simple ways that can be discovered, analyzed and corrected through application of CT concepts.

An example of the first scaffolded example used in GCI-1 appears in Figure 4. It shows the starting screen of a story about a vet who encounters a mother wolf, but who behaves in an
inappropriate way (she does not show any fear or emotion). The girls figure out the problem, then brainstorm and build solutions to that problem (e.g., they make her show some anxiety, walk away from the wolf, etc.). 12 different scaffolded examples were created over the course of the three workshops, including four isomorphic examples within each workshop instance. A complete listing of the scaffolded examples can be found in Appendix B.

Figure 4 - GCI-1 Scaffolded Example in Alice 2.2

I designed the scaffolded example activities to be experienced as a sequential series intended for the same population of middle school girls (i.e., in all three cases, participants signed up for the entire workshop series); this allowed me to reinforce CT concepts encountered earlier in the workshop while also expanding to include more detailed concepts. To enhance the coherence of the participants’ learning, I used a common theme to integrate the scaffolded examples (for example, creating a museum display for an animal park, creating environment awareness projects or vacation plans). However, across the three workshop series, I used same general type of scaffolded example to illustrate similar CT skills and concepts (see Table 8).
Indeed, a number of the same general learning objectives were illustrated in all of the examples (e.g., problem solving (analysis and design), algorithmic thinking, and use of appropriate vocabulary). As the workshop progressed however, more complex projects were initiated and these tended to expand the range of CT knowledge the examples could illustrate.

Table 8 – Scaffolded Examples with Associated CT Concepts and Skills

<table>
<thead>
<tr>
<th>Activity used to convey CT concepts</th>
<th>CT Concepts and Skills</th>
</tr>
</thead>
</table>
| A story in which characters interact but in an incomplete fashion; participants fill in the story (GCI-1, GCI-2, GCI-3) | • Design and algorithmic thinking  
• Use the vocabulary  
• Recognize abstractions and move between levels of abstractions |
| A simple interactive game that is functional but that can be extended to illustrate new concepts (GCI-1, GCI-2, GCI-3) | • Design and algorithmic thinking  
• Use the vocabulary  
• Recognize abstractions and move between levels of abstractions  
• Test and Debug |
| A pre-existing online database that can be viewed, queried, and extended, with query results displayed as part of a web page. (GCI-1 only) | • Design and algorithmic thinking  
• Use the vocabulary  
• Recognize abstractions and move between levels of abstractions  
• Test and Debug |
| An interactive language translation application using lists of data; participants added error handling, created new lists and modified existing lists to complete the project. (GCI-2 and GCI-3 only) | • Design and algorithmic thinking  
• Use the vocabulary  
• Recognize abstractions and move between levels of abstractions  
• Test and Debug |
| Build an interactive structure with Lego bricks and art supplies to use the Lego sensors and bricks. Program adjustments were made by the participants for their structures and downloaded to the external RCX brick. (GCI-1, GCI-2, GCI-3) | • Design and algorithmic thinking  
• Recognize abstractions and move between levels of abstractions  
• Use innovation and creativity with technology  
• Test and Debug |

Note that each scaffolded example activity also provided opportunities for the participants to add embellishments or creative extensions to their projects, such as adding
additional scenery, modifying existing objects by changing color or size or even importing new objects.

As the scaffolded examples were being developed I realized the workshop participants would need additional support to explore and understand the example code. As a result I prepared workbooks to guide the learners through a number of problem solving and algorithmic thinking to work on the subtasks. In a long-term study of CT learning, these extra supports (and indeed the scaffolded examples themselves) would be gradually “faded” from the learning context, so that the learners take control and responsibility for their own learning (Puntambekar and Hübscher 2005). In my work, however, each week introduced new computer concepts and (for GCI-1) related computer tools; thus it was more important to support learners with the guidance needed to succeed at the programming tasks. As a result, fading of the scaffolding was not part of my research design as it would be for a longitudinal (e.g., class-based) intervention. In practice, the supporting workbooks played a secondary role as reference material during any free time activities (e.g., “on your own” time after an activity had been completed). After completing a scaffolded example activity, participants had one or more artifacts they had modified or newly created, along with any notes they may have made in their workbooks. These products cumulated as part of a “showcase” session scheduled at the end of each workshop.

3.2.3 Activity Workbooks

A scaffolded example provides information implicitly to students, but we also wanted to offer guidance aimed at students who are between the stages of concrete and abstract development stages. They may need additional scaffolding to grasp key concepts like abstraction or reasoning about alternate solutions. For this reason, I decided to create activity-specific
workbooks that could guide the girls through the activities and help them to learn more about the environment through exploration.

The workbook that I gave the learners at the start of each session was designed to act simultaneously as a secondary piece of scaffolding (i.e., guided exploration) and as a collector for learning reflections. Each workbook refers to the scaffolded example that will be explored, as well as the programming tool the girls will use to produce the computing artifact for that week. Examples of workbook pages can be found in Appendix E. The workbooks were designed as “minimal manuals”, an instructional approach that offers goal-evoking questions and prompts to help learners activate and think about relevant knowledge, but expects them to formulate and enact the specific action plans that are needed (Carroll 1990). Learning materials that provide opportunities and support for users to actively explore computing tools helps them to engage more actively in the learning domain and build richer understandings (Carroll, Mack et al. 1985).

Figure 5 illustrates one page from a workbook designed for the GCI-1 workshop, where in the third week the girls worked on a Scratch-based interactive game called OceanFun. The game presents fish (who are hungry) and eating options that include both good food (healthful and eventually game winning) and bad food (poisonous and eventually game losing). At the game start, the arrow keys are not working correctly and there is no mechanism to keep track of fish health (i.e., as a result of eating good or bad food). Thus the workbook page for this game project guides the girls through an analysis activity aimed at testing the current game to see what is working and what is not working. This provides a launch pad for them to use algorithmic thinking to plan and implement possible “fixes” or solutions.
The example in Figure 5 also shows the use of the workbooks as a data collection device; the intention was that the girls would write down their ideas and that I would be able to analyze these later to assess their success or understanding with these more structured activities. I also included more open-ended questions to probe their understanding of specific concepts as well as more general reflections about CT during the workshop. Example pages illustrating these workbook features can be found in Appendix E. Unfortunately, as reported in Chapter 4, I observed that the girls did not want to write down their ideas or answers to embedded questions. Despite persistent coaching efforts by me and my assistants, the typical workbook had many blank sections. Thus by GCI-3, I had given up using the workbooks for data collection and shifted to the more direct method of interviewing.
3.2.4 Data Collection Methods

Because I was working with children aged 10-14, I was concerned about gathering sufficient data to assess what the girls had grasped while working on the activities as well as the consequential impacts on their career attitudes and self-efficacy. Also, given that the initial setting for the work was an afterschool enrichment program in a middle school classroom, I knew it would be important for the girls to feel engaged in the activity and to have fun while they worked through the problem solving activities; I could not be stopping them at every point along the way to see what they were thinking and how they felt.

For these reasons I explored a range of data collection methods that could converge into an overall view of the entire research study. Table 9 lists the different forms of data collected throughout the research activity. Each activity was associated with a workbook that provided high-level guidance and important concept definitions; however the workbooks also included CT-related probes and reflection areas to document their experience as they worked throughout the activity sessions. The expectation was that these probes would produce individual reflections that would convey the girls’ current understands or misunderstandings of the CT concepts being explore. In the GCI-1 and GCI-2 after-school programs, I noted that the girls often failed to respond to the probes leading to somewhat sparse data about their understandings. As a result during the GCI-3series I used a more proactive method, interviewing each girl at the end of each workshop session.

Other data were collected to investigate the girls’ learning experiences and perceptions of the computer–based activities. Demographic information and initial measures of computing attitudes and career interests were obtained from pre-workshop surveys in all the sessions. After the final session of each workshop series, a post-workshop survey was used to assess possible changes in the girls’ attitudes and career interests.
With respect to real-time data, research assistants observed the girls during the workshops. These assistants primary role was to answer questions and encourage the girls in their work, but they also noted down informal classroom observations. Finally, project artifacts from each of the workshop sessions and any new project created during any of the sessions was archived for later analysis.

As the study progressed from the GCI-1 to GCI-3 series the measurements were evaluated and revised as necessary. For example, I shifted from a journaling approach using the workbooks during the fall and spring workshops, to a direct interview in the summer camp. As a result the workbooks evolved to be more of a guide for the students who needed the structure rather than a data collection method.

### 3.2.5 Data Analysis Methods

Partly because my data collection methods were evolving during the GCI series, the amount of data collected (e.g., across the workbooks and artifacts) was large and diverse. As a result, I typically used relatively informal methods to first make sense of a set of data on its own, and then determine how it related to the research study as a whole. For the case of the survey

<table>
<thead>
<tr>
<th>Measurements</th>
<th>GCI-1</th>
<th>GCI-2</th>
<th>GCI-3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workbook with journaling</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Workbook</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Classroom Observations</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Scaffolded Examples</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Pre Survey</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Post Survey</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>End-of-Day Questions</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Interviews</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
methods, much of the data was numerical in form and much simpler to analyze. In the following I comment briefly on my approach to the methods listed in Table 9.

*Workbooks.* To analyze the workbook entries all the responses to the questions that guided the scaffolded activities (GCI-1, GCI-2, GCI-3); conceptual questions for the CT concepts (GCI-1 only); and the questions relating to the general workshop (GCI-1 only) were placed in an MS Excel spreadsheet. By reviewing the comments in this fashion, I formed an impression of what activities were understood well and engaging and what activities were less clear. I was also able to determine that many questions had been left blank or given responses that were so vague that I could make little sense of what was being described. It was for these reasons that I gradually came to see the workbooks as scaffolding and introduce direct interviews as a vehicle for understanding what individual girls had grasped.

*Artifacts.* Upon first glance at the artifacts collected in each workshop it was apparent they would be a rich source of data. However, the artifacts collected during GCI-1 were not scrutinized as those intensely because they came from completely different computational tools and languages. Because GCI-2 and GCI-3 used a single common platform, it was more possible to assess the accumulating knowledge and skills that the girls were applying to their work.

My analysis of the Scratch artifacts occurred in three phases. First I produced a summary report of the Scratch program code (an undocumented feature of the environment that I discovered and proved to be quite important). Second, I placed information from these summaries (including number of sprites, backgrounds and programming constructs) in a spreadsheet. However this first analysis was at once too detailed and not well connected to my research questions. Thus in a third phase, I returned to each artifact to gather information by answering questions I developed that pertained to introductory programming concepts, listed in Table 10. These questions came from my initial review of the artifacts for interesting and distinctive features and also issues I had seen as an educator working with novice programming students.
Table 10 - Questions for analysis of Scratch Projects

<table>
<thead>
<tr>
<th>CT-related Probes Regarding the Scratch Programs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Did the team over-vision their solution implementations? (e.g., extra characters and/or sprites; extra or specially refined costumes)</td>
</tr>
<tr>
<td>Did they introduce code that was not needed?</td>
</tr>
<tr>
<td>What Scratch design and implementation concepts did the girls demonstrate in the stories or games? (variables, broadcasting, repetition, iteration)</td>
</tr>
<tr>
<td>Did they use hide/seek, broadcasting to change the state of their characters?</td>
</tr>
<tr>
<td>Did they explore instruction blocks not covered in the activity to expand the activity?</td>
</tr>
</tbody>
</table>

Pre and Post Surveys. The surveys contained open-ended questions that were not coded in any systematic fashion but rather were used to gather general demographic and background information and to also provide additional information to supplement my interpretation of the workbook entries, classroom observations and end of day questions. All of the rating scale used ratings from 1-7; these were recoded as necessary (e.g., negatively worded statements) and analyzed using SPSS. Simple tests for comparison of means were used to test for the hypothesized increases in career or self-efficacy positivity from pre- to post-workshop.

End-of-day questions. This data collection method was used in GCI-1, but initial review revealed largely blank or vague entries by the girls. I discovered this after entering these responses into a spreadsheet. This persuaded me that I should conduct interviews because during that review I found myself asking “Why?” and “Can you explain how you did this?” when I read their very brief (if at all) responses.

Interviews. I conducted interviews at the end of each session in GCI-3, generally in the time between their scaffolded example activity and the time set aside for on your own activity. I transcribed each interview and placed them in the qualitative data analysis tool atlas.ti to code for use of computing vocabulary, general concepts, and specific Scratch programming constructs. During the interviews, I took care not to mention specific CT concepts or vocabulary recently
encountered in the learning activity. Thus the girls’ responses provided one indication whether and how well they had grasped the target CT concepts, for example if they were able to appropriately use the computing vocabulary that had just been introduced in the workshop activities. This coding also helped me to gather and document the more general reflections that had been so elusive with other methods.

*Classroom Observations:* In GCI-1 and GCI-2 my assistants and I informally collected information on what the girls were doing during the workshop activities, comments the girls made while working on their computing activities, and overall impressions on the workshop flow for that day. After each workshop series these observations were informally analyzed for completeness and to assist with the modifications of the next workshop series. In GCI-2 these observations were collected only by me and a single assistant; my informal assessment of the other assistant indicated that he was doing too much editing of what he considered important. As a result he filtered out too much relevant information. Because these observations were not structured in any fashion, my interpretation involved an admittedly subjective extrapolation of the comments and anecdotes we recorded; this helped to build a rich picture along with workbook entries and corresponding artifacts.

### 3.2.6 Research Participants

Participants for the GCI-1, GCI-2 and GCI-3 workshop series were middle school girls in the age range of 11 to 13 years, recruited from middle schools in the State College area. Each workshop, including the GCI-3 session, was designed to accommodate a maximum enrollment of 16 girls; this cap was to ensure that all participants had an opportunity to receive coaching if needed. With the increase of home computers and use of computers in the school classroom it is expected that the girls would be familiar with computing technology and in fact may consider
themselves to be relative experts in many computer-based activities (e.g., social media, sharing of music, photos and videos). I also recognized that the region surrounding Pennsylvania State University is relatively rich with opportunities for extra-curricular activities such as summer camps; indeed in other projects my research group has initiated such activities within this general context (Webb and Rosson 2011). However, given their age and the general middle school curriculum (PDE 2009) I did not expect participants to have received formal schooling in computer science programming topics.

Note that in all activities, the girls worked in pairs (sometimes a threesome), with each girl taking a turn to “drive” the computer that holds the computing tool (e.g., Alice (Figure 4, p. 46) or Scratch (Figure 3, p. 36)) and the scaffolded example. This was of benefit to them, in that they could collaborate in the problem solving, design and algorithmic thinking needed to produce a solution; it was also of benefit to us as observers as we could pick up on problems, confusions or surprising successes.

3.3 Research Study Synopsis

Figure 6 provides an overview of the three GCI workshops that I conducted to answer my three research questions. Because my research objectives were to examine the nature of CT concepts and skills it was important that I first conceptualize computational thinking concepts that middle school girls could comprehend which RQ1 addresses. Once I had articulated the concepts I wanted to explore with this age group, I created an intervention by designing and evaluating a set of scaffolded computer-based learning activities that included programming constructs to convey the CT concepts, thereby investigating the role of scaffolding (RQ2). Finally I investigated the self-perception consequences of these learning activities by tracking levels of self-efficacy and career attitudes (RQ3).
In order to carry out this research plan I followed an evolutionary path, iteratively developing and refining both the learning activities and the outcome measures (Figure 7).

Because I designed and developed all of the materials for the workshops (aside from a few existing self-perception scales drawn from the literature) it was necessary to gather formative data concerning the learning activities to ensure that the information I was interested in learning during the research would provide the answers I needed for my research questions. Although even the first workshop (GCI-1) provided quite positive indications of the girls’ ability to learn CT constructs, it was not until GCI-3 that the materials and procedures stabilized enough to conduct a more complete summative evaluation.
Figure 7 - Evolution and Refinement of Research Methods
4 Formative Research

An important element of my research plan was to conduct one or more preliminary studies to determine if the scaffolded example activities would be effective in illustrating the CT concepts, and to assess different approaches to tools and the supporting materials contained the contexts for CT learning in the workshops. It was also important to assess what is learned and how it affects attitudes. This formative work was accomplished via two after-school workshop series, GCI-1 (Fall 2011) and GCI-2 (Spring 2012).

Perhaps the most important goal of these initial studies was to design and evaluate a set of computing activities that illustrated and required the application of CT concepts, such as problem solving while also engaging and motivating girls of this age. Because I expected the girls to be between the ages of 11 to 13, I focused on the analysis problem solving and construction of “programming” activities that had been shown to be motivating to this population – storytelling, games, and working with robotic components (Kelleher and Pausch (2007)). In addition, activities on webpages and databases were included because earlier research had shown this to be motivating and accessible by slightly older girls (in 9th and 10th grades (Rosson, Ioujanina et al. 2009)). Thus at the highest level, I began with the goal of determining whether the activities and associated tools would be feasible as learning problems for girls at this age.

4.1 Fall 2011 (GCI-1): After-School GirlsCreateIT! Program

The GCI-1 was a six-week after-school program offered during Fall 2011. In this I experimented with several different computing technologies – Alice 2.0, Scratch, a custom web
development tool, and Lego® Mindstorms (Webb and Rosson 2012). A series of six workshops was designed, conducted and evaluated, for a period of about 90 minutes after school one day per week. The workshops were organized into activity units that comprised interactive stories, games, database, and interactive structures.

In this initial study it was important to determine if the 90 minutes allotted in an after-school setting was the correct amount of time to cover the CT concepts in the computing activity and still engage the girls. I did not know whether the initial selection of tools would be the right choice for this population of middle school girls over a six-week series, where the focus of the activity shifted each week. I was also hoping to learn more about the scaffolding mechanisms I had included. Was there enough or had I used too much? Another issue was how to evaluate what is learned and its impacts on attitudes. Would questions in the workbooks and the surveys provide the information needed? To do this I looked at computing tools used with this population and selected those which supported computing activities which provide an engaging and motivating learning environment. In the following sections, I provide more detail on what and how this first GCI workshop series was conducted.

4.1.1 Participants

The GCI-1 workshop series took place at Park Forest Middle School as part of the State College School District Outreach Program. The original agreement with the district was for me to offer one workshop per week for six weeks, with a maximum enrollment of 16; however due to high registration volume, a second (identical) set of workshops was delivered in parallel, resulting in a total of 30 participants. Though the program was advertised to girls at all grades in middle school, the majority of attendees (67%) were in 6th grade; across the 24 girls the average age was 11.7 years (Figures 8 and 9). Research consent was obtained from 24 of the participants across
the two groups and in general I have combined the data obtained in both sets of sessions, as there were no apparent differences in the girls’ experiences.

On the first day of the workshop series I did not know whether or how many of the girls were comfortable with each other or with the technology to be used in the computing activities. Previous studies have found that working in small groups or pairs can increase confidence and enjoyment in working on assignments (McDowell, Werner et al. 2006), so I had decided in advance to operate the sessions on a pair-wise basis; of course this also simplified the logistics as we had fewer active sessions to monitor.

