

Teaching Computer Science Principles using StarLogoTNG

by

Tamika P. Tannis

B.S. Computer Science and Engineering
Massachusetts Institute of Technology, 2011

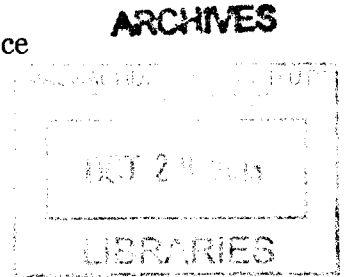
Submitted to the Department of Electrical Engineering and Computer Science

In Partial Fulfillment of the Requirements for the Degree of
Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

August 2013
[SEPTEMBER 2013]
© 2013 Tamika P. Tannis. All rights reserved.



The author hereby grants to MIT permission to reproduce and to distribute publicly paper and electronic copies of this thesis document in whole or in part of any medium now known or hereafter created.

Signature of Author: _____
Department of Electrical Engineering and Computer Science
August 23rd, 2013

Certified by: _____
Eric Klopfer
Professor of Science Education and Engineering Systems
Director, Scheller Teacher Education Program
Thesis Supervisor

Accepted by: _____
Professor Albert R. Meyer
Chairman, Masters of Engineering Thesis Committee

Teaching Computer Science Principles using StarLogoTNG

by

Tamika P. Tannis

Submitted to the Department of Electrical Engineering and Computer Science

In Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Electrical Engineering and Computer Science

August 2013

ABSTRACT

This thesis outlines the development of a 3-module set of lesson plans implemented using StarLogoTNG. The purpose of these lesson plans are to serve as a vehicle for teaching and reinforcing specific learning objectives of the CollegeBoard's Advanced Placement Computer Science Principles course, which has 7 main themes. Each lesson plan has as its focus a subset of learning objectives from one of the themes of Creativity, Data, or Internet, while simultaneously incorporating additional learning goals from the themes of Abstraction, Programming, Algorithms, and Impact. These interactive lesson plans go beyond the use of StarLogoTNG to complete specific tasks by integrating meaningful class discussions and occasional peer instruction and peer review activities. Such activities become catalysts for students to develop a deeper understanding of the course materials. By connecting learning goals from different themes of the course and packaging them in cohesive lesson plans that utilize methods of teaching for understanding, this thesis aims to provide a useful and effective set of materials for the instruction of computer science principles.

Thesis Supervisor: Eric Klopfer

Title: Professor of Science Education and Engineering Systems

Acknowledgements

I would first like to thank my thesis supervisor, Eric Klopfer, for allowing me the opportunity to work on this project with MIT STEP, and for his willingness to provide me with the resources I needed, whether it be ideas, information, suggestions, or funds for my workshops. I'd also like to thank Daniel Wendel who was a constant source of feedback and inspiration for new ideas when I found myself at any form of technical or creative roadblock, and who was always available for brainstorming sessions. A thank you goes to Wendy Huang for also making herself available to give me feedback on my work, for providing me with resources for research, and for playing a major part in helping me get attendees for my workshops. Lastly to everyone in the MIT STEP Lab. Working in an atmosphere of fun, friendly, and creative minds, being able to see what other projects were going on, and listening in on the interesting lab lunch discussions introduced me to many unknown ideas, concepts, and perspectives within the fields of learning and educational technology.

I want to acknowledge the sources of inspiration that set me on the path to wanting to pursue a career in computer science. To Mr. Richard Nixon who was my introduction to computer science and taught me programming for several years at the NYU Science and Technology Entry Program, and to Mr. Michael Zamansky who taught me computer science at Stuyvesant High School during the 10th and 11th grades. Thank you for being such amazing and inspirational teachers. Teachers such as you need to be present in schools across the country so that more children can be inspired to pursue computing related careers.

Most importantly, I'd like to thank my family. To my sister for giving me a reason to excel in school, even if that reason was just that I wanted to match or even beat your scores on standardized tests. To my brother for being part of my tests phases, even if it was because Mommy and Daddy made you do it. And lastly, I'd like to thank my parents and my grandmother for raising me, and for the endless and priceless ways they have contributed to my academic success.