To facilitate the pairing-up process, at the beginning of the workshop on the first day I used an icebreaker activity during which my assistants and I were able to meet the girls and they could also learn a little about each other before they chose a partner for the series of workshops. For the few girls who still were a little uncomfortable choosing a partner the classroom assistants provided some help based on the responses given during the icebreaker (for example, books they had read, places they wanted to visit, etc.). To help in further breaking the ice between the newly formed pairs, I asked the girls to come up with a name for their team. The result was one team of 3, with all the rest of the girls in pairs. On a few weeks, partners were not present; for example
one week all of the 6th graders were on a school field trip. On those occasions the girls were given the option to join another team that was missing a member or to work by themselves.

Girls’ experience with computing tools varied. From classroom observations a few girls were worried they would be work with MS Excel, implying that they already had experience with this tool. As one participant put it “Please don’t tell me we’re doing excel, I hated doing excel!” One girl had worked with Scratch while attending a private school. In general however, most of the girls indicated that they were unsure if they had worked on any of the tools we would be using in the workshop series.

I recruited two workshop assistants to assist the participants; this was particularly important as the activities increased in complexity over the six-week period. It also helped me to make the most of the 90-minute session, during which I needed to set up laptops (which we provided in this case), introduce the topic of the day, and answer questions as the session progressed. Except for two non-overlapping occasions the assistants were there each week to provide support to me in collecting classroom observations and helping the girls with any problems they encountered. The week the database material was covered, one of the assistants led the actual workshop because he had built the tools and was most familiar with its operation. This was also the week we discovered that some of the girls had gained experience working with data from previous schoolwork where they used a combination of MS Excel and Google Docs tools. The week of the robotics was the most novel and challenging for the participants; this was true particularly for the 6th graders because they had just begun the tech education class offered by their school, and had not yet been introduced to the Lego Robots activity often introduced in this class.
4.1.2 Procedures

The first week of the GCI-1 workshop series concentrated on introducing the several computing tools that the girls would be using during the upcoming weekly activities, as well as giving them time to learn about each other before choosing a partner. During an icebreaker activity, they were offered three different types of candy and invited to take 1 to 3 pieces, though there was no expectation that they would eat any of it. The girls then answered questions such as “What is your favorite book?”, “What is your dream vacation location?” and “What is your favorite movie?” The questions they answered depended on the type of candy they selected. After introductions they were asked to find a friend and move to one of the laptops.

Because I did not know how comfortable the girls were with the technology and in particular whether their relative lack of experience would cause a problem with entering unknown passwords, I chose to have the laptops logged in and set up for the girls so that they could start working right away. In the first week this provided an opportunity for the girls to follow along while I demonstrated the Alice and Scratch environments.

The girls were also provided a workbook that guided them through the introductory activity I had planned for that day, giving them an opportunity to try out both Alice and Scratch but without a specific learning focus. They were encouraged to explore the environments, try out some of the suggestions in the workbook and in general ‘play’ with the stories and games provided that day. At the end of the session the girls were handed the recruitment packets and turned their workbooks into one of the assistants. Some of the girls asked if they could save the programs they had started and an assistant helped ensure they did not lose any of their work.
During the 2nd through 5th weeks the first 15-20 minutes of each after-school session was devoted to introducing the scaffolded example activity for the week, ensuring that the girls had everything they needed for the activity, and re-pairing anyone with an absent partner who wanted to be part of a different team. The introduction included a short demonstration of the computing tool being used for the session, handing out of the workbook, and ensuring that all pairs could open the scaffolded example file on their laptops. After the brief introduction, the girls worked together, using the workbook to guide their activities.

For GCI-2, the workbooks also contained areas for the girls to document reflections and requested responses to conceptual questions concerning the day’s activities. As the assistants and I walked around the room to answer questions, we noticed that many of the girls were leaving questions blank or entering answers that were identical to their partners. Thus while the workbooks were effective in suggesting goals, sub-goals and approaches to the tasks (i.e., operating as guided exploration manuals, (Carroll and van der Meij 1998), we began to sense that their dual use as data collection mechanisms would not be very robust.

The sixth week was set aside for what I termed “Girls Choice”, an on your own time session during which they could choose to create new projects, work on previously covered material and in general prepare for the parent showcase scheduled for the end of that day. For GCI-1, the parent showcase revolved around an animal park exhibit that used the interactive structures the girls created the week before and two additional interactive structures (a waterfall and animated giraffe) available as an extra activity for the day. I worked closely with the two teams who decided to work on these new structures, providing assistance with the coding needed to interact with the structures. The girls were able to envision how the structures needed to work without workbooks to guide them. The assistants helped the other teams by answering questions or ‘fixing’ programs that were not working. After all the structures were working and the computing projects were fixed the girls put the animal park display together (Figure 10).
4.1.3 Materials

The participants completed two surveys as a part of the GCI-1 workshop series (See Appendix C). A pre-workshop survey collected demographic data (grade, age) as well as rating scales designed to investigate the girls’ computer self-efficacy (ref), how they feel when they use a computer (the microcomputer playfulness scale, ref), and their identification with careers that are focused on computers. The girls completed a post-workshop survey at the end of the last workshop; this included a repeat of the computer self-efficacy scale, open-ended short answer questions related to a CT concept from one of the computing tools, and reactions to the activities in the workshop series (Appendix D).

The girls received a workbook to help guide them through each of the weekly scaffolded examples to complete the computing activities. In addition to responding to questions about the activity and CT concepts the workbooks in the GCI-1 also had areas to journal their workshop experience. End of day questions were also placed on the last page of the workbooks with
questions like “What did you enjoy doing today?” and “What do you wish you could have done today?” Appendix E documents selected pages from the workbooks.

Most importantly, the materials each week included a scaffolded example of the forms described in Chapter 3, and for this workshop relating to the overall theme of the animal park, which was introduced on the first day. Details of each activity are provided in Appendix B, but Table 11 summarizes the four scaffolded example learning activities and objectives for this workshop series.
Table 11 - Activities and Objectives for GCI-1

<table>
<thead>
<tr>
<th>Workshop</th>
<th>Description</th>
<th>Specific Learning Objectives</th>
<th>CT Goals</th>
</tr>
</thead>
</table>
| Interactive Stories | The activity involves a story about a Vet and a Mama wolf which needs the sequential sequence of the story fixed, dialogue corrected to match the character activities and use variables to control the action sequence based on users input. | • Be able to use brainstorming to develop potential solutions  
• Use methods to organize solutions  
• Understand computing vocabulary | • Design and algorithmic thinking  
• Use the vocabulary  
• Recognize abstractions and move between levels of abstractions |
| Games             | The activity involves working with a game for the animal park exhibit that teaches young children about fish and what they should/shouldn’t eat. The game is considered “broken”, what needs to be fixed is determined and algorithmic thinking is used to create solutions to implement. | • Use and understand program constructs to control game objects  
• Understand the use of variables  
• Understand broadcast messaging  
• Test and Debug | • Design and algorithmic thinking  
• Use the vocabulary  
• Recognize abstractions and move between levels of abstractions  
• Test and Debug |
| Database/Webpage  | The activity for this workshop is working with the GirlsCreateIT! Database to tables of animal information (habitat, weight, diet) for the animal exhibit by creating queries to access stored data and then produce a webpage from the queries. Images are worked on with a pixel editor to work with alternate data sources for the webpages. | • Understand how data is stored  
• Design simple queries to access stored data  
• Use html editor to create a webpage from the queries  
• Test and Debug | • Design and algorithmic thinking  
• Use the vocabulary  
• Recognize abstractions and move between levels of abstractions  
• Test and Debug |
| Interactive Structures | The activity for this workshop involves working with the Lego Robotic Mindstorm System to create interactive structures (bushes) that interact with the visitors at the animal park. The participants learn about external sources of input by working with sensors (touch and light) and output by activating a motor for the structure in the overall animal park exhibit. | • Understand the difference between input and output  
• Work with a touch or light sensor to control a motor  
• Download programs from one computer to another(RCX)  
• Test and Debug | • Design and algorithmic thinking  
• Recognize abstractions and move between levels of abstractions  
• Use innovation and creativity with technology  
• Test and Debug |

Table 11 provides an overview of the activities, the learning objectives and CT concepts covered in weeks 2 through 5 during GCI-1.
4.1.4 Results

As previewed in Chapter 3, the exploratory nature and field setting of my research program requires a rich body of converging measures. In GCI-1, these included surveys used to both characterize the girls’ background knowledge and attitudes and to investigate whether these changed over the course of the workshop; informal notes taken during the sessions; and comments written into the workbooks in response to either scaffolding instructions or CT reflection probes. Detailed examples of these instruments are in the Appendix; in this section I summarize key findings across the different assessment methods.

Pre- and post-GCI-1 surveys. One goal of the pre-workshop survey was to investigate general inclinations concerning careers. I drew seven items from a career inventory that had been developed and used in prior research with middle school girls (Craig and Horton 2009); we also included “computer programmer” to investigate where it would be in the mix. Girls rated their career interest on a scale from 1-7 and the ranked list based on means and standard deviations (n=24) appears in Table 12. The ordering is not surprising; many girls this age are interested in teaching and when drawing from a community of highly educated families with many professors as parents, one would expect high interest in science and engineering jobs. Computer programming is relatively low in the list; even fashion design seems more attractive to these girls.

Table 12 - Career Inventory Ratings

<table>
<thead>
<tr>
<th>Career Possibility</th>
<th>Mean (SD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teacher</td>
<td>4.50 (1.7)</td>
</tr>
<tr>
<td>Veterinarian</td>
<td>4.29 (2.2)</td>
</tr>
<tr>
<td>Scientist</td>
<td>4.25 (1.9)</td>
</tr>
<tr>
<td>Engineer</td>
<td>3.83 (1.9)</td>
</tr>
<tr>
<td>Doctor</td>
<td>3.79 (2.0)</td>
</tr>
<tr>
<td>Fashion Designer</td>
<td>3.63 (2.2)</td>
</tr>
<tr>
<td>Computer Programmer</td>
<td>3.37 (1.8)</td>
</tr>
<tr>
<td>Nurse</td>
<td>2.96 (1.7)</td>
</tr>
</tbody>
</table>
I also included five items from a computer science career identification scale that had been used with older participants (high school and college age females; (Rosson, Carroll et al. 2011)), to determine whether middle school girls were able to respond to these items in a meaningful way. Unfortunately in contrast to the work with high school and university students, these items did not correlate well enough to form a single scale ($\alpha = .57$). However, I found that if I dropped the two items with poor correlation (“Computer jobs are boring”; “I have a family or friend who works in a computer-related job”), the reliability of the remaining three items was good ($\alpha = .71$). The resulting “Career ID” construct had a mean of 4.79 (1.3), suggesting that these girls were just a bit beyond neutral in identifying with computer-based jobs. I also included another scale that has been used to assess the strength of stereotypes about computer-based jobs (geeks, hard-working, smart, anti-social, creative). After recoding the negative stereotype words, the scale was reliable ($\alpha = .76$), producing a mean value of 5.11 (1.2). Finally, I piloted a computer playfulness scale; this construct is a surrogate for people’s intrinsic motivation for using computer-based tools and has been shown to be a strong predictor of identification with computing careers (Rosson, Carroll et al. 2011). The scale had high reliability ($\alpha = .89$) and yielded a mean value of 5.10 (1.2). In general, these findings point to several scales that might be useful as indicators of the girls’ interest in computer-based jobs and activities.

Another important scale concerned the girls’ perceived self-efficacy for computer-based tasks. To investigate this, I adapted an existing 6-item scale that has been used in many settings to measure computer self-efficacy (Compeau and Higgins 1995; Bandura 1997). The reference task was specified quite generally (“what you expect when you try to use new computer tools”), and the six ratings had only a moderate level of reliability ($\alpha = .63$). Surprisingly, the mean value for pre-workshop self-efficacy ratings was 5.37 (0.9), suggesting that at least for general computer-based tasks, the girls felt a moderately high degree of self-efficacy. Interestingly, we repeated this
scale after the workshop, but framed it differently, asking them to focus on judgments for tasks similar to the Alice, Scratch and other tools they had used in GCI-1. In this case, the mean ratings for the items declined, with a mean value of 5.04 (1.2). A paired t-test indicated that this decline was significant (t(21)=2.20, p<.04), leading to the uncomfortable interpretation that the workshop activities had made the girls less confident about their capacities for computer-based tasks. However because I changed the efficacy task frame - it was more specific and referenced the actual workshop tools and activities – I do not know whether or how to interpret this decrease in mean value. Instead I speculate that assessing general computer self-efficacy in this fashion is not a reliable method for tracking changes in CT understanding. Thus I created my own specialized scale to use in GCI-2 and GCI-3.

Another set of items in the post-workshop survey explored the perception of project confidence and fun; I expected these to be indicators of whether girls like this might continue to engage in CT-related activities. I used five items to probe confidence in their work on the various activities comprising GCI-1; this scale had good internal reliability (α = .81), with a mean value of 5.68 (1.0), suggesting that the girls were moderately confident about their understanding and use of the tools. In contrast, the items asking for “fun” using the various tools (Alice, Scratch, GirlsCreateIT! DB, Lego RCX) were not intercorrelated (α = .16). Different girls responded differently to the different activities, but in general the ratings were positive, ranging from the database tool (5.24) to the Lego RCX activity (6.15).

The responses to the open-ended questions on the post-workshop survey provided insight into the girl’s experience with the multiple computing tools. In regards to Alice, the responses indicated while some liked it, it wasn’t a favorite:

\textit{P4: I liked working with Alice, but I enjoyed some other activities more}

\textit{P6: Fun but I liked Scratch better}

\textit{P14: I had lots of fun because of the weird and silly things we had to change.}
P17: I really enjoyed working with Alice, but some of the glitches are annoying.

P21: Um it is short of boring cause it wasn't as creative as the game.

It was encourage find CT related responses in their experience responses:

Scratch:

P2: It was great to get to play the game, and make it work, after we fixed it.

P15: I loved Scratch because it was more challenging to think of solutions to a problem.

Interactive Structures:

It is always fun to design things and even more when they move.

P9: I (heart symbol) Legos!! and it was cool to program and make them work.

P14: Because we got to build and fiddle with it and I laughed the whole time!

Classroom observations: During the workshop sessions the assistants and I walked around the room offering help or suggestions as needed. Each of us tried to informally document our observations after each workshop; however one assistant did not contribute many such observations. Thus the data summarized here are drawn two researchers’ notes.

In week two, during the Alice 3D story activity, we observed that girls became frustrated when characters’ limbs would become detached or wolf characters would walk into or underneath the grass. Another pair of girls had trouble articulating what might count as “acting scared” behaviors; for them, the problem related to the abstraction of working with a character in a story. It was too removed from their day to day experience; when instead we encouraged them to consider how this behavior related to them personally it became clearer what they needed to do.

We also observed that pairs of girls seemed to work well together, but that the group of three was not optimal. The third girl in the group was treated like an unwanted extra and thus appeared seemingly uninterested making it difficult for them to finish the activities, however
when the girl was not in attendance for some of the sessions, the remaining two girls were more communicative and appeared to enjoy the activity.

The workshop activity involving the GirlsCreateIT DB was very easy for the girls to complete; in fact some of the girls had an opportunity to work on activities from previous weeks when they finished early. Interestingly, a few groups asked to create more complex queries for the database of animals, but this was not possible because the query tool had been designed to work only with simple queries.

I found the classroom observations from my assistant during the Lego sensors activity to be particularly interesting. She observed that though the session was confusing and complicated, including for her because she was not very familiar with this type of activity, she was still able to help the girls complete the activity. She also noted in her write-up that the girls helped others once they were finished with their project, “The good thing was eventually a few groups finished and were nice to help other groups when we were not able to help them”.

Workbook comments. The workbooks contained open areas for the girls to document reflections as they worked; it also requested responses to conceptual questions concerning the day’s activities. Each week I found that the girls started out by using the areas provided to brainstorm their ideas, come up with solutions and document what worked and what did not work. However as each day’s activity progressed, the girls became less interested in writing in the workbooks and more engaged in the activity itself. This was confirmed when the collected workbooks had blanks in many of the questions on the later pages; or the girls “shared” their answers to speed through a section with reflections.

The workbook entries did help to confirm classroom observations related to confusion about abstractions they girls were encountering related to human emotions. For example, the workbook entries to the question “What can we use from the list that could be used in an ActScared method? (i.e., browse the existing methods in the scaffolded example,) some of the
girls’ entries showed a mixed understanding of moving from the real world to a virtual world. Responses included: P1: “frantic arms, move at speed, faint, run away, say something, back away”, P7: “Say Oh No!” and P12 “We gave you a treat!”. The methods available included faint, frantic arms; the others might have been real world solutions but they were not available in the list of methods in the Alice programming environment.

During the game activity I asked “What can we do to stop the game with a nice message screen? List your ideas here.” The girls’ responses suggested confusion about the role of a win or lose screen message. For example “Game over better luck next time” (P3); “Make a control and make it stop everything and put up the nice message screen” (P5); “Stop script, broadcast” (P22). It was clear they girls knew what they needed to do but were unable to articulate the code changes at this point.

By the second week the workbook entries did suggest that the girls were starting to understand programming concepts like methods and variables. On pair commented they enjoyed “Experimenting with all the different methods” (P22). In addition we asked a comparison question asking for reflection across the first two weeks: “What did you need to use in the FishFoodFun Game that you also needed for the Wolf and Vet Story?” Almost 50% of the workbook responses mentioned the word “variables”, which is a reasonable suggestion. However, with respect to control logic the understanding was less clear: when asked about how they used the “IF” statement, one pair responded “What if statement?” (P2) and another simply said they did not have fun working with the “if thingy” in the story.

Each workbook also contained end-of-day questions to provide information on what they enjoyed doing that day, what they needed more time to work on and if there was something they had hoped we would cover. Responses to these questions provided insight into some of the fun the girls had doing the activity and also provided an insight into the complexity of the activity. For example, the girls asked for more time to work on games that they would create themselves;
they also wanted to spend more time on the sensors and motors so that they would have “been able to make them work without help”.

4.1.5 Discussion

To investigate the first research question, *Are girls this age able to grasp CT concepts and skills?*, the CT concepts of problem solving and abstraction were practiced in an interleaved fashion as the girls worked to understand and solve the problems that were posed. To guide these general activities, a computational problem solving process that included algorithmic thinking while brainstorming ideas for potential solutions, choosing among alternatives, and designing and implementing a computer-based solution was introduced in the first week. For example, during the interactive story activity the vet needed to act scared because the example story illustrated her coming up to an angry wolf and showing no affect. To solve this problem the girls first brainstormed about real world reactions when a person is scared. They sketched out a sequence of events (the logic of an algorithm) for initiating a scared reaction, and finally explored features of the programming environment (method calls in Alice) to the computational elements they needed for their solution (e.g., they found the Vet character had the methods move away from so she could leave and also pass out if they had thought about fainting).

As I noted earlier, in one case a pair of girls had trouble articulating what might count as “acting scared” behaviors; for them, the problem related to the abstraction of working with a character in a story. It was too removed from their day to day experience; when instead we encouraged them to consider how this behavior related to them personally it became clearer what they needed to do.