Table of Contents

1. Introduction.....	13
2. Background.....	15
2.1 Computer Science Principles.....	15
2.2 StarLogoTNG.....	16
2.3 Teaching CS Principles using StarLogoTNG.....	18
3. Learning Frameworks.....	21
3.1 Acquire-Meaning-Transfer.....	21
3.2 Peer Instruction.....	22
3.3 Peer Review.....	24
4. Design Approaches.....	25
4.1 ADDIE Model.....	25
4.2 Concept Activation.....	27
4.3 Connecting Computing – Impact.....	29
5. Creativity Module.....	30
5.1 Creativity: Analysis.....	31
5.2 Creativity: Design.....	36
5.2.1 Concept Activation & Impact.....	36
5.2.2 Game Description Narrative.....	37
5.2.3 Character Design Process.....	38
5.2.4 Functionality Classification.....	38
5.2.5 Algorithm Design.....	40
5.2.6 Programming & Testing.....	40
5.2.7 Applying Peer & Instructor Review.....	41
5.2.8 Assessing Creativity using Quality Indicators.....	42
5.3 Creativity: Development.....	43
5.4 Creativity: Implementation.....	45
6. Data Module.....	48
6.1 Data: Analysis.....	48
6.2 Data: Design.....	51
6.2.1 Concept Activation.....	51

6.2.2	Simulation & Activity Theme.....	52
6.2.3	Applying AMT & Peer Instruction.....	53
6.2.4	Impact.....	53
6.3	Data: Development.....	55
6.4	Data: Implementation.....	58
6.4.1	Data Module Implementation Phase 1.....	58
6.4.2	Data Module Implementation Phase 2.....	59
7.	Internet Module.....	61
7.1	Internet: Analysis.....	61
7.2	Internet: Design.....	63
7.2.1	Concept Activation.....	63
7.2.2	DNS Lookup Activities.....	64
7.2.3	Applying AMT.....	65
7.2.4	Impact.....	65
7.3	Internet: Development.....	66
7.4	Internet: Implementation.....	68
8.	Results, Evaluation, and Future Work.....	69
8.1	Creativity.....	69
8.1.1	Results.....	69
8.1.2	Evaluation.....	71
8.1.3	Future Work.....	72
8.2	Data.....	74
8.2.1	Phase 1 – Results.....	74
8.2.2	Phase 1 – Evaluation.....	74
8.2.3	Phase 2 – Results.....	75
8.2.4	Phase 2 – Evaluation.....	77
8.2.5	Future Work.....	77
8.3	Internet.....	78
8.3.1	Future Work.....	78
9.	Conclusion.....	79

List of Figures

Figure 1: An example of code in the StarLogoBlocks window (left) and corresponding interactive SpaceLand environment (right)	17
Figure 2: A sample workspace of Eclipse, a very popular integrated development environment.....	19
Figure 3: Diagram of the ADDIE model.....	26
Figure 4: An example of how to code and test procedures that execute certain actions that only happen when a specific event occurs in the game.....	39
Figure 5: Buttons and monitors that students interact with during the data analysis activity.....	56
Figure 6: An example of a graph that students may add to the model using the information they are given in an attempt to plot data that clearly shows the distinctions between turtles with different Gene B variations living in an infected environment.....	57
Figure 7: A message is forwarded to a website using it's IP Address.....	67
Figure 8: The home computer (diamond) is requesting from the DNS server (large cube) an IP Address of a website (small cubes). All requests are transferred through the router. (cylinder)	68
Figure 9: A diagram that shows that running the simulation with 100 turtles leads to graphs that aren't very conclusive. These side-by-side comparisons of the results of the same graph generated on two separate occasions using the same parameters do not contribute to students finding a clear trend.....	75

List of Tables

Table 1: A table showing the learning goals of Creativity selected for the Creativity module.....	31
Table 2: A table showing the learning goals of Algorithms selected for the Creativity module.....	33
Table 3: A table showing the learning goals of Programming selected for the Creativity module.....	34
Table 4: A table showing the learning goals of Impact selected for the Creativity module.....	35
Table 5: A table showing the relative experience and ages of test subjects for the Creativity module.....	45
Table 6: A table showing the learning goals of Abstraction selected for the Data module.....	48
Table 7: A table showing the learning goals of Impact selected for the Data module....	49
Table 8: A table showing the learning goals of Data selected for the Data module.....	50
Table 9: A table showing the learning goals of Internet selected for the Internet module.....	61
Table 10: A table showing the learning goal of Impact selected for the Internet module.....	63

1 INTRODUCTION

The unemployment rate in the United States has been one of the largest socio-economic issues in the country during the 21st century. Between 2008 and 2009 in the United States the unemployment rate of non-institutional civilians ages 16 years and older skyrocketed from 5.8% to 9.3%; the following year it hit 9.6% (“Employment Status, 1940s to Date”). This was the highest unemployment had been since the recession of the early 1980’s, and has only been surpassed in the recent century by the unemployment rates that occurred post World War I and during the Great Depression. Such a rise in the unemployment rate was correlated with the decrease in job opportunities due to the recession that occurred between 2007 and 2009. However, while job opportunities within certain fields have indeed continued on a decline in the 21st century, jobs within the Science, Technology, Engineering and Math (STEM) fields have been on a steady and impressive increase. In 2010, STEM workers represented 1 out of every 18 employed workers. Furthermore, by the year 2018 occupations in STEM fields are estimated to grow by 17%, which far surpasses the projected 9.8% growth rate of all other non-STEM fields (Beede, Doms, et al. 2011).