To address RQ2, *How did scaffolding help in learning CT concepts?*, the workshops explored different types of scaffolding to determine whether and how the techniques could
introduce the CT concepts in a manner that encouraged the girls without overwhelming them. Because there was a two-year age range among some participants, I expected that some girls might arrive with rather different levels of cognitive development, computing experience and general problem solving abilities. Thus it was important to provide customized assistance as needed during the workshops, to help encourage enough confidence and motivation for the girls to discover answers on their own. For this reason each workshop used scaffolded examples with an accompanying activity workbook where the girls would find instructions, tips and could record their answers to questions interwoven into the activity. Classroom assistants observed the girls’ work and offered advice as needed.

In the interactive story workshop where we used a 3D environment (Alice 2.2, where the camera is a core object used in construction activities), the camera added complexity to the introductory projects. Ultimately the girls’ efforts to control (or avoid controlling) it interfered with the primary learning objectives. This problem may not seem significant but it highlighted the general issue wherein the girls often shifted their focus from the CT concepts to the “aesthetics” of the computing artifact being created. Instead of focusing on problem solving and the storyline, girls spent valuable time to get angles positioned correctly. Thus in this particular case, the scaffolded example raised complexities (i.e., exposing code that is not useful as part of the learning activity). For this reason I decided to limit my focus to computing tools that de-emphasize camera control and movement.

In contrast, the Scratch-based FishFoodFun scaffolded example allowed the girls to work in a simpler 2D environment; this in turn helped to keep the learning focus on key aspects of problem solving, use of appropriate vocabulary, and game testing and debugging. One measure of this during the Alice story activity the girls tended to write workbook comments that wondered about how to position characters in the Alice story; in Scratch they spent most of their time
solving the (deliberate) inadequacies of the example game and also embraced the script repetition and decision control logic.

While using the GirlsCreateIT DB tool helped place the focus of the workshop activity on learning how data is stored, accessing the stored data and problem solving when returned results were not the ones expected, it was not able to scale to a more complex activity. This was an encouraging finding, indicating the girls wanted more challenging activities as the series progressed. This desire for more challenging activities was met the next week with the robotic activity. Of all the activities this was the most complex, involving additional hardware (e.g. sensors, RCX brick) and a new software tool. However, problems encountered during this workshop indicated that the technology was too old and that too much extra time would be needed to learn a new programming environment.

GCI-1 provided opportunities to learn what worked and what needed changed. The number of computing tools needs to be narrowed to a platform that keeps the focus on the CT concepts and doesn’t add to the complexity of the activity. The decision to bring our own laptops were not an ideal solution; not only did they take additional time to set up (timing of the after-school workshops did not provide any extra time to set up before the girls arrived), they also caused confusion when finding previous work when the girls worked in a new team (for example, when a partner was not in attendance and two girls formed a temporary team). Finally, I had envisioned a much more extensive role for career discussion and exemplification; however was simply not enough time in each 90-minute session to address these more general issues, and I eventually decided to eliminate the broad discussion of how computers are used in different careers. Going forward, I focused more specifically on the girls’ more general identification (before and after the workshop) with computing-related jobs.
4.2 Spring 2012 (GCI-2): After-School GirlsCreateIT! Program

The GCI-2 program for middle school girls was held during Spring 2012, again organized as an after-school workshop that was offered to girls in 6th to 8th grade, one day per week for six weeks. The general format for the workshops was the same as the GCI-1 workshop series; however the scaffolded example activities and measurements were refined based on findings from the fall study. The weekly sessions continued to use scaffolded learning activities to introduce and build on CT concepts requiring minimal instruction, and the girls again worked in pairs, with one group expanding to three girls when one girl’s partner left.

4.2.1 Participants

The GCI-2 workshop series was created and delivered as part of the State College School District Outreach Program, in Spring 2012. As with the GCI-1 workshop series this offering was designed for a group of 16, though only 9 girls consented to participate and one girl left the program after a few weeks, so I have just partial data from her. The girls in the team of three were asked if they wanted to work with the girl who lost her partner however none of them wanted to leave the trio and the other girl asked if she could work by herself, which was not optimal but worked well for her.

Though the program was advertised to girls at all grades in middle school, it again attracted more 6th graders than 7th graders (44%) but no 8th graders (Figures 10 and 11). In addition one participant was a repeat; she had been in GCI-1 and was now attending with her friend as her partner. They were the only pair who missed one of the weeks; they also were not able to stay for the parent showcase because of a schedule conflict.
Because GCI-2 was held on one day a week, the assistants from GCI-1 were again to help with set-up, answer the girl’s questions and collected classroom observations. In this workshop series, the girls used school computers, so the assistants did not need to help with computer login. Instead, the assistants spent time at the end of each day helping the girls to download their work onto USB drives. On occasion the assistants also helped the girls upload the scaffolded examples to the Scratch programming environment.

As in the GCI-1 study the first week remained the same by me leading an icebreaker after all the girls arrived. This time the 3 candy questions not only served to introduce the girls to each other it also provided a quick pick me up since they informed us they were exhausted from all the tests taken that day. After introductions they were asked to find another girl to work with as a partner and come up with a team name. Each team then chose a classroom station (with the switch to one computing tool the classroom computers could be used in GCI-1).

Since GCI-2 was held only one day a week, the assistants from GCI-1 were available to again help with set-up, answering the girl’s questions and conducting classroom observations as time permitted. With the girls using the school computers the assistants didn’t need to help the girls’ login to the computers, but at the end of the first session it was found the assistants needed
to help the girls download their work on to the USB drives. On subsequent weeks the assistants also helped the girls when they needed to upload the scaffolded examples to the computing tool.

4.2.2 Procedures

I followed a session and week-by-week scheme similar to that of the GCI-1 series, with an emphasis on four different scaffolded example activities designed to exercise particular CT concepts or skills (see Appendix B). The workbook with journaling probes for reflections used in GCI-1 was redesigned into a simpler workbook to help guide the girls through the self-paced activities. Also in GCI-2 the conceptual questions posed throughout the workbooks in GCI-1 were converted to end-of-day questions.

As in the GCI-1 study, during the first week I used an icebreaker to make the girls comfortable and to help my assistants and me get to know them. After introductions the girls paired up and came up with a team name. Each team then chose a classroom station; the use of a school based lab and equipment was a convenience but it was only possible because we had simplified the activities to use a single platform (Scratch). While the assistants handed out workbooks, I gave a short overview of the upcoming week’s activities and briefly introduced them to Scratch. They then used the workbook to explore the Scratch user interface; they also were guided to work with a sprite (the underlying concept for all Scratch characters); finally there was a short review of the overall workshop. As part of their Scratch exploration, all of the pairs discovered the built-in pixel editor; as a result everyone ended the day creating or modifying sprites and backgrounds. At the end of the first day the assistants showed the girls how to save their work on the USB drive (for safekeeping until the next workshop), because the default in Scratch is to save all work on the computer drive (in this case a shared school computer). Once the girls handed in the workbooks and USB drives they could then select a small snack.
The next four weeks followed the same format as I used for GCI-1: I began each session with a high level overview of the day’s activity (10-5 minutes). One difference for this workshop series is that I no longer needed to preview the computing tool to be used each week; the use of Scratch (even for the Lego sensor activity) simplified the “Start up” for each activity tremendously. Note that throughout the learning activities, the girls continued to explore the pixel editor, even though it was never mentioned or encouraged in the workbooks; clearly the design of “looks” in Scratch is an attractive activity for this population. As before, my assistants and I walked around the room each week to help girls who had questions or were struggling. At this time, if they were focused on pixel editing, we would encourage them to follow the workbook so that they could complete the activity, pointing out that they could work with the pixel editor once they were finished for the day.

In the fourth activity, the girls were guided to work with Lego bricks and the Lego WeDo sensors. I helped the girls to construct a music box and the assistants helped the girls work with the motors. At the end of each session the assistants also ensured that all of the Scratch artifacts the girls had built were downloaded to their team USB drives to be turned in with their workbooks. Once the girls turned in the materials for the computing activity they were offered a snack while they completed end-of-day question, which in GCI-1 were a part of the workbook but were separated to ensure the girls filled them out (Appendix F).

The last week was again set aside for “Girls Choice”, an on your own time session to create new projects, work on previously covered material and prepare for the parent showcase. For the GCI-2 workshop series, the parent showcase did not include a large display for all the girls to work on together. Instead the teams worked on Scratch projects that they wanted to show their parents; they spent the final 30 minutes or so preparing games or other projects to share with their families. One of the more advanced teams (the one with a member who had participated in GCI-1) worked on a new two-player game, but the other teams chose to refine example they had
worked on in previous sessions. During the showcase, each team selected one project for me to demonstrate to all parents at the beginning of the showcase. After the ‘project show’ the teams then shared their other projects. Many girls used these showcase interactions as a sort of “mini Scratch lesson” for their parents.

4.2.3 Materials

The GCI-2 activities were similar to those in GCI-1; each session revolved around a scaffolded example activity designed to illustrate and exercise particular CT concepts or skills (see Table 13). However, unlike GCI-1, all of the scaffolded examples were built using Scratch. Even the Lego WeDo sensors and motors activity, which operated through a USB hub, used the Scratch programming environment. The theme for the scaffolded examples was “Protect the Environment” and the story, game, data and sensors/motors illustrated different aspects of this theme while conveying the CT concepts in ways similar to what I had done with the previous set of animal theme park examples. Detailed descriptions of each example can be found in Appendix B and a summary of the activities and their CT learning objectives appears in Table 13.

Table 13 - Activities and Objectives for GCI-2

<table>
<thead>
<tr>
<th>Workshop</th>
<th>Description</th>
<th>Specific Learning Objectives</th>
<th>CT Goals</th>
</tr>
</thead>
</table>
| Stories  | The activity involves a story of four friends who are worried about the pollution in the water. The activity involved working on the broadcast messages to correct the sequential sequence of the story, correct the dialog to match the character activities | • Be able to use brainstorming to develop potential solutions  
• Use methods to organize solutions  
• Understand computing vocabulary | • Design and algorithmic thinking  
• Use the vocabulary  
• Recognize abstractions and move between levels of abstractions |
### Games
The activity involves working with a game to teach young children about fish eating pollution for food. The game is considered “broken”, what needs to be fixed is determined and algorithmic thinking is used to create solutions to implement.

- Use and understand program constructs to control game objects
- Understand the use of variables
- Understand broadcast messaging
- Test and Debug
- Design and algorithmic thinking
- Use the vocabulary
- Recognize abstractions and move between levels of abstractions
- Test and Debug

### Lists of Data
The activity involves working with Earth Day phrases that are translated into other languages. The phrases and languages are stored as lists of data that are partially set up, in addition error processing to handle the problems with incorrect input from the user.

- Understand how data is stored
- Create and initialize new lists of data
- Modify existing lists
- Incorporate error handling for user input
- Test and Debug
- Design and algorithmic thinking
- Use the vocabulary
- Recognize abstractions and move between levels of abstractions
- Test and Debug

### Interactive Structures
The activity working the Lego WeDo system with an animated story of a bird cleaning up its living space. The participants learn about external sources of input by working with sensors (tilt) and output by activating a motor to create a moving external structure.

- Understand the difference between input and output
- Work with a tilt sensor to control an animated character that will activate an external motor
- Understand the interaction between sensor input and motor output within a Scratch animation
- Test and Debug
- Design and algorithmic thinking
- Recognize abstractions and move between levels of abstractions
- Use innovation and creativity with technology
- Test and Debug

The workbook was simplified by removing the brainstorming and reflection activities (i.e., the CT journaling that was not very successful as a data collection mechanism), so that it was simply a workbook that guided the girls through the self-paced activities. In the GCI-2 iteration, the conceptual questions about CT were converted to end-of-day questions that the girls completed after their USB drives and workbooks had been turned in and they had finished the daily activity.

The classroom observations from GCI-1 had suggested that the Lego RCX hardware and programming environment was too complex to set up and learn in a single 90-minute workshop session. The programming “code” and associated editor was very different than what they were doing with the other tools, and it became even more complex when the girls began to have polarity issues with the older sensor technology. Switching to the Lego WeDo system allowed me to create a more seamless transition to sensors and motors for the final activity. This provided
an opportunity for the girls to encounter and practice their skills in a single environment as they were being introduced to foundational concepts in computer programing.

The surveys used in GCI-1 were modified after classroom observations found the girls had trouble translating their career choice into one of the careers listed in the background survey and also when the girls needed help in understanding how to interpret the self-efficacy questions. For GCI-2, the career assessment list was removed and replaced by the simpler open-ended question “We know you have a long time to decide this, but what career do you think you might pursue in the future?” Details of the revised surveys can be found in Appendix C.

In addition a new computational thinking self-efficacy scale was created for use on both the background (pre) and post surveys. The goal was to create a set of items that could probe the girls’ perceived self-efficacy for computational thinking capacities, rather than computing tasks more generally. I first developed a set of questions that posed tasks related to the CT concepts I had been seeking to convey (e.g., problem-solving, abstraction, design and algorithmic thinking). The CT concepts of each question came from the CSTA focus as discussed in Chapter 2 in Table 2 and were selected as those that would work best in a classroom setting. However, the resulting items seemed rather abstract. Thus, I approached one of my GCI-1 participants (seventh grade honors student) to refine the questions: First I would read the original question and ask her what she thought it meant, and then I explained what the question was trying to ask, using examples she might encounter in her school work. The eventual result was a new self-efficacy scale designed to assess computational thinking for middle school students. See Table 14.

### Table 14 - New Computational Thinking Self-Efficacy Scale

<table>
<thead>
<tr>
<th>Items and CT construct being assessed</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>I can start with an assignment and guidelines of what to include, and come up with my own ideas on how to solve the problem, even if I’ve never worked on that type of problem. (problem-solving)</td>
<td></td>
</tr>
<tr>
<td>Once I know what I need to solve a problem, I can decide which parts can be solved with a computer, even before I work out a detailed plan to solve the problem (design solutions)</td>
<td></td>
</tr>
</tbody>
</table>
If someone shows me a computer application, I will be able to understand the words, commands, or images that I need for solving my problem, even if it’s the first time I used the application.

(*computational thinking vocabulary*)

After I have created a plan for how to use the computer to solve a problem, I can write down a set of instructions the computer can use in the solution, even if there is no one there to help me.

(*algorithmic thinking*)

If there are two possible ways to solve my problem with a computer, I can talk about why one or the other is better, even if I have not tried out either one yet.

(*abstraction*)

### 4.2.4 Results

As in GCI-1, I gathered a variety of different forms of data to study the girls’ understanding and application of the CT concepts and the impacts of the learning activities on their attitudes about computing. However I made several changes in the survey questions to refine my measurement methods. Note that because the number of participants was considerably smaller than GCI-1, the survey data are more challenging to summarize and interpret. However, I was more systematic about collecting and archiving the girls’ artifacts each day (i.e., their Scratch projects), so that I was able to use artifact analysis as another converging method.

*Pre- and post-workshop surveys.* The pre-workshop survey was considerably simpler than the version piloted in the fall: I assessed Career Identification (using the three items with highest internal reliability in Fall 2011); Computer Playfulness (the same seven items); and the new scale of CT self-efficacy. The means and standard deviations for these three constructs appear in Table 15; it was somewhat surprising but also encouraging that even with the small sample size (n=9) the internal reliability of these three scales was strong. Also note that the mean values for playfulness and self-efficacy tend to be lower for GCI-2 participants than they were for the larger fall group, perhaps at least partly because the girls were younger overall.
Table 15 - Attitudes Assessed via the Pre-Workshop Survey in GCI-2

<table>
<thead>
<tr>
<th>Item Index</th>
<th>n=8</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Career Identification ($\alpha=.87$)</td>
<td>4.70 (1.5)</td>
<td></td>
</tr>
<tr>
<td>Computer Playfulness ($\alpha=.78$)</td>
<td>4.60 (1.0)</td>
<td></td>
</tr>
<tr>
<td>CT Self-Efficacy ($\alpha=.90$)</td>
<td>4.38 (1.2)</td>
<td></td>
</tr>
</tbody>
</table>

The survey used to assess reactions after the GCI-2 workshop was similar to what I had piloted in the Fall 2011 session, except that I again used the abbreviated 3-item Career ID scale, and the new CT self-efficacy scale. Means and standard deviations appear in Table 16. The Project Confidence and Fun items were virtually identical to those used in GCI-1, except that Scratch was always used to refer to the programming tool in use. Interestingly, in this workshop series where all activities took place in Scratch, the Fun construct now also had strong internal reliability, suggesting that it might have been the programming platform rather than the type of example that caused the lack of coherence in the Fall 2011 survey. As I had observed for GCI-1, the girls seemed to be moderately confident about the projects they had completed over the six weeks, and had fun doing the different Scratch projects. For this group, the first (stories) project was rated most positively with respect to fun, followed by the second (games) project.

Table 16 - Attitudes Assessed via the Post-Workshop Survey in GCI-2

<table>
<thead>
<tr>
<th>Item Index</th>
<th>n=8</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Confidence ($\alpha=.97$)</td>
<td>5.68 (1.3)</td>
<td></td>
</tr>
<tr>
<td>Fun on Projects ($\alpha=.89$)</td>
<td>5.59 (1.2)</td>
<td></td>
</tr>
<tr>
<td>Post Career ID ($\alpha=.88$)</td>
<td>5.00 (1.5)</td>
<td></td>
</tr>
<tr>
<td>CT Self-Efficacy ($\alpha=.94$)</td>
<td>5.08 (1.9)</td>
<td></td>
</tr>
</tbody>
</table>

With respect to changes in attitudes, the difference in construct means (i.e., post-workshop data minus pre-workshop data) is in the right direction for both the Career ID measure
(Δ=.13) and CT Self-Efficacy (Δ=.78). However these increases are too small to be significant in a paired comparisons t-test. Nonetheless, the data collected from this small sample of eight girls (one of the nine did not complete the post-workshop survey) did serve to increase my confidence that these measures would be a useful set to include in my final study.

Classroom observations. In GCI-2 the assistants and I again walked around the room offering help or suggestions as needed. Each of us tried to informally document our observations after each workshop; however one assistant again did not contribute many such observations. Thus the data summarized here are drawn from two researchers’ notes.

During the introductory week we had observed the girls interest in the pixel editor. As we walked around during the story activity to encourage the girls to complete the learning activity we told them there would be a showcase of their projects at the end of the day, however we found the pixel editor was too distracting and some of the girls were racing to finish the story at the end. One girl even commented that she wished they had not spent so much time creating new sprites and backgrounds because they had wanted to add more to the storyline. We told them they could work on it the following week.

The desire to customize their activities was very important to many of them from the very first week. As we checked on the different teams we noticed there was a morbid fascination on how sick they could make the octopus look. One team which had spent so much time in the pixel editor during the introductory week was determined to stay focused on the activity. One of the girls would comment to her teammate “We can do that later, let’s get this done first!” We also saw the girls were intent on getting new characters in the story and including them into the story by adding additional broadcasts such as “falling to the ground” and “leave the screen”. All of the girls were very involved in the story and I seemed to be interrupting them when I came around to find out how they were doing. Instead of turning around they stayed focused on the project, nodding their heads while saying “Okay”.
The group of three in GCI-3 worked better than we thought it would. The girls took turns working on the computer and were willing to try each other’s ideas, it may have helped that there were 3 very vocal, strong personalities in the group.

It was interesting to discover how much fun the girls had with the translator activity that used lists of data. The girls ended up spending the entire afternoon adding new phrases and using the Google translator to convert their phrases into new languages. There were teams interested in Chinese and Japanese, however they were uncertain if Scratch would display the characters correctly, we encouraged them to try it but found they found other languages to use. There was a little confusion at the beginning of the activity since the girls felt there should be more instructions in the workbook and they problems they needed to fix were not as apparent to them.

The sensors and motors used the new Lego WeDo system and again we observed the girls extending the scaffolded examples with their own sprites, extra characters and new code to change the activity. They enjoyed moving the characters in the animation with the tilt sensor and one team was very meticulous in their creation of a music box using a lattice pattern of Lego bricks. This extra attention to detail frustrated one of the team members who said “Okay you guys work on that and I’ll write the code!”, because she was able to see that the code could be written while the music box was being constructed her team finished on time.

On the final week the girls had the opportunity to create a new project, though only team decided to create a game, which they completed in the 90 minute session. As they worked on how the users would control the characters, they decided to try out the tilt sensors but ran into trouble when the instruction blocks couldn’t handle two different tilt sensors. Instead they ended up using two different keyboard keys to move their characters to the finish line. When I asked what they had done for the win/lose conditions they said “We don’t need a lose screen, since there are only winners in our world!”
The other teams spent the afternoon working on projects from the previous weeks, they wanted to ensure that everything they had worked on up to that point not only looked the way they wanted but also worked the way they wanted so they could showcase them to their parents.

*End of Day Questions.* Each week the girls were asked a question relating to the activity just completed, for instance after the story activity, one question invited the girls to reflect on the story they had just finished: “Think back to the story you worked on in Scratch involving the Octopus and her swimming area. Suppose you wanted the Octopus to invite more ocean friends to a party before she discovers the pollution. What steps would you follow to make this change? Don’t worry if you cannot remember all of the details of how Scratch works, but try to be as specific as you can.”

Some of the girls provided specific answers that included code examples, for example “You would not make Olivia go to the garbage, redo the broadcasts so the Starry say ‘wanna play...’. Make her broadcast something. All the other fishes receive it and they will go” or “I would create a new sprite & import a character then use the information block (control, look, motion) to put them in.” Others provided more general answers, such as “change the script add new character” or “You make new characters, then have them speak. Then have them react to Olivia’s illness. Then you can change the background.”

Another example comes from the week the girls worked with lists of data: “Describe what you did during today’s activity? What different Scratch blocks did you use to make it do that?” Answers tended to address the first question but not the second for example, “We solved problems to the Earth day thing. We used lists and all that data.”, or were even more vague, “Edit a game” and “Edit, language”. Given the low content of these end-of-day reflections, I concluded that a more direct and personal knowledge interview (i.e., a face-to-face interview) would provide better data for my final summative study (Chapter 5).
Artifacts. I spent considerable time examining the Scratch projects that the girls worked on over the course of GCI-2, especially because my classroom observations from the very first week indicated the girls were more fascinated with the pixel editor (not a focus in the CT learning) than the scaffolded example learning activities. Not surprisingly, the final artifacts produced through the story activity did reveal that all teams had worked in the pixel editor to enhance their character and even the background (Figure 13).

Figure 13 - Artistic changes to sprites and background

However I also noted that three of the four teams added new characters to the story; two teams expanded the storyline in ways that involved new Scratch code - including broadcasts - to enhance their stories. For example one team included new characters and changed the storyline to incorporate them, and also added new dialog for the main character (Figure 14). When Starry receives the sick broadcast from Olivia, Starry now states needing revenge on the humans and broadcasts ‘Dead human’ which triggers the shark sprite to eat the human that appears in the scene which gives the story a new ending (Figure 15).
Figure 14 - GCI-2 Storyline changes with new characters

Figure 15 - GC-2 Alternate story ending
During the games activity, the girls were not as distracted by the pixel editor. The game code is more complex than the story; in the presence of “broken” win and lose conditions, the girls concentrated more closely on finding and fixing the problems in the game code. One team of girls added an additional healthy item to be counted, working with a custom sprite that they created on their own, but also adding code for it to be a part of the gameplay (Figure 16).

Figure 16 - Custom sprite with added code

Another team had extraneous code in their final artifact; this suggests that these girls may have been confused about what the code was doing (or in this case not doing) when they worked on their corrections to the game.

One of the most complex computing artifacts created during GCI-2 was the Earth phrase translator. During this activity, none of the teams spent time creating new sprites or backgrounds. Two teams did add a background to their projects, but these came from the standard folder of Scratch background images. During the activity classroom observations seemed to reveal that all of the girls spent most of their time trying to determine what language to use, i.e., in lieu of the Norwegian language suggested by the workbook. My examination of the artifacts showed that two of the teams tried to add additional languages and phrases. Unfortunately in both cases the
teams encountered problems coding the new languages; this goal may have been overly ambitious given the 90-minute time frame.

The artifacts created for the Lego sensors and motors had no any extra characters or updates to the visual backgrounds. The girls were able to make the necessary code corrections to get their sensors and motors working. They directed considerable effort toward creating a holder for the tilt sensor and the square box to hold the motor for the music box. At first it appeared that using the physical Lego bricks would distract them from the programming needed to control the objects. However, perhaps because some of the girls were active in chorus and orchestra, there was a lot of interest to add the musical notes for the code to play a simple tune.

Interestingly, once sound was introduced as part of the Lego WeDo music box activity (week 5), a few teams returned to and modified some of their previous activities to incorporate sound. It was also interesting to note that problems that I encountered in the artifacts produced in the story, game and translator, were corrected during this last week of on your own time. Only one team decided to work on a new project, a two-player game (Figure 17). This provides good evidence that the projects were generally engaging to this population, and that the girls felt that they should be able to refine them to work as desired.
Figure 17 - GirlsChoice Week Two player game

4.2.5 Discussion

As in the GCI-1 workshop series the girls in GCI-2 were introduced to CT concepts with a scaffolded example and a corresponding workbook to guide them through the activities. Changes in the materials not only included removing the conceptual questions from the workbooks, but also adding end of day questions for the girls to answer. Instead of embedding conceptual questions on the CT concepts covered in each activity, end of day questions were administered.

In addressing the first research question, *Are girls this age able to grasp CT concepts and skills?*, as was done in GCI-1 the CT concepts of problem solving and abstraction were practiced in an interleaved fashion as the girls worked to understand and solve the problems that were posed. In each of the workshops the girls were able to complete the activities and in many cases
their customizations included a new storyline, additional complexity in their game code and in the translator activity there were attempts to add new phrases and languages which added additional conditions to consider.

From classroom observations I found there were times the teams were so distracted by the pixel editor that I wondered if the girls had grasped the material. However, as I analyzed the artifacts the girls completed I found that the girls did incorporate additional programming constructs, added new storylines that required new broadcasts and even included event handling instruction blocks.

To address RQ2, How did scaffolding help in learning CT concepts?, the change to a single platform was one of the largest indicators that the CT concepts were introduced in a manner which encouraged the girls without overwhelming them. The workbooks from GCI-1 were modified to remove the conceptual CT concept questions and have the workbooks become a guide for the scaffolded example. The workbooks had reduced areas for the girls to document project information as they worked on the activities, but they still contained areas that were left blank, especially towards the end of the activity. I found that as the girls became engaged with working on the scaffolded example in the Scratch environment the girls were comfortable enough with the programming environment that they concentrated more on adding customizations to their activities or trying out new instruction blocks.

Though the number of participants was lower for the GCI-2 workshop series, RQ3 Do learning activities introducing CT concepts lead to enhanced feelings of computer self-efficacy? was addressed by data collected in the pre and post surveys. The new computational self-efficacy scale appeared to be a step in the right direction, in that it had good internal reliability and the mean differences showed some indication of improvement. They girls also completed short answer questions related to a CT concept from one of the computing tools and their reactions to the activities in the workshop series (Appendix C). This survey assessed the girls’ interest in
attending advanced versions of the workshops (Alice, Scratch, web development); to what extent they felt successful with different aspects their projects (e.g. confidence in the computing activities and which tools); and whether they had fun using each of the tools during the workshops. In general, the girls’ reactions and attitudes were quite positive.

The classroom observations, project artifacts and survey results provided encouraging results that the CT concepts of involving algorithmic thinking of designing and implementing solutions to problems were being used, as well as moving between levels of abstraction were carrying over to workshops using different computing tools.

4.3 Summary of the Formative Research

The fall and spring workshop studies provided an opportunity for me to determine if the CT integrated computing activities for the middle school girls were engaging and motivating; and still be delivered in an after school format. The findings from both sessions reveal that the CT concepts associated with problem solving and abstraction that were integrated into the computing activities were appropriate for this age group and additional complexity can be considered when activities are created.

Using the various types of scaffolding (e.g. workbooks/journals, classroom assistants, pair programming, etc.) provided a learning environment that engaged and motivated the girls to work with different computing tools and at times even had them strive to do more than what was expected. However, it is also important that the learning environment provide some consistency in computing tools; given that the workshops were only 90 minutes, switching between 3D to 2D tools proved to be distracting for the girls in GCI-1. Thus I simplified my research design to use only the Scratch 2D programming environment. Findings from the use of only one computing tools during GCI-2 did show the girls were more engaged and motivated to try new instruction
blocks, increase the complexity of the computing artifact they produced in each workshop and also motivated them to make the projects their own with extra embellishments.

The results from the GCI-1 pre and post-workshop surveys pointed to changes needed to gather more useful data. From GCI-2 it was found that while the end of day questions provided better responses there was still a need for additional information that could be gathered if interviews are conducted at the end of each workshop. It may also be possible to minimize the IDK (I don’t know) notations or “don’t know” responses, though this may not be feasible with this age group. Removing the CT probes from the GCI-1 workbooks and reduction of the journaling areas in the GCI-2 workbooks did reduce the number of blank pages. However, since the blank pages occur later in the workbooks, a redesign of future workbooks with more instruction as the activity progresses is warranted instead of questions to be answered.

In each workshop series the activities were conducted in a connected series, though they were designed to be modular, such that they could be held as stand-alone units. This was shown to be possible when girls who missed a session due to fieldtrips or illness could catch up at a later date.
5 GirlsCreateIT! Summer Camp

The GCI-3 program for middle school girls was held during summer 2012 as a summer camp for four days; the sessions took place each morning for three hours. The program was offered to girls who would be entering grades 6-8 in Fall 2012 in the State College Area School district. As in the previous GCI workshops, I continued to plan for a group of 16 girls working in pairs; in this case only 15 girls enrolled for the summer camp and of those only 14 agreed to participate in the research project.

The general format for the workshops was similar to the GCI-2 workshop series, though the scaffolded example activities were updated to be consistent with a new theme (summer vacation). The pre- and post-workshop surveys used in GCI-2 were used without change; I modified the end-of-day questions so that I could use them as 1:1 interview questions. The daily sessions each had a scaffolded learning activities to introduce and build on CT concepts requiring minimal instruction in the first 90 minutes and the remaining 90 minutes became the on your own time, similar to the final session of the GCI-1 and GCI-2 workshops (Girls Choice).

5.1 Participants

The GCI-3 workshop series was created and delivered as part of the State College School District Summer 2012 Outreach Program. The program was advertised to girls who would be entering 6th, 7th or 8th grade in the fall. The half-day camp sessions were held on a Monday-Thursday from 9:00 am to 12:00 pm. Though the program was advertised to girls at all grades in middle school, it again attracted more 6th graders and 7th graders (43%), with only two of 8th graders (14%) (Figures 18 and 19).
The first day of the week I began with the choice-of-three candy icebreaker that had been used successfully in GCI-1 and GCI-2. After introductions, the girls were tasked with finding a partner, coming up with a team name and choosing a classroom computer station. One girl decided to work alone because she had attended GCI-1 and had also helped to teach Scratch with her mother as a part of a special school program. Two of the teams knew each other from the school district’s robotics camp at the beginning of the summer. After my introduction to the summer camp series of activities and the girls started work on the first scaffolded example, my assistant and I quickly realized that all but one of the teams were already comfortable with Scratch. Only one team of two 6th graders had no experience with Scratch prior to the camp. This was confirmed by their responses on the pre survey to the question “Have you ever done computer programming before? If so, what did you do?” one said no, and the other indicated she had worked on a computer on her Dad’s computer, but didn’t leave the tool she used.

Because GCI-3 was held in the towards the end of July only one of my prior assistants was able to help out in the classroom; she helped to set up the girls with USB drives that contained the scaffolded examples, copies of workbooks, answering general questions, and
conducting classroom observations. After training from me, she also conducted all but 2 of the interviews (I conducted the interview for one girl since she was a younger sister of the assistant.) Though the girls were comfortable using the school computers, I did spend some time on the first day instructing the girls on how to upload the file they would need for the day’s activity.

5.2 Procedures

I followed a session and activity scheme similar to that of the GCI-2 series, except that the sessions took place in four back-to-back mornings, and they were conducted in a 90+90 minute format. In the first half of each morning, the girls worked with scaffolded example activities isomorphic to the ones used in GCI-2 (see Appendix B); the second half of the morning was set aside for work on projects of their own choice. I told them at the start of the workshop that each day, they would spend the first 90 minutes on a daily activity using a pre-existing story or game, guided by a workbook. To keep them focused on the learning activity, I also told them the pixel editor would not be used until 10:30. They responded very well to this instruction, waiting to use the editor even when they had finished the scaffolded example activity before the 90 minutes were up. The on your own time allowed the girls to create their own projects, practice problem solving in Scratch, and test and debug new features. Each girl was also interviewed separately (in another room, for 10-15 minutes) during the second half of each morning (See Appendix D for the semi-structured interview guide we used).

On the first day, after pairing had been completed, each team chose a classroom station; the use of a school based lab and equipment was a convenience but it was only possible because we had simplified the activities to use a single platform (Scratch). While my assistant handed out workbooks, I gave a short overview of the upcoming activities for the week and briefly introduced them to Scratch. The workbook was used to assist the girls in exploring the Scratch
user interface and they also were guided to work with a sprite (the underlying concept for all Scratch characters); finally there was a short review of the overall workshop. As part of their Scratch exploration, all of the pairs encountered the built-in pixel editor and were reminded that it could be used during the *on your own time* part of the morning session. As a result everyone politely held off using the pixel editor and concentrated on the scaffolded activity. At the end of the first day my assistant and I showed the girls how to save their work on the USB drive (for safekeeping until they came back the next day), because the default in Scratch is to save all work on the computer drive (in this case a shared school computer). Once the girls handed in the workbooks and USB drives they began work of their own choice. At the end each morning the girls turned in their USB drives and could help themselves to a snack.

The next three days followed the same format as I used in GCI-2 but with the modification that each day had time for the girls to work on their own projects: I began each session with a high level overview of the day’s activity (10-5 minutes). As in GCI-2, I no longer needed to preview the computing tool to be used. Note that the inclusion of the *on your own time* session postponed the potentially distracting use of the pixel editor, and because the girls used it when creating their own projects, its use was more purposeful for them. As before, my assistant and I walked around the room to help girls who had questions or were struggling.

On the fourth day, the girls were guided to work with Lego bricks and the Lego WeDo sensors (See Appendix B). I helped the girls to construct their suitcases to work with the motors and build a bird to hold the tilt sensor. The extra butterfly scaffolded example was merged with the scaffolded activity for GCI-3.

The “Girls Choice” weeks that were a part of GCI-1 and GCI-2 were converted to 90 minute *on your own time* session during the second half of each morning during GCI-3; the girls were able to create new projects, work on previously covered material, or prepare for the parent showcase. As in the GCI-2 workshop series, the parent showcase did not include a large display
for all the girls to work on together. Instead the girls prepared one or more projects to demonstrate over the course of the week. One team created three complete projects and was working on a fourth by the parent showcase. None of the teams created a game, though the girl who had most experience in Scratch (she worked alone) began a scrolling game but was unable to build her planned win/lose elements by the end of the week. For the parent showcase, we highlighted each pair’s final version of the scaffolded example activities, so that parents could see what the girls had learned. After this ‘project show’ the teams then shared their other projects. Many girls used these showcase interactions as a sort of “mini Scratch lesson” for their parents.

5.3 Materials

The GCI-3 activities were similar to those in GCI-2; each session revolved around a scaffolded example activity designed to illustrate and exercise particular CT concepts or skills (see Table 13). Like GCI-2, all of the scaffolded examples were built using Scratch and the sensors and motors activity was supported by the Scratch integration with Lego WeDo. The workbooks were also isomorphic to the ones used in GCI-2, though they were updated to match the Vacation theme. In addition there were no modifications needed to the surveys used in GCI-2 so they were also used in this series of workshops.

The conceptual questions about the activity and CT concepts used as end of day questions in GCI-2 were revised into a format for use as in conducting individual interviews. The interviews were done after the girls finished the scaffolded activity and before they began the on your own segment of the workshop.

Classroom observations were noted down as time permitted by my assistant and me during each workshop. We experienced more difficulty handling this during the second half of
each day, as one of us was conducting the interviews in a separate room, leaving just one person
to help girls who ran into problems.

5.4 Results

Again I gathered a many types of data to study the girls’ understanding and application of
the CT concepts and the impacts of the learning activities on their attitudes about computing.
Here I report on the results from the survey data, the interviews, the workbooks and the
computing artifacts the girls created in the on the own time segment. I again collected the girls’
Scratch programs after each 90 minute learning activity, as well as anything they developed the
on your own time at the end of morning, so that I was able to use artifact analysis as another
converging method.

5.4.1 Pre- and Post-Workshop Surveys

Like for GCI-1 and GCI-2, I asked the girls to complete brief surveys both before and
after the workshop sessions. The form of these surveys was identical to the ones used in GCI-2,
because my analysis of the measures had indicated that they should be satisfactory and that they
might be good indicators of the girls’ feelings of success and general attitudes about computing.
Table 17 summarizes the pre-workshop ratings these girls provided concerning Career
Identification, Playfulness and CT Self-Efficacy. One surprising finding was that the internal
reliability of the Career Identification scale was rather low, even though the number of girls in
this group was greater than in GCI-2. I decided to use the construct regardless, because it had
been a reliable scale in both earlier workshops; however any specific conclusions tied to this scale
should be viewed as more speculative. (We also noted that when the girls responded to the same
three rating scales after the workshop, reliability was again high.) With respect to the other mean values, the Playfulness scale was higher for this group of girls, perhaps indicating some general motivational characteristics of girls who attend summer enrichment camps in technology domains. The CT self-efficacy scale had a value quite similar to what I observed pre-workshop for GCI-2. Thus even though some of these girls arrived with more experience using Scratch, it seems unlikely that they had been exposed to the CT concepts in focus for my work (many young people use Scratch simply to create fun figures and move them around in a background; this requires little if any CT skills).