For many young American students these statistics suggest the coming of a golden opportunity within their future to fill these jobs and lower future unemployment rates, especially if these students pursue a computing related career. Jobs in computing are expected to make up 71% of these projected STEM jobs. In fact, it is predicted that by 2016 the need for computer scientists alone would increase as much as 53% (Cuny 2010). However, American universities are disappointingly only turning out enough computer science graduates to fill about 33% of these projected job openings (Cuny 2010). The promising chance for the American youth to fill the need for skilled workers in various computing fields is en route to being missed by many.

These potential missed opportunities are the result of an education system that fails to motivate students to pursue computing and other STEM related fields. A report published by a college planning service that conducts yearly research on the interests of high school students revealed a startling statistic:

“Nearly 28% of high school freshmen declare interest in a STEM-related field—around 1,000,000 students—each year. Of these students, over 57% will lose

interest in STEM by the time they graduate from high school”

(“Where are the STEM Students?”).

As a result of the insufficient interest of our nation’s youth in pursuing STEM related careers—specifically in computing—the National Science Foundation (NSF) has initiated the Computer Education for the 21st Century (CE21) program, which aims to reverse the declining curiosity of middle and high school students in computing careers. As part of this CE21 program, the NSF has provided funding to the College Board to aid in the development of a new Advanced Placement Computer Science course. This course, referred to as Advanced Placement Computer Science: Principles (AP CSP), is in its pilot stages with the goal of being an official AP Course by 2016 or later. This course differs from previous AP Computer Science courses in the way that it is designed to stray away from a curriculum centered strictly on code, algorithms, language syntax, and data structures—which can be intimidating and off putting for many students. Rather, the main goal of this course is to expose students to the central ideas of computing, which go far beyond programming and algorithms. This course will involve activities that encourage creativity and highlight how computer science applications affect the world. It is expected that this approach towards teaching computer science concepts will appeal to a greater audience than current computer science curriculums do, and increase the number of high school students who become interested in computing related careers.

The project detailed in this paper aims to use StarLogoTNG, a visual programming software created by the MIT Teacher Education Program, to develop curriculum activities that will cover specific learning goals of the AP CSP course. In this paper I will detail the approaches taken to create a set of 3 independent modules that help teach and reinforce subsets of learning objectives from all 7 Big Ideas of the AP CSP course, using StarLogoTNG as a vehicle. Throughout the paper I will touch upon the pedagogical approaches and methods used to guide the design and development of the modules, discuss the learning objectives of each module, and evaluate their effectiveness and potential usefulness in the classroom.

2 BACKGROUND

2.1 Computer Science Principles

The curriculum framework for the AP Computer Science Principles course has been evolving since its inception in Fall 2010. This section gives an overview of the framework of the course as it existed as of August 2012.¹ The AP CSP course is centered on 7 “Big Ideas”:

1. *Creativity: Computing is a creative activity.*
2. *Abstraction: Abstraction reduces information and detail to facilitate focus on relevant concepts.*
3. *Data: Data and information facilitate the creation of knowledge.*
4. *Algorithms: Algorithms are used to develop and express solutions to computational problems.*
5. *Programming: Programming enables problem solving, human expression, and creation of knowledge.*
6. *Internet: The Internet pervades modern computing.*
7. *Impact: Computing has global impacts.*

Each of these themes has specific key concepts and learning objectives that serve to “introduce students to the central ideas of computer science”, “install ideas and practices of computational thinking”, and “have students engage in activities that show how computing changes the world” (“AP Computer Science Principles” 2013). These learning objectives are also designed to have students engage in the following 6 “Computational Thinking Practices”:

1. *Connecting Computing*
2. *Developing computational artifacts*
3. *Abstracting*
4. *Analyzing problems and artifacts*

¹This project was completed with the August 2011 and August 2012 frameworks as a guide. All references to the curriculum framework in this paper and in the documentation of the developed project materials align with the August 2012 curriculum framework, unless otherwise stated.

² Unlike in an academia and research setting where the process of peer review can also used to judge the legitimacy of a piece of work, in a classroom setting peer review will not used in these materials to deem whether or not a student’s work is “acceptable” or “good enough”. Such

5. Communicating

6. Collaboration

Each learning objective of the 7 Big Ideas is matched with one of these 6 computational thinking practices, resulting in these practices being constantly reinforced and exercised throughout the course.

The intent behind a course with such a structure is that it will not only help students learn and practice computational thinking skills—something that the current American education system fails to do—but also allow students to learn, understand, and experience how important computing is in our present day society. Students will learn basic computing concepts and discover the incredibly broad spectrum of computer science applications. As a result, the AP CSP course has the potential to lead to an increase in students that develop and maintain an interest in pursuing computing-related careers.

2.2 StarLogoTNG

StarLogoTNG is a visual programming language in which the programming is not done through text, but through StarLogoBlocks that mimic puzzle pieces and represent pieces of code. StarLogoBlocks connect together like puzzle pieces to form programs that are executed in the SpaceLand environment, a 3-dimensional graphical environment that “inherits many of its characteristics from video games” (Klopfer et al. 2009), making the software very appealing to students.

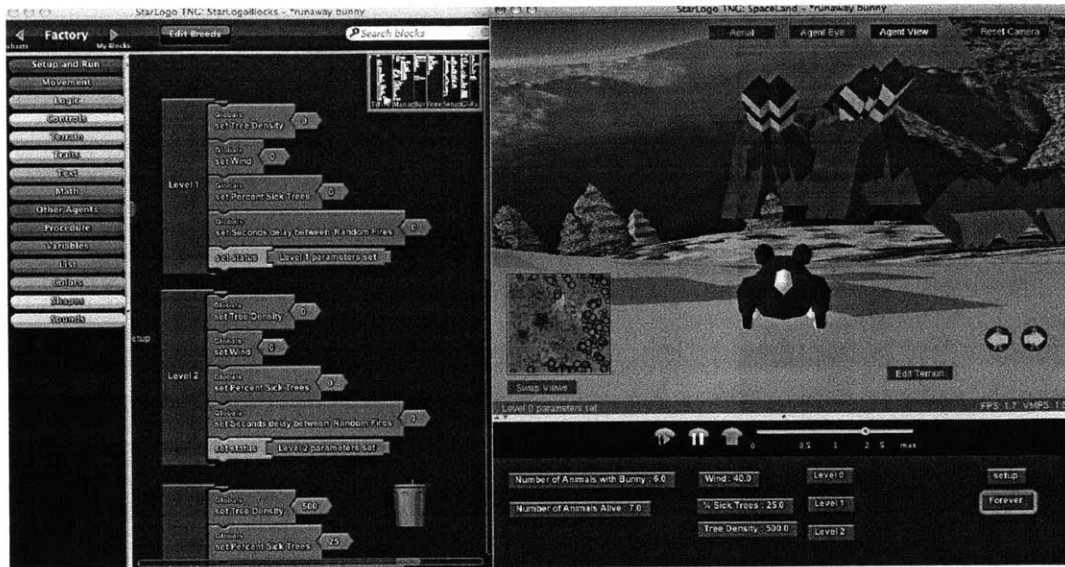


Figure 1: An example of code in the StarLogoBlocks window (left) and corresponding interactive SpaceLand environment (right)

Devised for educational purposes, a second distinctive characteristic of StarLogoTNG is that it can be used to create interactive simulations and models. This trait makes the software a useful tool for enriching the learning of science and engineering concepts as it allows students to “express and explore their own understanding of a given phenomenon” (Klopfer et al. 2009).

Third, the programming aspect of StarLogoTNG serves to expose students to many basic computer science programming and algorithm concepts in a less intimidating manner. This occurs because the burden and confusion of memorizing syntax in order to write and debug code is removed through the abstraction created by the code blocks. Therefore, programming concepts can be easily learned because students are able to “focus on what the commands should be doing rather than syntax” (Klopfer et al. 2009). This allows students to gain the intuition they need to write programs without the distraction, frustration, and intimidation that can arise when learning a specific textual programming language.

The modules that I will discuss in this paper leverage these three significant strengths of StarLogoTNG in a manner that supports the goals of the AP CSP course.

2.3 Teaching CS Principles using StarLogoTNG

In the past, StarLogoTNG has been used to supplement learning in STEM courses such as math, physics, biology, and general technology classes. The lessons and activities focused on specific concepts within the particular subject's curriculum, and were interspersed between traditional lessons to reinforce and further students' understanding of the information and notions being taught. StarLogoTNG has also been used to teach programming and a set of activities that introduces students to programming concepts has been developed. However, these activities were not designed to branch into other themes and principles of computing, and a set of activities in StarLogoTNG that incorporate programming but primarily focus on computing-related principles has yet to be created.