Table 17 - Attitudes Assessed via the Pre-Workshop Survey in GCI-3

<table>
<thead>
<tr>
<th>Construct (n=14)</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Career Identification (α=.42)</td>
<td>4.88 (1.0)</td>
</tr>
<tr>
<td>Computer Playfulness (α=.81)</td>
<td>5.23 (1.2)</td>
</tr>
<tr>
<td>CT Self-Efficacy (α=.69)</td>
<td>4.35 (.86)</td>
</tr>
</tbody>
</table>

Given that the CT self-efficacy scale was a novel instrument, it is interesting to investigate its characteristics more carefully, preferably with a larger sample size. To do this, I combined the nine girls who completed this scale in GCI-2 with the 14 girls from the summer camp, giving me a larger sample of 23. The resulting means and standard deviations for each of the self-efficacy items appear in Table 18, broken down by the ratings provided before and after the workshops. As the individual means suggest, all of the ratings tended to increase after the workshop, although the ratings for the first item are virtually the same. Note that the largest mean difference is for the fourth item, which is the one that is most clearly related to what might be termed “programming”. Interestingly, this item is also the one that is least likely to co-vary with the other items in the scale; across all instances of the self-efficacy scale (individual workshops and the combined sample, both pre- and post-), this item has the lowest inter-item correlation.
Table 18 - Pre- and Post-Workshop CT Self-Efficacy Ratings, for combined sample from GCI-2 and GCI-3

<table>
<thead>
<tr>
<th>Item</th>
<th>Pre-Mean (SD)</th>
<th>Post-Mean (SD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I can start with an assignment and guidelines of what to include, and</td>
<td>5.32 (1.1)</td>
<td>5.38 (1.1)</td>
</tr>
<tr>
<td>come up with my own ideas on how to solve the problem, even if I’ve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>never worked on that type of problem.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Once I know what I need to solve a problem, I can decide which parts</td>
<td>4.41 (1.3)</td>
<td>5.24 (1.3)</td>
</tr>
<tr>
<td>can be solved with a computer, even before I work out a detailed plan</td>
<td></td>
<td></td>
</tr>
<tr>
<td>to solve the problem.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>If someone shows me a computer application, I will be able to</td>
<td>4.23 (1.3)</td>
<td>5.29 (1.2)</td>
</tr>
<tr>
<td>understand the words, commands, or images that I need for solving</td>
<td></td>
<td></td>
</tr>
<tr>
<td>my problem, even if it’s the first time I used the application.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>After I have created a plan for how to use the computer to solve a</td>
<td>3.55 (1.5)</td>
<td>4.76 (1.3)</td>
</tr>
<tr>
<td>problem, I can write down a set of instructions the computer can use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>in the solution, even if there is no one there to help me.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>If there are two possible ways to solve my problem with a computer,</td>
<td>4.32 (1.5)</td>
<td>5.14 (1.4)</td>
</tr>
<tr>
<td>I can talk about why one or the other is better, even if I have not</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tried out either one yet.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In the survey administered at the end of GCI-2, I collected the same measures investigated during GCI-2: confidence that the various projects had been completed in a quality fashion; the extent to which participants had fun completing the various projects; and repeat measurement of their identification with computer-based careers and CT self-efficacy. As in the earlier workshop, the internal reliability of these scales was adequate, ranging from .68 (fun doing the four projects) to .86 (career identification). All of the mean values were moderately positive, with the “fun” scale particularly high. Of note, all of these values tended to be more positive than the ratings from the girls in GCI-2, perhaps at least partially because of the changes in how the workshop was delivered (4 back-to-back mornings, half the time spent on their own), but perhaps also because some of the girls were already familiar with some aspects of the Scratch user interface and interaction techniques.

Paired t-tests on the two scales administered pre- and post-workshop revealed positive increases. For the change in Career ID, the difference was .92 (t(12)=2.94, p<.05); for self-
efficacy the corresponding difference was .85 (t(11)=3.26, p<.01). Thus, to the extent that the scales I used have construct validity; it seems that the scaffolded example activities succeeded in raising the girls’ general identification with careers in computing as well as their perceived self-efficacy with respect to CT concepts and skills as seen in Table 19. (Recall, however, that there was some doubt about the reliability of the career scale for this group of girls).

Table 19 - Attitudes Assessed via the Post-Workshop Survey in GCI-3

<table>
<thead>
<tr>
<th>Item Index</th>
<th>n=14</th>
<th>Mean (SD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Confidence (α=.77)</td>
<td>5.93 (.81)</td>
<td></td>
</tr>
<tr>
<td>Fun on Projects (α=.68)</td>
<td>6.11 (.78)</td>
<td></td>
</tr>
<tr>
<td>Post Career ID (α=.86)</td>
<td>5.77 (1.2)</td>
<td></td>
</tr>
<tr>
<td>CT Self-Efficacy (α=.80)</td>
<td>5.21 (.92)</td>
<td></td>
</tr>
</tbody>
</table>

In both the pre and post workshop surveys the girls were asked “We know you have a long time to decide this, but what career do you think you might pursue in the future?” The responses varied from technical, medical and artistic careers, for example three of the girls indicated they wanted to be vets, two stated they were interested in becoming dentists and there were three who listed pop star or musician as a career interest. Except for two, all of the post survey responses to the same question did not change. There was one girl who listed engineering as one of her career choices, along with dentist and vet in the pre – survey, but on the post-survey she refined her entry to a robotic engineer or computer engineer. Another girl added to her post survey response the career of a Lego designer. These entries show promise that the computing activities in the GCI-3 workshops inspired some of the girls to consider other careers as they learned new skills during the week.
On the first day, the girls began with a short activity aimed at familiarizing them with the Scratch environment and associated materials. This extra material included learning how sprites, scripts were used in Scratch Programs and also the available instruction blocks (i.e. motion, looks, variables, etc.). However, because we had only four days for the entire workshop, they also completed the first scaffolded example activity (story; see Appendix B). Examples of workbook pages can be found in Appendix E.

After entering the girls’ workbook notes into a spreadsheet, I again observed a pattern wherein the girls answered more question guides at the beginning of the activity; our classroom observations suggested that once they became comfortable with the day’s activity they simply relied less on the workbook. Nonetheless, the comments in the first day’s workbook did provide some insight into confusion surrounding the broadcasting concept covered in that activity (In Scratch broadcasting is used to send messages among characters). While none of the 14 participants had problems with the diagram illustrating how broadcast messages are used in the story (see Figure 52 in Appendix E), many were not able to answer the question “What sprite receives the getball message?” This more detailed inference proved difficult even though I had provided a hint: “look at the scripts listed for each sprite”.

In the fourth learning activity, the girls integrated the input from a tilt sensor with corresponding output in the form of a motor movement, using the Lego® WeDo framework. Answers to questions in the workbooks confirmed that most of the girls could distinguish between input and output when we connected the concepts to input from a computer mouse and output on a printer; however when examples were drawn from their Scratch activities on earlier days (for example, prompts or text displayed on the screen) the answers were much more varied.
This suggests that at least at this early phase, their concepts for input and output are somewhat narrow and tied to physical objects.

Other than the additional information and questions to guide the girls through the Scratch programming environment at the beginning for the story activity workbook, all the workbooks used the same format with information and questions to guide them through the CT concepts in the scaffolded examples, as was done in GCI-2.

5.4.3 Classroom observations

In general, my assistant and I observed that the girls were enthusiastic about working with Scratch. However, we also noted that their enthusiasm for hands-on activity at times led them to ignore the workbooks that had been designed to guide them through the story. For example we often found ourselves asking “Have you turned the page? You will be doing that in the next part of the exercise.” Early in the first day, one team did ask to add characters to the story; this was quite encouraging as it suggested that the girls were engaged enough in the activity that they wanted to do more. In fact, at times I felt like an intruder when I tried to ask the girls how they were doing or if they needed any help; the girls would not even turn around to look at me, just commenting “We’re okay” and continuing to work on the activity.

At the end of the first day I noticed that many of the teams were still having difficulty using the broadcasting concept, for example once they had finished the learning activity and had begun work on their on your own projects. As a result I decided to insert a short review session on broadcasting at the start of the next morning, fitting it into my introduction to that day’s scaffolded example activity (games). This short review appeared to help: my assistant noted in her classroom observations for that day that one of the teams had most of their characters broadcasting some type of message and had scripts working well in their story. She also noticed
that while most of the teams were actively creating new sprites and backgrounds in the pixel editor, they were also enjoying all the time provided for working on their own projects.

The data activity was more difficult for some of the girls. I observed on this day that some of the computers the girls were using did not have Internet access. Though GCI-2 and GC-3 were held in the same school classroom, we were fortunate that with fewer girls in the spring all of the computers were networked. In this case I moved the pair to the instructor’s computer; not surprisingly though they did not like being away from “their computer” to do the Google translations. However, at the end of the learning activity my assistant observed that all of the teams were able to work through this more complex activity, and in some cases even added interesting languages such as Hebrew and new vacation phrases like “Where is the restroom?”

Though there were two teams of girls who had worked with Lego Bricks in the robotics camp earlier that summer, all of the girls were at a loss about how to construct a bird structure around their tilt sensor. I had not built a structure around mine so they had no example to work from. When I quickly started one for a team, using just a few Lego bricks, the other teams were able to visualize what they could do with a few more Lego bricks. The construction of the suitcase was also a challenge for the girls, though in this case I had created a demonstration structure. I did ask one of the girls if she had worked with Legos before and her answer was “My brother is always playing with his.” I responded “Okay, we know your brother loves Legos but what about you? “ She answered, “Oh, I don’t know”. Overall, the girls seemed to enjoy seeing their motorized suitcases and tilt sensor ‘birds’ interact as part of the animated story.

Across the four days of GCI-3 I found that the girls very much enjoyed having 90 minutes to work on their own projects. By the 3rd day they were experimenting with sound, and had found they could not only modify existing sprites in the pixel editor but could also create new ones. Though they knew they had problems that needed fixing in the early days, most of the teams were eager to continue with their existing projects until they had them completed.
5.4.4 Interviews

Each day semi-structured interviews were conducted at the end of the first 90 minutes. The interview questions are listed in Appendix D. On the first day of GCI-3 my assistant and I split the interviewing between the two of us. However, soon the girls became more involved in trying new code elements and working hard to complete their on your own time projects; as a result my assistant conducted all of the remaining interviews except for two girls. One of these was her younger sister; the second was the more experienced girl who had attended GCI-1.

The responses to the first day’s interview question “Can you describe broadcasting?” added more detail to what we had observed in the classroom and confirmed with the workbook entries. The girls were able to complete the activity and answer most of the workbook questions, suggesting that they had at least a general understanding of broadcasting. When asked, most said it involved sending and receiving messages; some were also able to describe how broadcasting is used to control behaviors:

- P5: Broadcasting is like talking to all of the sprites.
- P14: like sending a message to all the sprites or characters
- P2: broadcasting is where the characters say something and then you have to make sure they all hear it so they go on to the next scene.
- P10: It’s basically when a character, it's a form of conversation, so like one character is like saying like if KitKat were to say to Billy go work that would be a broadcast
- P11: Broadcasting is like one of the characters sends a message to all the other characters that triggers them to do some other sequence of things

As other researchers have noted, learners often understand new concepts by relating it to familiar material (Carroll 1990):
P8: It's sending out a message that, it's a script that broadcast something and there's another script, when I receive and so it's like kind of when I say Hi you say Bye thing, like when I receive Hi you say bye

P7: Broadcasting is where someone is like speaking and like a news reporter broadcasts the weather every day and maybe every couple of hours.

The responses to broadcasting did indicate a level of understanding with their examples coming from their everyday world. In contrast with the concept of Sprite the girls were able to provide examples in their responses that related to the computer programming they did in the learning activity:

P1: a sprite is basically a character in Scratch ... it's kind of the person you're kind of programming

Interviewer: Can you give me an example?

P1: It's kind of like the cat on it, like the main one that comes with it. And there's a bunch of other ones

Classroom observations indicated that a review of broadcasting may be warranted, though their responses during the interviews indicated they could make sense of this concept, relating it to material in their worlds, it was important they could make the connection to the concept in a Scratch program. As a result I decided to do a short review of broadcasting at the beginning of the workshop on the next day.

The game activity introduced the CT concept of abstraction through its reliance on variables to control the game action. Thus we I probed the girls’ understanding of this concept with the interview question “You also used the variable itemCount today; what was it used for?” The girls’ responses suggested that they did understand the concept of a counter variable:

P1: itemCount is whenever you collect a certain amount of things in the maze today, KitKat she went out and collected all those items so whenever she would collect an item
what we did she would, oh whenever she would touch it, it would increase her score by
one saying that she got it and it would just hide

P2: to count how many items you had so you could go to the car at the end so you could
either win or just to show how many items you had and how many more you have to get

P12: to count up all the items you collected

The algorithm construct of iteration was also covered in this activity, so I asked the question “An
instruction block called forever was used in your scripts. What does it do?” The girls seemed to
have a basic understanding of iteration, though they could not articulate how the loop stopped:

P10: it works so that throughout the game KitKat would repeatedly get stung by the bee,
she would repeatedly go back to the start, it'd keep doing it, it wouldn't just be like once.
P12: that it will just keep repeating, it wouldn't really stop so like if one game ended and
if you clicked the green flag again it would keep on going

These interview data suggest that many of the girls were grasping the basic constructs of
variables and iteration, namely that a count of items was being maintained and that there was a
mechanism for doing things over and over. It is less clear that they understood how variables like
counts might be used to terminate a loop.

The third activity increased in complexity; the girls were guided to work with more
complex data structures – known as lists – as well as checking keyboard data for errors. The girls
understanding of these concepts was probed with the interview question “In Scratch how would
you create a list?” However, I found little evidence of a “data structure” abstraction of a list. For
example, some girls simply described the physical procedure they would follow to create a list:

P6: I would go to the variables and you can go to the make variable and the variable
would be make a list, and you could make a list

P2: you click the list button and typed in the name of the list and it came with some list
Other girls’ understanding was tied even more concretely to the specific (language translation) example they had been using:

*P7:* you create a list, gosh what was that thing called, you create a list by using your languages that you already have then you put your new language in and then there's this little tab thing where you have to put that language in there so you know if you put in another language you know how many languages you have.

It should be noted however that these responses were undoubtedly influenced by the fact that we had asked a “how” question rather than a “what” question. This is further evidenced by comments that mentioned very specific user interface actions needed to manipulate a list:

*P7:* you touch the, if you want to use a new phrase you're going to have to touch the circle that has a plus sign and type in the phrase and then you're going to have to put it in on every one of them and you're going to have to translate it.

*P9:* click the plus sign in the list box

*P10:* I would click the little plus button down in the corner and then type in what I needed

In general, these responses seem to indicate a rather concrete understanding of the list concept I had introduced that day, perhaps tied to the physical procedures that one uses in this environment for manipulating what is otherwise an abstract concept for data storage.

After the Lego sensors and motors activity the interviews suggested that at least some girls understood how their scripts were controlling the activity. In response to “How did your story know when to close the suitcase?” there was an indication that some of them had become comfortable the broadcasting mechanism and how it is used to control a story.

*P10:* when the items broadcasted close suitcase

*P12:* when Holly broadcasts a word and when it received the word it triggered the motor to close the suitcase
However, other girls shared a more literal understanding of the closing function, tying it simply to the story line in place:

*P5: when everything was picked up all the toys and things she needed to pick up it automatically closed*

*P14: when Holly collected all 5 items*

These girls seemed to miss the connection of what was happening in their script to what they observed in the physical structures, almost implying that the structures were responding directly to the “plot” of the story acted out by the script.

After this activity, we also asked the girls about their efforts to test and debug the activity they had worked on that day. All of the girls grasped the role of testing and fixing the project, with some detailed answers containing references to program constructs such as if statements and variables:

*P1: well first whenever we came in we used the sensor and it wouldn't move left to right because instead of having it controls the value 1, 2 3 and 4 for the ifs it was 1, 20, 3 and 40, so I figured out that it was 1, 2, 3, 4 instead of 1, 20, 3 and 40. We also had some problems where it was with the item count where we figured out it was six, we used 5 and the second to last item would close the suitcase.*

*P6: we made sure that the tilt sensing control thing went up by one instead of by 19 and 20 and then we had to make sure that the motor was not too fast and not too strong*

Over the course of the week I also noticed the girls were using the computing vocabulary in their responses more frequently after each of the daily activities; they were referring to the variables, scripts and sprites correctly when they gave examples in their responses. Discussing the concept of broadcasting on the first day evolved into responses later in the week that indicated
they had finally grasped the material and understood not only how to use a broadcast but knew what it did in their projects.

5.4.5 Artifacts

Classroom observations revealed considerable enthusiasm among the girls as they worked with Scratch and the Sprites. Some even added their own embellishments to the scaffolded examples, especially the first day’s story example. The computing artifacts from each of the learning activities were collected along with the on your own projects each day. As the learning activity artifacts were analyzed I found that the customizations to sprites, backgrounds, and addition of new sprites or backgrounds tapered off as the week progressed. This was seen especially with the final day’s sensor and motor activity. All of the teams followed the workbook and of the eight teams only one created an additional instruction block (a stop all) that was not a part of the activity. Also, it was found that 37% of the artifacts had code that should not have been there; in one case the problematic code produced a run-time error.

Each day of GCI-3 the girls had the opportunity to work on their own projects in the last 90 minutes of the day. This provided them more time to work on the learning activity (from that day or a previous day), or create something new. If desired they could continue with their new projects over the course of the week. The girls were encouraged to consider doing a story or a game. While no specific CT concepts were addressed in this portion of the day, the girls did need to do problem solving to work through the problems they encountered when new code was added, design of where the story was going was also needed and the projects also provided them an opportunity to use broadcasting, variables, iteration and event handling.

In my analysis of the on your own time projects I observed that the girls did enjoy using the pixel editor by creating or modifying existing sprites and backgrounds (Figure 20). As the
week progressed not only did the girls enhance their projects with custom sprites and backgrounds, but also they used broadcasting, iteration, changed the state of their characters with show/hide instruction blocks and added event handling where they could in their projects. Event handling was kept at a minimum with the teams’ only using keyboard input to bring in backgrounds or move their stories along.

Figure 20 - On your own project with custom sprites and backgrounds
All of the teams began the week working on stories when they had a choice; only one chose to work on a game on the final day. No teams chose to work with lists of data or to incorporate sensors or motors in their stories. The teams were able to use broadcasting on the very first day but there were some who didn’t understand what they were doing. As the girls used the concepts to move the story along and to bring characters into the scene they did begin making the connections of what broadcasting was doing in their computer programs. For example the story in Figure 21 uses a series of broadcast messages to bring new characters into the story.

Figure 21 - GCI-3 on your own time project with broadcasts
When the other characters receive the broadcast there is corresponding code in their scripts to include them in the scene. For example in Figure 22, when the “away.” broadcast is received there is an instruction block to show the character, have them move with a glide and has dialogue for them to say.

![Code Snippet](image)

**Figure 22 - GCI-3 Corresponding broadcasting receives**

In addition to broadcasting, instruction blocks that involve goto, say and think were used early in the week as the teams began setting up their stories. By the second day of GCI-3 all of the teams had added 2 – 4 sprites and 1 to 2 backgrounds to their stories. It was interesting how many projects the girls were able to complete during the week. Half of the teams worked on a single project during the entire week, but three teams completed two projects and one team worked on three and came close to finishing a fourth (Figure 23).
Figure 23 - Second on your own project created on 3rd day of GCI-3

Only one team chose to use the original story from the scaffolded example as their on your own time project. Over the week they included additional sprites, broadcasts and additional programming constructs to create their own version of the original story.