The characteristics of StarLogoTNG make it an ideal learning tool for the AP CSP course since the properties of the software directly relate to several of the courses main themes. One of the Big Ideas of the AP CSP course is creativity, where the aim is to do more than simply teach the concept that “computing requires creativity”. The course wants students to “actually be creative: creating artifacts that they want to show off to their friends and family”. The 3-dimensional environment of StarLogoTNG and video game like characteristics make the software an appropriate medium for students to develop their own games. Game programming using StarLogoTNG allows students to envision their very own world, characters, character interactions, and “allows students to invest themselves personally in their products”. This behavior is what the AP CSP course desires within its classrooms (“AP Computer Science Principles” 2013).

Another reoccurring theme found in many learning objectives of the AP CSP course is to have students understand abstractions, analyze problems and artifacts, and use simulations to generate new knowledge. The signature quality of StarLogoTNG as a tool for learning through modeling and simulations align with these computational thinking practices, and make the software it an efficient tool for developing activities that will highlight and reinforce these themes.

Third, the visual programming aspect of StarLogoTNG makes it easier for students to learn programming concepts. The AP CSP course still wants to introduce students to programming and algorithms, yet it may not be appealing to many students

to have to become familiar with intimidating development environments or software development kits such as the one depicted in Figure 2. The text based nature and complicated functionalities of traditional programming tools make them a very useful tool for college-level computer science majors and professional developers, but can make learning programming a daunting task for K-12 students. Combined with the task of having to memorize specific syntax and gain a certain level of proficiency in order to even be able to create something that appears interesting or worthwhile to a student, the use of traditional programming environments is one of the factors that make programming appear dull to many K-12 students in the first place. A visual programming language like StarLogoTNG can better serve the objectives of the AP CSP course by providing students with a vibrant and interactive environment. Students will find more pleasure in learning programming in the context of a development environment that to them has a more learnable, usable, and attractive user-interface.

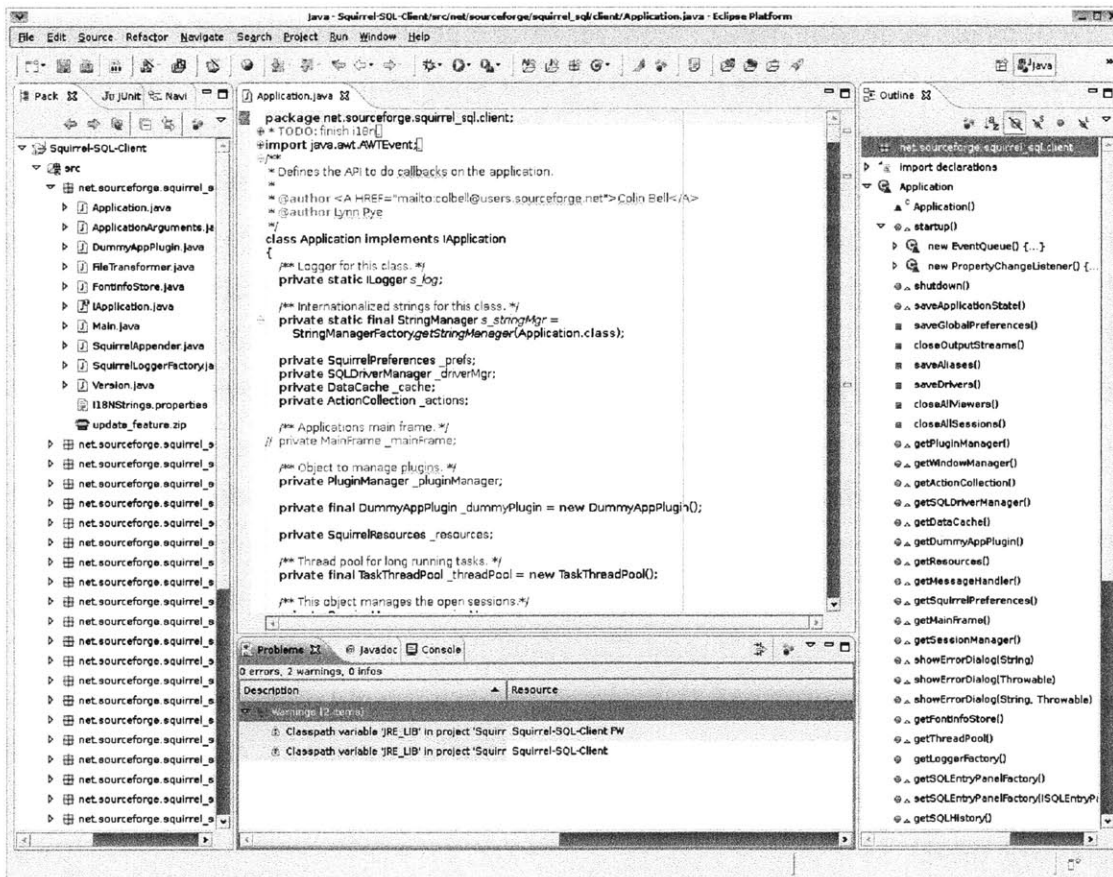


Figure 2: A sample workspace of Eclipse, a very popular integrated development environment.