The sense of urgency to complete the learning activities observed during the first half of each day could be explained by the number of projects the teams completed by the end of GCI-3. At the beginning of my analysis I could see that some of the teams had problems that needed fixing in their projects but by the time the parent showcase began the projects were all working with no complications.
5.5 Discussion

In GCI-3 the girls were introduced to CT concepts with a scaffolded example and a corresponding work book to guide them through the activities. Except for a change of theme and including the introductory materials in the first workbook, the general format of the workbooks as a guide for the scaffolded examples did not change from GCI-2. Also the end-of-day questions from GCI-2 were modified into questions for interviews. The half-day allotted for each of the 4 days in GCI-3 provided a 90 minute session for a learning activity and a 90 minute session for applying the CT concepts covered in the first half of the day into their own projects. Classroom observations and the interview responses confirmed that the girls were excited with having the extra time to work on their ‘own’ project.

In addressing the first research question, *Are girls this age able to grasp CT concepts and skills?*, interleaving the CT concepts of problem solving and abstraction into the computing activities provided an opportunity for the girls to understand and solve the problems as they were presented each day. This was demonstrated as the girls were able to complete each of the daily activities and use the concepts they just learned into their own projects. It was also seen that while the girls did spend time with the pixel editor in the beginning of the week to create their own sprites and backgrounds, they created complex stories using broadcasting, iteration and event handling. Problem solving and abstractions were also evident when the girls projects were analyzed, problems they had in earlier versions were fixed before they demonstrated the projects to their parents and families.

To address RQ2, *How did scaffolding help in learning CT concepts?* the workbooks used with the scaffolded example provided clarification on the code constructs and the girls could use the examples as a point of reference. Since GCI-3 was held for 4 continuous days the girls there was less need for an introduction to the day. As in GCI-2 the workbooks were used less by the
girls as the project progresses. I found that as the girls became engaged with working on the scaffolded example in the Scratch environment the girls were comfortable enough with the programming environment that they often forgot that they had a workbook. Often we found ourselves reminding the girls their questions on what to do next could be found if they ‘turned the page’. During the on your own time portion of the day we did observe that the girls were able to include CT concepts they had used in the learning activity in their own projects. working with the CT concepts introduced in the activity. They included broadcasting, iteration and even worked with problem solving as they tested and debugged their activities to get them working by the parent showcase. .

To address RQ3, Do learning activities introducing CT concepts lead to enhanced feelings of careers in computing and computer self-efficacy? the same surveys used in GCI-2 were administered. The CT self-efficacy scale had a value quite similar to what I observed pre-workshop for GCI-2. Paired t-tests on the two scales administered pre- and post- revealed positive increases. For the change in Career ID, the difference was .92 (t(12)=2.94, p<.05); for self-efficacy the corresponding difference was .85 (t(11)=3.26, p<.01). The scaffolded examples used in the learning activities succeeded in raising the girls’ identification of careers in computing and their self-efficacy with respect to CT concepts.

Self-report data was collected before and after the program. For instance, I created a set of scales that assessed self-efficacy for computational problem solving, abstraction, debugging, and terminology. I measured these in advance and after the four activities had been completed. On a scale from 1-7 (where 1 is lowest efficacy), average ratings increased from 4.4 at the start to 5.2 after the workshop, t(11)=3.26, p<.01. I also assessed the extent to which the girls had fun in the program (6.1 out of 7), and whether they felt they had been successful (5.9 out of 7). All of these summary results are positive and consistent with our informal observations that girls were engaged and active throughout the program.
6 Prospects for Computational Thinking by Middle School Girls

My dissertation research has contributed to the computer science education community by providing new examples and assessment methods for computing activities that introduce CT concepts and skills to middle school girls. Similar activities that build on CT concepts are needed across the K-12 curricula, and the activities and lessons learned that I have described are already providing a set of new resources for teachers who might use them in their own teaching or after-school programs. The activities developed for use in this study will be available for teachers in K-12 to incorporate into curriculums or after-school programs.

Similarly, the learning activities and assessment methods that I have developed and refined over the GCI series provide an important starting ground for other researchers interested in CT by younger students. Whether the activities are used as is, or adapted and refined for other purposes, researchers like I will no longer need to start from the beginning. The CT self-efficacy scale and the mix of qualitative methods that I used to assess the process and impacts of the CT learning activities will also be useful to other researchers.

My research has investigated the CT concepts and skills comprehensible and usable by middle school girls. To do this I looked at computing tools used with this population and selected those which supported computing activities which provide an engaging and motivating learning environment.
### 6.1 Summary of Findings

In investigating RQ1: *Are girls this age able to grasp CT concepts and skills*, I articulated specific CT learning objectives and developed scaffolded examples and workbooks to convey these objectives. Over the course of the three GCI workshops, I found converging pieces of evidence to answer this question (Table 20). Evidence of the girls’ grasp of CT concepts and skills were found in their workbook entries, the artifacts created from the learning activity and their own newly created projects. By creating alternate solutions to the learning activities they also demonstrated that they could implement their own novel solutions.

<table>
<thead>
<tr>
<th>CT Objective and Activity</th>
<th>Evidence of CT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem-solving, analysis (storyline inaccuracy, broken games, missing code, non-working event handling)</td>
<td>Workbooks, modified scaffolded artifacts, new project artifacts, interviews</td>
</tr>
<tr>
<td>Problem-solving, design (creating a working solution, discussing and specifying how to solve the problem)</td>
<td>Workbooks, classroom observations, interviews</td>
</tr>
<tr>
<td>Algorithmic thinking (fixing the story or game, writing missing code)</td>
<td>Modified scaffolded artifacts, new projects</td>
</tr>
<tr>
<td>Software abstractions (broadcasting, variables, if/else, iteration)</td>
<td>Workbooks, activity artifacts, new projects</td>
</tr>
<tr>
<td>Understand computer vocabulary (sprites, variables, methods, broadcasting)</td>
<td>Workbooks, classroom observations, interviews</td>
</tr>
<tr>
<td>Testing and debugging the code (fixing the game, making the input and output work right)</td>
<td>Classroom observations, interviews</td>
</tr>
</tbody>
</table>

In addition, from the classroom observations of the girls interacting during the time they worked on their projects and from their responses to the CT probes during the interviews, I found that the girls were able to talk about programming and CT concepts in a natural way, through the computer vocabulary they used in communicating with each other and us.
The CT probes in the interviews, along with the artifacts the girls created, revealed that the girls understood what they needed to fix in their projects and could work with the varying levels of abstraction required by the problems. For example, the girls used lists of data for the language translator, they had to understand that data can be stored in a sequential fashion in a computer program and we have the ability to initialize the information in the list (set it up), we can change the data internally in the program by expanding the list or change existing data and we can create new lists of the data we want to use (new phrases or languages for Earth Day and vacations).

By relying extensively on scaffolding in the learning activities, I was able to investigate RQ2: *How did scaffolding help in learning CT concepts?* It was important that the scaffolding used would not only help to express the CT concepts being investigated in RQ1 but also provide just the right amount of assistance so as to not overwhelm or be of no use to the girls. Four example-based learning activities that were seeded with “fixable” problems, were designed in several genres: working with stories, games, a language translator and interactive structures using sensors and motors. Each computing activities had two important components, a scaffolded example that presented useful but incomplete code, and a workbook that was designed as a ‘minimal manual’ to guide the learners through the analysis and algorithmic thinking for implementation of changes that would improve the working example.

In each iteration of the study, the scaffolded activities changed. This was partly due to the chance of repeat participants, but also because I discovered in the first workshop that I should move from multiple computing platforms to a single platform that did not distract from the the CT concepts in the learning activity. This repeated development of the learning activities, all based on a scaffolded example with a minimal manual workbook, created four sets of activities that are isomorphic in style and learning goals. For the second and third workshop, the activities were almost identical except for the integrative theme. By concentrating on the CT concepts I was
investigating, I was able to create a set of computing activities that used different storylines and themes without affecting the CT concepts being covered, allowing me to transition smoothly from GCI-1 to GCI-2 and finally GCI-3.

Each learning activity revolves around a body of working code (ultimately in Scratch) and an accompanying workbook. Thinking more generally about these, I can offer a more generative scheme for others wishing to address similar CT learning objectives. Table 21 summarizes this more abstract view of four scaffolded example “genres”, where each could be instantiated according to an integrative theme such as I used in the GCI series (animal park, green world, vacation plans). If I were to continue the GCI series, I would be able to reuse these genre definitions to create new examples fitting into new themes; other researchers or educators could do the same.

<table>
<thead>
<tr>
<th>Genre</th>
<th>Features to instantiate</th>
</tr>
</thead>
<tbody>
<tr>
<td>The wacky story with an inaccurate storyline</td>
<td>Real world setting, interaction among characters, odd behavior by one or more characters</td>
</tr>
<tr>
<td>The broken game</td>
<td>Winning is based on a count variable, characters move and consume objects, control keys that are non-functional or poorly mapped</td>
</tr>
<tr>
<td>The list translator</td>
<td>Multiple lists that use position to cross reference but with incomplete look up, keyboard input and output</td>
</tr>
<tr>
<td>The interactive structure</td>
<td>Sensors and motors are controlled by code, but the object receiving the impact of the sensors or motors does not exist</td>
</tr>
</tbody>
</table>

It is important to note however, that in all cases the scaffolded example was accompanied by a custom workbook designed to serve as a minimal manual. The creation of these workbooks was very much determined by the nature of the example. For instance in the story, the first task that was scaffolded was analyzing what was wrong with how the character(s) were behaving; the workbook suggested this task and offered a semi-structured representation for doing this (e.g., a
question, a list or table for brainstorming). It is less clear what the abstract characteristics of the workbook should be; they borrow primarily from the body of work on minimalist instruction. However for this age group, and in this multi-pair workshop setting, they were a critical support mechanism.

The pre- and post-workshop surveys that evolved from GCI-1 to GCI-3 were used to investigate RQ3, Do learning activities introducing CT concepts lead to enhanced feelings of careers in computing and computer self-efficacy? In GCI-1 I found that some of the scales were not providing the results that I needed to completely address this RQ. Thus I made revisions to create a new CT self-efficacy scale that would better assess the self-perceived capacities of middle school level children, using vocabulary that related to the use of computers in their academic world. The career identification scales were also abbreviated to focus on only the items that seemed to make sense to this age group. I was not able to collect enough data to conduct a comprehensive evaluation of the statistical properties of these scales for this age group, but the initial results are quite promising and worthy of further study.

6.2 General Conclusions

As was discussed in Chapter 2, my research incorporating CT concepts and skills into computing activities drew upon other’s research conducted in the K-12 pipeline. Using the venue of after-school and summer enrichment programs I was able to extend the type of computing activities used in these settings by developing activities that concentrated on CT concepts and skills. My research draws upon the use of distributed scaffolds to create a learning environment to support the learner in acquiring new skill sets to problems that can be challenging in computing activities. By using distributed scaffolds in my research I was able to create a learning environment that provided different supports for the student. Scaffolding was used in the form of
a computing activity that used a workbook and scaffolded example to guide the girls to complete a computer artifact. Procedural, conceptual and social scaffolds were used as students worked in pairs on carefully constructed learning activities involving computing.

To design the computing activities, I researched different computing tools other researchers had used in similar settings and also drew upon their experiences with using stories, games and simulations to help guide my development of the computing activities for my research study. It was important the girls worked on activities relevant to them that engaged and motivated them. For this reason, the computing activities in each workshop session had a relevant theme and a consistent computing platform that supported the stories and games used in each workshop series.

I was also interested in learning about the girl’s attitudes towards computing, researchers from other studies have shown that girls at the middle school age level are beginning to notice which friends and family members are using technology. Girls this age are also looking for new subjects or electives to add to their school class schedule that are not only acceptable to their peers but also conducted in comfortable learning environments. I was able to create an after-school and summer workshop series where middle school aged girls could work with a friend on computing activities to increase their awareness of computer related careers and their self-efficacy. My original design of the computing activities was to introduce the girls to computer related careers, however this proved to be difficult to implement in the computing activities.

My research has added to the prior research work conducted in computer-related education programs. The overall research plan included creating computing activities that concentrated on CT concepts and skills, however the possibility of having repeat attendees in GCI-2 and GCI-3 led to isomorphic computing activities. These activities provide additional resources for other educators to use in their computer-related enrichment programs. In addition, the summative research study (GCI-3) succeeded in raising the girls’ general identification with
career in computing as well as their perceived self-efficacy with respect to CT concepts and skills. The community of computer science educators can use the activities and new CT self-efficacy scale produced in this research to conduct further studies in CT in the K-12 pipeline.

6.2.1 Computing Activities as Learning Modules

Each computing activity designed for this research was created to include the CT concepts of problem-solving, algorithmic thinking, abstraction and computing vocabulary for use as a stand-alone unit in classroom and enrichment programs. The scaffolded examples along with a corresponding workbook can be transformed into different stories and even be designed for use with other computer applications. Additional sub-activities may need to be added to the workbooks depending on the complexity of the problems and content that is introduced to the students. As the activities were developed for the GCI series of workshops I was able to incorporate different themes and levels of difficulty by concentrating on the CT concepts during the construction of each learning module of computing activities.

To construct a computing activity I started with a complete story of characters and dialogue that corresponded to the story. The complete story was then ‘broken’ to create problems for the students to solve. Each problem introduced into the story was analyzed to determine the level of assistance I wanted to include in the workbooks. It was important the level of help was balanced to provide the girls the assistance they might need but still allowed the girls opportunities for feeling a sense of accomplishment when they fixed all the problems and created their own working solution. Appendix E provides workbook examples of the different forms of assistance provided in the workbooks.
Table 22 - Process to Create a Learning Activity

<table>
<thead>
<tr>
<th>Activity Process</th>
<th>Considerations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td></td>
</tr>
<tr>
<td>• Create a story with characters and dialogue incorporating a central theme</td>
<td>Is there a subject or theme this activity needs to cover?</td>
</tr>
<tr>
<td>(e.g. Being Green, vacation summer vacation plans)</td>
<td></td>
</tr>
<tr>
<td><strong>Step 2</strong></td>
<td></td>
</tr>
<tr>
<td>• Create problems in the story for the students to solve.</td>
<td>What levels of difficulty need to be included in this activity</td>
</tr>
<tr>
<td>• Include problems with varying levels of difficulty to both encourage and</td>
<td>without distracting from the covered CT concepts?</td>
</tr>
<tr>
<td>challenge the students. For example, have characters omitted from scenes,</td>
<td>Are there opportunities for the students to create alternate solutions or</td>
</tr>
<tr>
<td>mismatched dialogue that doesn't fit the current storyline, loops not</td>
<td>to expand the original storyline?</td>
</tr>
<tr>
<td>working correctly, counters omitted</td>
<td></td>
</tr>
<tr>
<td><strong>Step 3</strong></td>
<td></td>
</tr>
<tr>
<td>• Construct a workbook with areas for students to record observations, ideas</td>
<td>CT concepts to include: algorithmic thinking, abstractions and computing</td>
</tr>
<tr>
<td>from brainstorming</td>
<td>vocabulary</td>
</tr>
<tr>
<td>• Include suggestions, tips, and code examples to convey the concepts being</td>
<td></td>
</tr>
<tr>
<td>covered in the sub-activities.</td>
<td></td>
</tr>
<tr>
<td><strong>Step 4</strong></td>
<td></td>
</tr>
<tr>
<td>• Work through each activity and sub-activity in workbook with scaffolded</td>
<td>Do the activities contain the correct amount of information and level of activity to cover the CT concepts?</td>
</tr>
<tr>
<td>example to ensure problems are being solved, abstractions apply and</td>
<td></td>
</tr>
<tr>
<td>implementations can be derived from concepts covered</td>
<td></td>
</tr>
</tbody>
</table>

The format of one and a half hours could be expanded for younger students or for those who computer skills are still developing. With each activity following a similar format that included problems to discover by finding what was and wasn’t working, documenting the findings and then working on possible solutions for the activity, there are areas in each activity that can be expanded to include additional content and be set up to work with other computer applications. It is recommended for younger students or those with less experience in working with computer applications that the time for completion be adjusted to allow the students more time to work on the problems. Though my participants did rush through some of the activities to
find a working solution to the computing activity, a teacher could provide additional learning modules that complement the activity as goals and sub-goals for the students to achieve.

Prior to each GCI session, my assistants and I worked through each learning activity to verify the scaffolded example and workbooks were consistent in covering the CT concepts and to correct any omissions. This served two purposes, one the learning activity could be double-checked and also the assistants would be familiar with the activity prior to our using it in the GCI program.

The success of the learning environment was also achieved by using different forms of scaffolds, for instance the girls worked in pairs to assist them in solving the problems and formulating solutions, providing support through a social scaffold. Assistants in the classroom not only aided in data collection but added another layer of support by answering questions posed by the girls and to encourage the girls to try new ideas as they worked to finish their projects.

The computing activities created for this research can be incorporated into different learning environments to not only cover subject matter content but also introduce students to computing in both formal and informal classroom settings. Each activity can introduce CT concepts in a format for one and a half hour workshops or be expanded to a longer workshop format.

6.3 Limitations and Future Work

Holding workshops as part of the existing outreach programs of the school district was clearly convenient for the girls and their parents; it also allowed the school district to expand their outreach program to include afterschool offerings with computing technology. However, the use of a naturalistic field setting did not provide the opportunity to conduct variations of the program with control groups, with controlled settings to observe the girls as they used the materials to
work on the activities. Thus the data that I was able to collect are necessarily complex and open to alternate explanations.

The school setting also meant that I could not install new software; it was for this reason that I initially brought in laptops for GCI-1. However, the laptops were a mix of Macs and PCs, making the girls’ interaction experiences more varied; they were also difficult to transport to and set up in the classrooms. The USB drives are easier to transport but they required the assistants to assist girls in uploading their files in the early days of the study and to also ensure each day the girls saved their work on the drives. Because the girls were using the school classroom computers it was unknown if other students had the ability to erase or move other students file.

While the CT probes in the interviews used in GCI-3 did provide evidence of the CT objectives, there were no additional interviews conducted after the girls finished the on your own projects. As a result, I may have missed important insights that they gained during these self-initiated activities. In future studies I will include additional interviews.

Though the study was conducted with middle school girls, the computing activities were developed to be used in studies that could be used with girls and boys. In the future I hope to use similar scaffolded computing activities as part of other enrichment programs or school coursework for boys and girls. I also envision an expanded set of activities, perhaps moving into more complex CT concepts. Additionally, the computing activities related to building webpages and graphic designs needs to be explored further; the one day in GCI-1 that explored this showed promising results, even though I had to move away from it once I decided to focus on Scratch.

In GCI-1 it was planned to include more computing career information in the activities. However, it was found that a carefully constructed scenario was needed to present the material in a manner that would not overwhelm the students as they worked on the computing activities. In addition the career information needs to be relevant for this age group and relate to the current activity. In the future I want to work on creating new scenarios that are directly related to
computing careers that will work with the CT concepts and also provide the students a rich learning environment.

6.4 Contributions and Conclusion

In my research I have articulated the learning of CT concepts and skills as a set of scaffolded computing activities that educators can use in school classrooms or as part of an outreach enrichment program. Teachers in the K-12 community can modify the learning activities while still focusing on the CT concepts. I plan to disseminate the materials through the Scratch website for educators, but also to continue my own work with educators working with CT.