The pilot sites of the AP CSP course have in fact already acknowledged the benefit of using visual programming tools to teach programming concepts. Out of the pilot sites of the 2010-2011 and 2011-2012 school years, several high schools had integrated activities using Scratch and/or App Inventor, two other visual programming software environments also developed at MIT. StarLogoTNG has not yet been a choice to be integrated into high school pilot curriculums although it possesses similar characteristics to Scratch and App Inventor, as well the ability to facilitate learning through models and simulations. Scratch & App Inventor tend to be used in the pilot curriculums mainly as a tool for exercising learning objectives within the Big Ideas of Creativity, Programming, and Algorithms through game programming and/or completion of existing tutorials and activities developed for those software. StarLogoTNG is not only capable of providing the same game programming experiences, but its usefulness as a tool for creating simulations and models can be leveraged to help address learning objectives from other Big Ideas as well.

Lastly, the AP CSP course plans on integrating special performance tasks into the curriculum to set a standard for instructors to evaluate their students. These performance tasks require that students complete specific tasks with detailed guidelines that test their understanding of particular areas, and are to be conducted throughout the course and used to help determine a student's overall AP score. Activities created with StarLogoTNG have the potential to serve as precursors for these performance tasks, and present an opportunity for students to learn and/or practice the same concepts and learning objectives that will be tested in the official performance tasks.

In my project I combined all of these strengths and potential benefits of StarLogoTNG to create 3 modules that focus on the Big Ideas of Internet, Creativity, and Data, while incorporating aspects of the remaining 4 Big Ideas of Abstraction, Programming, Algorithms, and Impact. The result is a set lesson plans that together cover all of the Big Ideas of the AP CSP course, yet do not build upon each other. This way, the modules can be used separately and not lock instructors into having to commit to incorporating all of them into their curriculum in order for any 1 particular module to be able to be used.

3 LEARNING FRAMEWORKS

This section details several learning techniques and frameworks that have been incorporated into the StarLogoTNG AP CSP materials.

3.1 Acquire-Meaning-Transfer (AMT)

In order for students to fully grasp the materials they are presented in classrooms it is not enough to only teach the necessary information and test their ability to recall that information. This is a pattern that many high school courses follow, especially Advanced Placement courses where there are “external test pressures that demand superficial content coverage” (McTighe and Wiggins, 2001) causing instructors to teach with the main goal of covering the AP exam content without prioritizing teaching for understanding. Although covering all the required topics of a particular subject is indeed necessary, dangerous trends begin to emerge when the goal of teaching for understanding is excluded.

Discounting the objective of teaching for understanding has proven to result in negative effects for students. When students’ abilities to simply recall information is prioritized, the mental stimulation that comes from actually understanding the content is replaced by the tedious task of pure memorization. Due to the monotony of the process a student’s desire to learn can be decreased or even replaced with passiveness and indifference, which leads to poor performance. Jay McTighe and Grant Wiggins, teachers and educational consultants, state in an article that:

“In one district, the results of end-of-year science exams reveal a troubling pattern: Students typically perform adequately on items requiring recall and basic skills but do poorly on items requiring application or careful analysis and explanation.”

McTighe and Wiggins champion the idea that high school education should not be a task of covering content, but rather should help students become more thoughtful and productive about what it is they are learning so that they can take what they learn and apply them in the future situations. They propose that in order to learn for understanding, classroom lessons need to have students address the following 3 tasks:

1. Acquire important information and skills

2. Make meaning of the content

3. Effectively transfer their learning to new situations both within school and beyond it.

This framework, known as Acquire-Meaning-Transfer (AMT), aims to have students go beyond just being able to recall information. It challenges them to understand the content in a way that allows them to apply what they learn on exams, new problems, and future courses. The ability to apply concepts is the mark of true understanding and the goal of effective learning. To have students obtain such skills is one of the ambitions of the AP CSP course; many of the courses learning objectives concern students' ability to demonstrate understanding by applying what they have learned to solve a new problem or complete a particular task. Therefore, the AMT framework is an ideal model for a lesson plan that aims to help students along the path of learning and comprehending AP CSP content on more than just a superficial level.