This study also provided me an opportunity to create a CT self-efficacy scale. Although in a sense this contribution is small it is important, as the role of CT concepts and skills are being evaluated across the K-12 curriculum. While I believe that the scale I developed can benefit from more detailed analysis and refinement, it is a solid starting point for others trying to study this sort of CT experience, particularly if scaffolded activities similar to the ones I developed are in use.

My research has shown that it is feasible to focus on CT concepts in computing activities for afterschool workshops and summer camps, while still engaging and motivating the children in attendance to try new ideas and produce a computing artifact they are proud to show their parents and families. With computing technology always evolving the use of scaffolded examples and workbooks designed as minimal manuals are materials that can provide a gentle introduction for those who are novices with the computing tool while still providing challenges for those who are comfortable working with the tool.
7
References


## Appendix A: Comparison of Teaching Tools

Table 23 - Application Categorization (from Webb 2012)

<table>
<thead>
<tr>
<th>Computer Tools</th>
<th>Supported Platforms</th>
<th>Conceptual Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Stand Alone</td>
<td>Internet Based</td>
</tr>
<tr>
<td></td>
<td>PC</td>
<td>Mac</td>
</tr>
<tr>
<td>NetLogo 4.1</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Starlogo TNG</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>MicroWorlds</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Alice 2.0, 2.2</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Alice 3.0</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Scratch</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Squeak</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Story Telling Alice</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ToonTalk</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>ActiveWorlds Edu Universe</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Adobe Flash</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>StageCast Creator</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Agent Sheets</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Kudo</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Quest Atlantis</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>GapMinder</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>SecondLife</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Whyville</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>BrickLayer</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Pex</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>
Appendix B: Scaffolded Example Activities

Fall 2011 (GCI-1) Theme: Create an Animal Park Exhibit

1) Interactive Story - The Vet and the Wolf

**Tool used:** Alice 2.2

**Synopsis:** The activity involves a story about a Vet and a Mama wolf (Figure 17). The example has “mistakes” in the sequential sequence of the story fixed, dialogue corrected to match the character activities and the addition of variables that can be used to control the action sequence based on users’ input.

**CT Learning Objectives:** Problem solving, design and algorithmic thinking, learn and apply computing vocabulary

The activity requires the girls to break down a problem into smaller units, design a workable solution to each problem piece, turn these solutions into steps (an algorithm), and implement the solution to enhance the story (Figure 24 and Figure 25).

Programming-related learning objectives of methods, variables and decision logic were emphasized in this activity (Figure 26). The artifact created from the activities in this workshop shop was an interactive story that could be controlled by the user’s response to a question.
Figure 24 - GCI-1 Alice Vet Story

Figure 25 - GCI-1 Vet Story in Alice 2.2
2) Games– Fish Food Fun

**Tool used:** Scratch 1.4

**Synopsis:** The activities begin with a semi-functioning game that needs to be corrected. The game has a hungry fish that is trying to avoid eating bad fish food; if the fish eats enough good food the player wins the game. For instance, the example has no game counter to track and decide about a win, the keyboard controls are not set up correctly for full game control (only up/down work), including avoiding the barriers that are part of the maze.

**CT Goals:** Problem solving, design and algorithmic thinking, abstractions, use computing vocabulary

The CT concepts covered in this activity included problem solving to find the errors in the activity (Figure 27) and to form solutions, abstraction in the form of data variables and program constructs and working with event handling and user input to control game characters (Figure 28).
Figure 27 - Fish Food Fun Game - only bad food count showing

Figure 28 - Fish Food Fun- good variable count and code needed
3) Working with Data: Animal Database

**Tool used:** GirlsCreateIT Database (custom application created for the GCI-1 workshop)

**Synopsis:** This activity involves using a pre-existing online database of animals and their characteristics that can be viewed, queried, and extended, with query results displayed as part of a webpage. The database contains data on animal habitats, weight and diet.

**CT Goals:** Problem solving, design and algorithmic thinking, use computing vocabulary

Problem solving is used to determine the information needed to formulate simple database queries (Figure 20). Abstraction in the form of variables is used to work with the tables to data and generate the results that are presented as part of a webpage (Figure 21). Computing vocabulary associated with databases is covered to assist in completing the activity tasks.

![GirlsCreateIT Database Interface](image)

*Figure 29 - Drag and Drop interface to create queries*
4) Sensors and Motors: Interactive Structure for an Animal Park

**Tool used:** Lego RCX w/Robolab software

**Synopsis:** In this activity the girls work on a fictitious animal park display, where interactive structures are needed to help young visitors learn more about the animals in the park. These structures are created using the Lego RCX unit (Figure 31), a sensor (touch/light) and additional art supplies (opportunity to be creative). A working example program is provided as a starting point, once it is downloaded from the Robolab application (Figure 32) to the RCX unit.

**CT Goals:** Problem solving, design and algorithmic thinking, input/output, testing and debugging

The CT concepts included novel input and output abstractions (sensors, motors) and more complex solution processes needed to combine the different hardware components into a single computational artifact. Problem solving occurred in the testing and debugging of the implemented interactive structure artifact.
Spring 2012 (GCI-2): Protect the Environment

1) Interactive Story – Olivia’s Story

Tool used: Scratch 1.4
**Synopsis:** This activity had two scenes; one with friends on the boardwalk who became concerned about the sea creatures after someone threw garbage into the ocean, and a second where the Olivia the octopus is curious about the strange food but not avoiding it (Figure 33). Olivia goes up to the strange food but acts as if nothing is wrong. The story changes included having Olivia get sick from the pollution and leaving the swimming hole for a cleaner location.

**CT Goals:** Problem solving, design and algorithmic thinking, use computing vocabulary

The CT concepts covered in this activity involved working on analyzing and solving the problem of the dialog not matching the story line. Algorithmic thinking was used to design and implement the solutions for the event handling in the program through the use of broadcast messages sent and received by the sprites in the story (Figure 34), as well as learning the computing vocabulary used by the programming environment (i.e. sprites, scripts, broadcasting).

![Figure 33 - Olivia Story dialog problems](image-url)
Figure 34 - Broadcasts used in friends’ scene

2) Games—Ocean Fun

Tool used: Scratch 1.4

Synopsis: The example game presents two bad food items for the Starry character to avoid and when encountered would add to the bad counter variable. There was one good food item for Starry to eat, which would then increment the good food counter. A lose background would display when the bad counter was greater than 10. If the good food item was encountered the game was to increment a good counter variable and then display the win background when the good count was greater the girls would choose. The Starry character was controlled by the user using the arrow keys (Up, down, right and left). This activity needed the addition of the good counter as well as an IF statement to display a winner’s background that is also created by the participants. Only two arrow keys were working, two others needed to be programmed to work.

CT Goals: Problem solving, design and algorithmic thinking, abstractions, use computing vocabulary
The CT concepts covered in this activity included problem solving to find the errors in the activity and form solutions (Figure 35), abstraction in the form of data variables and program constructs and working with event handling and user input to control game characters (Figure 28).

Figure 35 - Ocean Fun with missing variable

3) Working with Data: Earth Day Messages

**Tool used:** Scratch 1.4

**Synopsis:** This activity involved building a simple language “translator” that could be used to introduce the concept of related information stored in a set of different lists. There were two different pre-existing lists of Earth Day messages, one related to tree phrases and the other related to earth phrases. Each of these phrases was already translated into French and Spanish. The translator activity provided opportunities for the participants to work with user input (Figure 36) create new lists and add to existing lists.
**CT Goals:** Problem solving, design and algorithmic thinking, abstractions, use computing vocabulary

The CT concepts covered in this activity included problem solving to find what was or was not working in the activity (Figure 37), working with data abstraction in the form of lists and learning the vocabulary associated with list processing, such as initialization (also known as housekeeping).

![Image](EarthDayMessagesFun.png)

*Figure 36 - Earth Day Messages Screen*
4) Sensors and Motors: Music Box

**Tool used:** Lego WeDo w/Scratch 1.4

**Synopsis:** An interactive display was created in this activity using Scratch and the Lego WeDo sensors and motors. To complete this activity the students first became familiar with the hardware they needed for this activity by using the tilt sensor to control a single sprite (butterfly). The second part of the activity involved working with an example centered on an animated story that would trigger an external motor for a music box figure to spin around and music to play in the Scratch project; the girls completed the story to work in an effective way.

**CT Goals:** Problem solving, design and algorithmic thinking, input/output, testing and debugging

The CT concepts from the previous weeks’ sessions are again re-visited in the sense of analyzing a problem and designing an improvement to an existing situation (Figure 38); new concepts include novel input and output abstractions (sensors, motors) (Figure 39), and the more
complex solution process need to combine the different components into a single computational artifact.

![Scratch Animation](image)

**Figure 38 - Scratch Animation missing code**

![Lego’s WeDo Tilt Sensor, motor and USB hub](image)

**Figure 39 - Lego’s WeDo Tilt Sensor, motor and USB hub**

### Summer 2012 (GCI-3): Vacation Fun

1) **Interactive Story – Vacation Planning**

**Tool used:** Scratch 1.4

**Synopsis:** This activity was an animated story of four friends making plans for their vacation. It also included introductory material on sprites, scripts and related code blocks (e.g., motion, looks, variables, etc.). During their meeting the friends would each make a suggestion and the
backgrounds would change as each friend described their vacation idea. Modifications to the story included replacing or adding instruction blocks to control character movement, what the characters said in the scenes and controlling when they should be visible or hidden.

**CT Goals:** Problem solving, design and algorithmic thinking, use computing vocabulary

The CT concepts covered in this activity included learning programming vocabulary and working with event handling in a program through the use of broadcast messages sent and received by the sprites in the story so they were included in the scenes and part of the story (Figure 40). Problem solving was needed to determine when characters should hide and show (Figure 41).

![Vacation Plans story with missing character](image-url)

*Figure 40 - Vacation Plans story with missing character*
2) Games – Vacation Maze

**Tool used:** Scratch 1.4

**Synopsis:** This activity was an interactive maze that controlled a character (KitKat), using arrow keys to move around the yellow barriers to ‘pick up’ vacation items to pack in a waiting car. If at least 3 items were picked up and the time was under 60 seconds the player wins. The game had several embedded problems that the girls worked to correct: 1) the cat was able to move through the yellow barriers; 2) there were items that the cat could not collect (i.e. didn’t increase the count and did not disappear when encountered); and 3) the “lose” message did not display when the time was up.

**CT Goals:** Problem solving, design and algorithmic thinking, abstractions, use computing vocabulary

The CT concepts covered in this activity included problem solving to find what was/was not working in the activity (Figure 42), using algorithmic thinking to design solutions that were
implement for the given problems and learning the computing vocabulary associated with games. The activity also included using abstractions with the user of variables and keyboard input from the user to control the direction of the character (Figure 43).

![Figure 42 - Vacation Maze with missing code](image)

![Figure 43 - Vacation Maze User Input Code](image)
3) Working with Data: Vacation Translator

**Tool used:** Scratch 1.4

**Synopsis:** This activity involved refinement of a simple language “translator” that could be used to introduce the concept of related information stored in a set of different lists. The activity began with five pre-existing lists of phrases. There was a list of phrases a traveler may use during a vacation in English, translated in French and also Spanish. In addition there was a list of each character’s known languages. The translator activity provided opportunities for the participants to work with user input (Figure 35) create new lists and add to existing lists.

**CT Goals:** Problem solving, design and algorithmic thinking, abstractions, use computing vocabulary

The CT concepts covered in this activity included problem solving to find what was or was not working in the activity, working with data abstraction in the form of lists and user input (Figure 44) and learning the vocabulary associated with list processing, such as initialization (Figure 45).

![Figure 44-Working with Data Activity - Language Translator](image-url)
4) Sensors and Motors: Vacation Packing

**Tool used:** Lego WeDo w/ Scratch 1.4

**Synopsis:** This activity was a story with the character (Holly) who is moved around the room (Lego® tilt sensor) to collect her vacation items and place in her suitcase. Within the animated story the suitcase closed when all the items were packed. In addition to the tilt sensor an external suitcase was built using Lego® bricks and a motor; which would also close at the appropriate time (vacation items were all packed in the animation).

**CT Goals:** Problem solving, design and algorithmic thinking, input/output, testing and debugging

The CT concepts from the previous weeks’ sessions are again re-visited in the sense of analyzing a problem and designing an improvement to an existing situation (Figure 46); new concepts include novel input and output abstractions (sensors, motors) (Figure 39), and the more complex solution process need to combine the different components into a single computational artifact.
Figure 46 - Vacation Packing Tilt Sensor Code Errors
Appendix C: Participant Surveys

PARTICIPANT BACKGROUND SURVEY – Fall 2011

Thank you for coming to GirlsCreateIT! We hope that our activities will help you see how technology relates to the world around you. In the following, we are gathering background information. This means that your answers are feedback for us, not a test of your skills.

1. Please circle your current grade level: 6th 7th 8th What is your age? ________

2. We would like to know about your current interest in different careers. For each career in the table below, circle how interesting you think the career is in general.

<table>
<thead>
<tr>
<th>Career</th>
<th>Not at all Interesting!</th>
<th>Maybe</th>
<th>Very Interesting!</th>
</tr>
</thead>
<tbody>
<tr>
<td>doctor</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>nurse</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>engineer</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>school teacher</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>vet</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>fashion designer</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>scientist</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>computer programmer</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

3. From the list of careers above, which ones do you think use computer technology and how?

4. We know you have a long time to decide this, but what career do you think you might pursue in the future? (This doesn’t have to be from the list above)

5. The next few items ask you to think about CAREERS that involve the use of computers. Circle a number to show your response to each question.

<table>
<thead>
<tr>
<th>Expectation</th>
<th>Not at all</th>
<th></th>
<th>Maybe</th>
<th></th>
<th>Most definitely</th>
</tr>
</thead>
<tbody>
<tr>
<td>I understand what computer programmers do</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>
6. The next set of items ask about PEOPLE who use computers in their jobs. Circle a number to show your response to each question.

<table>
<thead>
<tr>
<th>People who choose careers that use computers are...</th>
<th>Not at all</th>
<th>Maybe</th>
<th>Most definitely</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geeks</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hard-working</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Smart</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Loners or anti-social</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Creative</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

7. Now, think about YOURSELF, how you FEEL when you do activities with a computer.

<table>
<thead>
<tr>
<th>When I work on computer projects I often feel...</th>
<th>Not at all</th>
<th>Maybe</th>
<th>Most definitely</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spontaneous</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Limited in my thoughts</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flexible</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Creative</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Playful</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uninventive</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bored</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

8. Last, we would like to understand what you expect when you try to use new computer tools. The tool could be anything, maybe to create pictures, web pages, do math, etc. Imagine using a new tool as you answer the questions. Circle a number to show your response to each question.

<table>
<thead>
<tr>
<th>I could use the new computer tool even if...</th>
<th>Not at all</th>
<th>Maybe</th>
<th>Most definitely</th>
</tr>
</thead>
<tbody>
<tr>
<td>There was no one around to tell</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
me what to do as I go.

I had never used a tool like it before.  

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

I had just the built-in help information for assistance.

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

I had seen someone else using it before trying it myself.

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

I had a lot of time to complete the activity the tool is for.

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

I could call someone for help if I got stuck.

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

PARTICIPANT POST WORK SURVEY (Fall 2011)

Thank you for coming to Girls CreateIT! We hope that your work with us helped you better understand how technology relates to the world around you and in many different careers. In the following questions, we are asking you to help us just a bit more, to give us ideas about how to improve the activities in this program. This means that your answers are feedback for us, not a test of your skills.

A. Workshop Follow-up Questions

Please circle the number that best represents your response to each question.

<table>
<thead>
<tr>
<th>I would be interested in attending an advanced workshop in...</th>
<th>Not at all</th>
<th>Maybe</th>
<th>Absolutely</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storytelling using Alice</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Storytelling using Scratch</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Games using Scratch</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Games using a new computer tool</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Creating my own Web Page</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2. We would like to know how you feel about the different projects you worked on during the workshops

<table>
<thead>
<tr>
<th></th>
<th>Not at all</th>
<th>Maybe</th>
<th>Absolutely</th>
</tr>
</thead>
<tbody>
<tr>
<td>I am confident that I understood the computer tools and other materials used for the activities.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I am confident that I understood the problem descriptions that were used to introduce the activities.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I am confident that I developed good Lego exhibits.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I am confident that I developed good stories.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I am confident I developed good games.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3. How much fun did you have using Alice to work with stories?

1   2   3   4   5   6   7
Miserable  Neutral  Lots of fun

Please explain this answer.

4. How much fun did you have using Scratch to work with games?

1   2   3   4   5   6   7
Miserable  Neutral  Lots of fun

Please explain this answer.

5. How much fun did you have using the database and website tools?
6. How much fun did you have working on the Lego exhibits?

Miserable Neutral Lots of fun

Please explain this answer.

7. Think back to the story you worked on in Alice involving the Vet and the wolf family. Suppose you wanted the Vet gently coax the Mama Wolf and pups to follow her to an ambulance for care. What steps would you follow to make this change? Don't worry if you cannot remember details of how Alice works, but try to be as specific as you can.

8. Think back to the game you worked on in Scratch involving the big Fish and the good/bad food it was eating involving. Suppose you wanted to give some medicine to the Fish could to make the bad food count reset to zero. What steps would you follow to make this change? Don't worry if you do not remember details of how Scratch works, but try to be as specific as you can.

9. Think back to the Web page you worked on with the GirlsCreateIT! database of animals. Suppose you wanted to display information about animals that live in the savannah but are also small animals? What steps would you follow to do this? Don't worry if you cannot remember details of how the web application works, but try to be as specific as you can.

10. Think back to the interactive display you worked on with the Lego Systems involving use of sensors and a bush or tree. Suppose you have downloaded your program to the RCX and the bush won't move when you use the light sensor. What steps would you follow to solve the problem? Don't worry if you cannot remember details of how Robolab or the RCX works, but try to be as specific as you can.
11. Finally, we would like to understand what you expect when you first try to use a new computer tool like you did in this workshop (i.e., Alice, Scratch, the web database, Lego Systems). The tool could be anything, maybe to create pictures, web pages, do math, etc. Imagine using a new tool as you answer the questions. Circle a number to show your response to each question.

<table>
<thead>
<tr>
<th>I could use the new computer tool even if…</th>
<th>Not at all</th>
<th>Maybe</th>
<th>Most definitely</th>
</tr>
</thead>
<tbody>
<tr>
<td>There was no one around to tell me what to do as I go.</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I had never used a tool like it before.</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I had just the built-in help information for assistance.</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I had seen someone else using it before trying it myself.</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I had a lot of time to complete the activity the tool is for.</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I could call someone for help if I got stuck.</td>
<td>1 2 3 4 5 6 7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

12. We are interested in any other reactions you have too – is there anything else you would like to share with the team?

PARTICIPANT BACKGROUND SURVEY (Spring 2012/Summer 2012)

Thank you for coming to GirlsCreateIT! We hope that our activities will help you see how technology relates to the world around you. In the following, we are gathering background information. This means that your answers are feedback for us, not a test of your skills.

5. Please circle your current grade level: 6th 7th 8th What is your age?

__________
In these workshops you will be doing different kinds of “programming” using computers. Don’t worry if you’ve never done this before, you’ll get plenty of help!