Activities within the StarLogoTNG modules can be structured such that they include the processes of acquiring knowledge, making meaning, and transferring learning of key concepts and learning objectives of the AP CSAP course. Alternatively, activities can even be structured to satisfy one of the three steps, such as transfer, allowing the module to be integrated into a larger lesson plan, giving instructors the opportunity to include other mediums of learning while teaching specific content.

3.2 Peer Instruction

Peer instruction is a teaching method that was developed to help increase student understanding. It involves students analyzing and discussing solutions to problems among themselves before the instructor discusses the solution. This kind of interaction between students allows them to “prompt each other to attend to small details” and “form arguments and provide rationales” in order to come to a conclusion about the particular problem at hand (Cutts and Simon, June 2012).

In order for peer instruction to work as desired the instructor must “develop questions that really engage students in deep and meaningful discussion” so that the process does not turn into students simply checking or obtaining their answers from others. (Cutts and Simon, June 2012) In addition to encouraging meaningful

discussions that lead to better understanding, the instructor can also develop problems that apply the transfer process of the AMT method to check understanding of a particular concept by requiring students to be able to apply previously examined knowledge to solve a problem. One common approach that achieves this goal is to present a problem that may have multiple correct solutions, as it will lead to students presenting various arguments during peer instruction and exercising their understanding of the material by examining various justifications for a particular answer.

Peer instruction has been proven to result in more active, engaging, and effective learning in classrooms. These are the same goals the developers of the AP CSP course seek to incorporate as much as possible, which makes peer instruction a useful method to incorporate into learning activities meant to teach AP CSP content. In a paper detailing 10 years of results from implementing peer instruction in classrooms Eric Mazur, the inventor of the peer instruction method, stated that utilizing this teaching technique resulted in dramatic improvements in how students fared on performance exams and with traditional quantitative problems. Many students enjoy this learning method as well. Students are more engaged as they take a more active role in their learning to increase their understanding of the material. When implemented in a computing course one student stated:

“Discussing my understanding in comparison to my seatmates helped me gain a larger understanding of the material by approaching it from different perspectives.” (Cutts and Simon, February 2012).

Although originally developed for physics instruction, the peer instruction method is applicable to computing courses due to the similar issues that arise in both physics instruction and computing instruction regarding students being able to gain an actual understanding of the material. In the same way that a correct answer to a physics problem does not guarantee understanding and could be the result of a student plugging and chugging various given equations, it is proposed that a student’s ability to create a working program doesn’t guarantee that they fully grasps the concepts. Cutts and Simon claim that it means that we “can only assure that our students can produce a working program”. The peer instruction method can be implemented in computing courses to combat the issue that “often the computing community seems focused on what to teach

but not how to teach it” (Cutts and Simon, February 2012), which can lead to students knowing basic information but not being able to apply it. This is one of issues that AP CSP aims to address within computing courses. In addition, the nature of the peer instruction method employs the AP CSP computational thinking practices of analyzing problems and artifacts, and communication. These factors make peer instruction another appropriate method to incorporate into materials that will assist in the instruction of AP CSP content.

3.3 Peer Review

Similar to peer instruction, peer review is a method that when implemented in a classroom setting² involves discussion amongst students about each other’s work and thought processes. This particular process is used for the purpose of increasing the amount of feedback a student will receive in order to broaden their current thoughts or ideas. Being able to give constructive criticism and point out strengths and potential shortcomings in the work of others helps students display an understanding of the concepts or skills that are being tested or exercised in the assignment because they are able to apply their current knowledge and logic to identify what works and what might not work. Likewise, receiving such constructive criticism allows students to learn from one another and merge opinions in order to broaden their scope of the concepts of the particular subject.

Peer review is used in courses such as writing or literature where they may not be any right or wrong answers to debate, yet there exists opportunities for students to refine their work through exchange of ideas before it gets evaluated by the instructor. In relation to the AP CSP course, there is a lot of room for students to benefit from the incorporation of peer review during activities that feature the types of learning objectives that require open-ended thinking, such as some of the Creativity learning objectives and discussions regarding the Impact of computing in society.

² Unlike in an academia and research setting where the process of peer review can also used to judge the legitimacy of a piece of work, in a classroom setting peer review will not used in these materials to deem whether or not a student’s work is “acceptable” or “good enough”. Such applications of peer review would be discouraging to students if those labels were attached to their work by their peers.

4 DESIGN APPROACHES

This section details three approaches that serve to help frame the development of the AP CSP StarLogoTNG modules. They were incorporated into the design process of all three. The first is the ADDIE model, which is an instructional design standard used to guide the creation of each lesson plan. The second approach is using concept activation at the beginning of each lesson for the purpose of peaking student interest and stimulating learning. The third is the desire to include the reoccurring theme of connecting computing to real word impacts at the end each of the modules.