6. What do you think it means to do “computer programming”?

7. Have you ever done computer programming before? If so, what did you do?

8. Besides people whose job is to be a computer programmer, what other jobs do you think might use computers and how?

9. We know you have a long time to decide this, but what career do you think you might pursue in the future?

6. The next few items ask you to think about possible CAREERS for you involving the use of computers. Circle a number to show your response to each question.

<table>
<thead>
<tr>
<th>Expectation</th>
<th>Not at all</th>
<th>Maybe</th>
<th>Most definitely</th>
</tr>
</thead>
<tbody>
<tr>
<td>I understand what computer programmers do</td>
<td></td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>If I choose to, I have the ability to do a job that uses computers.</td>
<td></td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>My parents would probably consider a computer-related job to be a good option for me.</td>
<td></td>
<td>4</td>
<td>7</td>
</tr>
</tbody>
</table>

7. The next items ask about how YOU FEEL when you do activities with a computer

<table>
<thead>
<tr>
<th>When I work on computer projects I often feel…</th>
<th>Not at all</th>
<th>Maybe</th>
<th>Most definitely</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spontaneous</td>
<td></td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td>Limited in my thoughts</td>
<td></td>
<td>4</td>
<td>7</td>
</tr>
</tbody>
</table>
For the next few items, we want you to think about how someone might solve problems with computers. This could be for fun, like making a game or writing a story, or for school projects, like setting up a science experiment.

8. I can start with an assignment and guidelines of what to include, and come up with my own ideas on how to solve the problem, even if I’ve never worked on that type of problem.

   1 234567
   Not at all Maybe Most definitely

9. Once I know what I need to solve a problem, I can decide which parts can be solved with a computer, even before I work out a detailed plan to solve the problem.

   1 234567
   Not at all Maybe Most definitely

10. If someone shows me a computer application, I will be able to understand the words, commands, or images that I need for solving my problem, even if it’s the first time I used the application.

    1 234567
    Not at all Maybe Most definitely

11. After I have created a plan for how to use the computer to solve a problem, I can write down a set of instructions the computer can use in the solution, even if no one is there to help me.

    1 234567
    Not at all Maybe Most definitely

12. If there are two possible ways to solve my problem with a computer, I can talk about why one or the other is better, even if I have not tried out either one yet.

    1 234567
    Not at all Maybe Most definitely
POST-WORKSHOP SURVEY (Spring 2011/Summer 2012)

Thank you for coming to Girls CreateIT! We hope that your work with us helped you better understand how technology relates to the world around you and in many different careers. In the following questions, we are asking you to help us just a bit more, to give us ideas about how to improve the activities in this program. This means that your answers are feedback for us, not a test of your skills.

A. Workshop Follow-up Questions
Please circle the number that best represents your response to each question.

1. We would like to know how you feel about the different projects you worked on during the workshops

<table>
<thead>
<tr>
<th></th>
<th>Not at all</th>
<th>Maybe</th>
<th>Absolutely</th>
</tr>
</thead>
<tbody>
<tr>
<td>I am confident that I understood the computer tools and other materials used for the activities.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>I am confident that I understood the problem descriptions that were used to introduce the activities.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>I am confident that I developed good Lego exhibits.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>I am confident that I developed good stories.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>I am confident I developed good games.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

2. How much fun did you have using Scratch to work with stories?

1 2 3 4 5 6 7
Miserable Neutral Lots of fun

Please explain this answer.

11. How much fun did you have using Scratch to work with games?

1 2 3 4 5 6 7
Miserable Neutral Lots of fun
Please explain this answer.

12. How much fun did you have using Scratch to work with the list objects?

1 2 3 4 5 6
Miserable Neutral Lots of fun

Please explain this answer.

13. How much fun did you have working on the Lego exhibits?

1 2 3 4 5 6
Miserable Neutral Lots of fun

Please explain this answer.

For the next few items, we want you to think about how someone might solve problems with computers. This could be for fun, like making a game or writing a story, or for school projects, like setting up a science experiment.

14. I can start with an assignment and guidelines of what to include, and come up with my own ideas on how to solve the problem, even if I’ve never worked on that type of problem.

1 2 3 4 5 6
Not at all Maybe Most definitely

15. Once I know what I need to solve a problem, I can decide which parts can be solved with a computer, even before I work out a detailed plan to solve the problem.

1 2 3 4 5 6
Not at all Maybe Most definitely

16. If someone shows me a computer application, I will be able to understand the words, commands, or images that I need for solving my problem, even if it’s the first time I used the application.

1 2 3 4 5 6
Not at all Maybe Most definitely

17. After I have created a plan for how to use the computer to solve a problem, I can write down a set of instructions the computer can use in the solution, even if no one is there to help me.
18. If there are two possible ways to solve my problem with a computer, I can talk about why one or the other is better, even if I have not tried out either one yet.

1 2 3 4 5 6 7
Not at all Maybe Most definitely

11. The next few items ask you to think about possible CAREERS for you involving the use of computers. Circle a number to show your response to each question.

<table>
<thead>
<tr>
<th>Expectation</th>
<th>Not at all</th>
<th>Maybe</th>
<th>Most Definitely</th>
</tr>
</thead>
<tbody>
<tr>
<td>I understand what computer programmers do</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>If I choose to, I have the ability to do a job that uses computers.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>My parents would probably consider a computer-related job to be a good option for me.</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

12. In the future we may offer advanced workshops and would like to know your interest level in the type of workshop you would consider attending.

<table>
<thead>
<tr>
<th>I would be interested in attending an advanced workshop in…</th>
<th>Not at all</th>
<th>Maybe</th>
<th>Most Definitely</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storytelling using Alice</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Storytelling using Scratch</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Games using Scratch</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Games using a new computer tool</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Creating my own Web Page</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

In these workshops you did different kinds of “programming” using computers.

13. What do you think it means to do “computer programming”?
14. Besides people whose job is to be a computer programmer, what other jobs do you think might use computers and how?

15. We know you have a long time to decide this, but what career do you think you might pursue in the future?
Appendix D: Interview Questions (GCI-3)

Each set of questions were posed to each participant (individually) at the end of a workshop activity, prior to them beginning the work on their own (i.e., one set per day).

I) Interview Questions for Story Activity:

Summarize: Today you worked on a story about friends planning a vacation. You learned how sprites (characters) talk to each other and how scripts of instruction blocks are used to tell a story. You also worked on getting the story to make more sense.

1. Suppose you had a new character you wanted to add to the story called Holly. Though Holly is late for the meeting she is eager to go with all of her friends. She is even okay with any location the group has already chosen. How would you bring her into the scene?
2. Can you describe what broadcasting is? Can you give me an example of using broadcasting with Holly?
3. Can you describe in your own words what a sprite is/does? Can you give me an example?
4. Can you describe in your own words what a script is? Can you give me an example?

2) Interview Questions for Game Activity:

Summarize: Today you worked on a game with KitKat being challenged to bag her vacation items that are scattered in a maze. Her friends want to see if she can pick up all the items in less than a minute while avoiding being stung by a bumble bee. You tested the game first to find problems needing correcting and then worked on adding a new item to be packed.

1. Suppose you wanted to have KitKat find her vacation items in two different mazes. She would need to find the beach ball and sunglasses in the second maze, not the first. What all would you need to do to make this change? (May need to prompt about a new maze background, having the items displaying correctly and also increasing the timer value)
   a. What do you need to have a second maze? What would be a good idea to get from one maze to the other?
   b. How would you display the items in the second maze but not the first?
3. How would you increase the timer?
2. How does the game currently know when to stop?
3. You also used the variable `itemCount` today; what was it used for? What did you need to do to get it working in the scripts?
4. You used something called an if statement today. What does it mean to have an if instruction block in your script? How does it work?
5. An instruction block called forever was used in your scripts. What does it do?

3) Interview Questions for Data Activity:

Summarize: Today you worked on a project to translate travel phrases. You used lists to organize your phrases.
1. Suppose you find out you will be going to Portugal on vacation. What would you need to do to add Portuguese to your set of translations?
2. You worked on error handling today. What error was occurring at the beginning of the project? How did you handle the error?
3. What do you need to do to create and then use a list in Scratch?
4. How would you add new information into a list?

4) Interview Questions for Lego Activity:

Summarize: Today you worked with sensors and a motor with an animated story. With the tilt sensor you had Holly the bird collect items to pack in her open suitcase. After all items were packed a motor was used to close the suitcase.
1. How did your story know when to close the suitcase? What would happen if the bird failed to collect one of the items? Why?
2. Suppose instead of the story closing the suitcase you wanted to close it yourself. How could you change the Scratch code to let you close the suitcase?
3. During this past week you tested a lot of different parts of stories and games. What did you do to test and debug the “Pack the suitcase” activity?
Appendix E: Workbook Excerpts

GCI-1 Interactive Stories - Alice 2.2 Vet and Wolf Story

Figure 47 - GCI-1 Interactive Story Workbook Problem Solving Page

Figure 47 is an excerpted page from the GCI-1 Interactive Story Workbook which shows the problem solving table provided as a guide to assist the girls in organizing their ideas, what methods they can use from the scaffolded example and also a column to document how their solutions worked when they tested it out.
Figure 48 is another table that was used in the interactive story activity in GCI-1 to assist the girls work with the code complexity of the `if` statement. While similar to the page illustrated in Figure 42, this table was provided to assist the girls in handling the Yes and No paths of the decision logic they were using.
GCI-2 Data Activity – Earth Day Messages

Activity 3: Error Handling

Before we go further we want to take care of problems that might occur. If things have been going correctly for you and you’ve typed in French, Spanish or English your project should be working just fine. But have you tried a language other than those three? Or did you accidently misspell the language? What happened? What did you do to fix it?

1. Right now if the language is not in our list then the program will continue to process the request. The character actually thinks “I know that language!” even though he doesn’t… that doesn’t make any sense.
   Let’s fix it!

2. We’ll need a control block to keep repeating the question and if block code if the language doesn’t exist in our list. Find the correct code block to correct this problem.

3. Test your project by using a language that we don’t have in the list. Did your fix work?

Figure 49 is a page excerpt from the GCI-2 Working with Data Workshop, it shows an example of the minimalist guided exploration prompts to guide them through the activities.

GCI-2  Working with sensors and motors activity
Activity 2: Using the Tilt Sensor and a motor

8. From your plan, you will need to add similar code blocks to the other toys to complete the Toy clean up scripts.

9. What does the repeat until block do? (Hint: right block it and choose help to find out)

10. Try changing the wait time. What happens when you increase the seconds?

Figure 50 - GCI-2 Sensors and Motor Workbook Analysis Page

Figure 50 provides an example from GCI-2 sensors and motor activity using a guided analysis of the code to review CT concepts the girls had encountered in previous workshop activities.
GCI-3 Story Activity - Vacation Plans

Activity 2: Instruction blocks

<table>
<thead>
<tr>
<th>Instruction Block Type</th>
<th>Color</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Motion</td>
<td>Blue</td>
<td>move 10 steps</td>
</tr>
<tr>
<td>Control</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Looks</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sound</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pen</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sensing</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operators</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Variables</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 51 is an excerpt from the story activity from GCI-3, since there was no introductory week for the girls to explore the environment on their own. I came up with this introductory material to assist the girls in learning more about the Scratch environment at the beginning of the activity on the first day.
Figure 52 is an excerpt from the GCI-3 Story Activity Workbook using analysis of the project code to assist in the learning of the Scratch concept of broadcasting. This was an important exercise that prepared the girls for working with the sprites and backgrounds to activate many of the actions needed in the story and game activities they would be doing.
GCI-3 Game Activity - Vacation Maze

Activity 1: Exploring the Vacation Maze Game

The game Vacation Maze has our friend Kittat needing to collect her vacation items to pack her car. Her friends don’t think she can pickup the items in 1 minute. Since Kittat wants to get on the road as soon as possible, she has accepted their challenge to show them how fast she can move through the maze. Unfortunately her items are all over the place and a bumblebee is buzzing trying to sting her (which causes her to go back to her start position). The objective of the game is for Kittat to move through the maze to collect her items and get to the car before 1 minute has passed. When she has done that she will win the challenge.

At this point the maze game needs some work to be done and before you make any changes you need to do some testing and debugging. Find the Vacation Maze game that is on your USB drive and open it. You will be using the arrow keys to move Kittat around. Select the green flag and run the game.

1. To help you in your testing and debugging use the following table as you analyze the game. Once you have played the game a few times you will want to record what you observed.

<table>
<thead>
<tr>
<th>Game Elements</th>
<th>Questions to consider</th>
<th>Write down what you noticed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yellow Barrier</td>
<td>Are they working?</td>
<td></td>
</tr>
<tr>
<td>Item Count</td>
<td>Who uses it? Are all of the points being given?</td>
<td></td>
</tr>
<tr>
<td>Win Message</td>
<td>Is there one? Should there be one? If there is one what makes it display?</td>
<td></td>
</tr>
<tr>
<td>Loss Message</td>
<td>Is there one? Should there be one? If there is one what makes it display?</td>
<td></td>
</tr>
</tbody>
</table>

Figure 53 - GCI-3 Game Activity Test and Debug Workbook Page

Figure 53 is an excerpt from the GCI-3 Game activity Workbook, showing the table used for testing and debugging of the game as a part of the problem solving process so the girls could determine what wasn’t working and what they needed to fix.
GCI-1 Game Activity Workbook CT Probes

Activity 5: Win and Lose Messages

It is always nice to know if you have won or lost a game. So let’s get this working!

1. We can switch backgrounds when our game is over. Select the Stage and the Backgrounds tab. You will see there are two additional backgrounds available. One is labeled Win and the other Lose.

2. Select the Scripts tab and find the script for change-message. When ‘change-message’ is received then all the scripts are going to stop.

Question: Which Sprite is broadcasting this message and why are they broadcasting this message?

3. We can use this message to help us display the right message when the game ends. Let’s take a few minutes to brainstorm this problem!

Question: What can we do to stop the game with a nice message screen? List your ideas here!

Figure 54 - GCI-1 Game Activity CT Probes

Figure 54 is an excerpt from the GCI-1 Game activity workbook illustrating how CT probes were embedded in the workbook activities.
Game Activity Go End of Day Questions

1. What did you need to use in the FishFoodFun Game that you also needed for the Wolf and Vet Story?

2. What did you enjoy doing today when you worked on the game?

3. What didn’t you enjoy doing today when you worked on the game?

4. If you had the time today what do you wish we had done today?

5. What careers did we talk about today?

Figure 55 - GCI-1 Game Activity End of Day CT Probe

Figure 55 is one example of the End-of-Day questions that were a part of the actual workbooks in GCI-1. In this example there is also a CT probe referring back to the previous week’s activity to determine if the girls had understood that they had now used variables in two different activities.
Appendix F: GCI-2 End of Day Questions

Interactive stories

In the following questions, we are asking you to help us just a bit more, to give us ideas about how to improve the activities in this program. This means that your answers are feedback for us, not a test of your skills.

1. Think back to the story you worked on in Scratch involving the Octopus and her swimming area. Suppose you wanted the Octopus to invite more ocean friends to a party before she discovers the pollution. What steps would you follow to make this change? Don't worry if you cannot remember all of the details of how Scratch works, but try to be as specific as you can.

2. We would like to know how you feel about what was covered today.

<table>
<thead>
<tr>
<th>Activity</th>
<th>I don’t know what this means</th>
<th>I don’t know how to do this</th>
<th>I may know how to do this</th>
<th>I’m okay doing this</th>
<th>I can do this well</th>
<th>I may be able to show someone how to do this</th>
<th>I can teach someone how to do this. I’m an expert</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write a script so a character will move and talk in a Scratch story</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Run a story</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Brainstorm to create new ideas for a story in Scratch</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Use the instructional blocks to turn my ideas into a story</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>
Games Activity

In the following questions, we are asking you to help us just a bit more, to give us ideas about how to improve the activities in this program. This means that your answers are feedback for us, not a test of your skills.

1. Think back to the game you worked on in Scratch involving the Starfish trying to reach the good food while avoiding the garbage in the water. Suppose you wanted to provide some bonus medicine that would reset the sick variable to zero. What steps would you follow to make this change? Don't worry if you do not remember details of how Scratch works, but try to be as specific as you can.

2. We would like to know how you feel about what was covered today.

<table>
<thead>
<tr>
<th>Activity</th>
<th>I don’t know what this means</th>
<th>I don’t know how to do this</th>
<th>I may know how to do this</th>
<th>I’m okay doing this</th>
<th>I can do this well</th>
<th>I may be able to show someone how to do this</th>
<th>I can teach someone how to do this. I’m an expert</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test a game to find out what to fix</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Debug a game</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Create a variable to store information</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Increase a variables value by 2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Use the if instruction block to make decisions</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>Brainstorm new game ideas</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>
Data Activity

In the following questions, we are asking you to help us just a bit more, to give us ideas about how to improve the activities in this program. This means that your answers are feedback for us, not a test of your skills.

1. Describe what you did during today’s activity? What different Scratch blocks did you use to make it do that?

2. Was there time today when you clicked the green flag and your Scratch project didn’t work like you wanted it to? If so, how did you find out what was wrong? What did you do to change it? Is that what you usually do?

2. We would like to know how you feel about what was covered today.

<table>
<thead>
<tr>
<th></th>
<th>I don’t know what this means</th>
<th>I don’t know how to do this</th>
<th>I may know how to do this</th>
<th>I’m okay doing this</th>
<th>I can do this well</th>
<th>I may be able to show someone how to do this</th>
<th>I can teach someone how to do this</th>
<th>I’m an expert</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialize a list</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Add another item to a list</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Create a new list</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>
END OF DAY QUESTIONNAIRE – Interactive Structures

In the following questions, we are asking you to help us just a bit more, to give us ideas about how to improve the activities in this program. This means that your answers are feedback for us, not a test of your skills.

1. Think back to the interactive display you worked on with the Lego Systems today involving use of sensors and music box. Suppose your tilt sensor is not moving correctly and the motor doesn’t turn on. What would you do to find out what the problem(s) might be? Don't worry if you cannot remember details of the Scratch code or how the Lego’s work, but try to be as specific as you can.

2. We would like to know how you feel about what was covered today.

<table>
<thead>
<tr>
<th>Task</th>
<th>I don't know what this means</th>
<th>I don't know how to do this</th>
<th>I may know how to do this</th>
<th>I’m okay doing this</th>
<th>I can do this well</th>
<th>I may be able to show someone how to do this</th>
<th>I can teach someone how to do this</th>
<th>I’m an expert</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set up a tilt sensor to work with a Scratch Sprite</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Debug an animation</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Use a list to keep track of items</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Use a counter variable to start a motor</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Set up a loop to repeat until something happens</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>
Curriculum Vita
Heidi Cornelia Webb

EDUCATION

Doctor of Philosophy (Ph.D.) - Information Sciences and Technology (IST)
The Pennsylvania State University, University Park, PA. August 2013

Master of Science in Computer Science
St. Mary’s University, San Antonio, TX, 1998

Bachelor of Science in Computer Science
Texas State University, San Marcos, TX, 1994

REFEREED CONFERENCE PROCEEDINGS


BOOKS