4.1 ADDIE Model

The approaches taken to design my 3 modules were rooted in methods of instructional design, “a technology which incorporates known and verified learning strategies into instructional experiences which make the acquisition of knowledge and skill more efficient, effective, and appealing” (Drake, L. et al 1996). Instructional design systems have the purpose of creating suitable learning activities that guide students towards understanding the required knowledge of their courses. Many instructional design systems tend to include backward design, a technique that designs lessons plans as the means to an end. In backwards design the initial focus lays on what the learning goals of the lesson will be then uses these end goals as a springboard to design the curriculum path that can be taken to get there. Given that the AP CSP course has clear learning objectives for all of the key concepts for each of the 7 Big Ideas, my modules were designed by first choosing which learning objectives would be highlighted in each module. This was done by

1. Deciding which of the 7 Big Ideas would be the main focus of the module and what learning objectives an activity involving StarLogoTNG might be suitable for addressing.
2. Incrementally considering which learning objectives from the other Big Ideas would also be included in the module by seeing where there exist opportunities to make connections to other AP CSP key concepts. This step included possibly re-thinking the main focus of the module.

3. Iteratively designing the lessons to include activities that display evidence of learning for the chosen learning objectives.

One very common method used in instructional system development is the so-called ADDIE model, which is an acronym that outlines the following steps for successful curriculum design:

1. **Analysis:** Problems, goals, objectives, and needs are identified and considered.
2. **Design:** Identify details necessary to meet goals dictated during Analysis.
3. **Development:** Create content and materials based on Design.
4. **Implementation:** The completely developed lesson plan is put through a trial with the target audience.
5. **Evaluation:** Formative and summative evaluations are conducted and revisions are made as necessary. Such revisions require going back to any 1 of the previous 4 steps, and repeating the recursive process.

The ADDIE model has been further developed and modified since its inception in 1985, resulting in many variations that have added additional sub-steps or iterative components to the original hierarchical version. Nevertheless, all variations of the ADDIE model, as well as more contemporary instructional system development techniques, involve some form of the aforementioned steps. Figure 3 visualizes these steps and the iterative cycle in which they are executed in order to design a successful lesson plan. This more iterative model depicted in Figure 3 includes evaluation after each step, and was the model I used to guide the development of my modules.

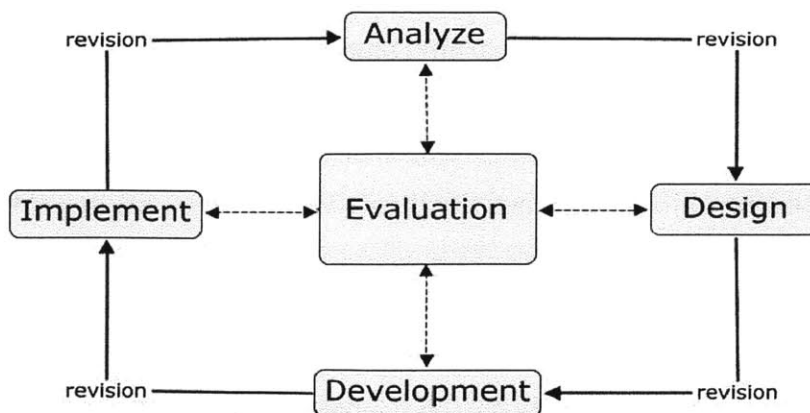


Figure 3: Diagram of the ADDIE model

The backwards design and iterative design elements of such a model holds many benefits. Since backwards design requires the first step to be a detailed analysis of what the lesson plan's goals will be, the following steps are all kept in line with achieving the desired end. In addition, the iterative aspect allows formative evaluation to be conducted at each step of the process, not just after all the work to complete the process has been done. This makes the development process more cost effective. Although less linear it presents the opportunity to make necessary modifications and improvements earlier in the cycle so that errors or unwanted choices, that will eventually be modified anyway, do not propagate further down the creation process. For example, an unfavorable design choice may be discovered during a post-Design evaluation before time is invested to develop, implement, and evaluate it as part of the curriculum. The benefits provided from using an instructional system development method such as the ADDIE model will allow for the creation of lesson plans that have been evaluated in depth and are successful in adopting the chosen learning objectives.

4.2 Concept Activation

Previous pilots of AP CSP courses revealed the effectiveness of beginning a session by activating concepts that would be presented in the lesson. Concept activation is the process of “engaging in an activity that results in desired information being transferred from long term memory to consciousness” (Paul, 2012). From a cognitive stand point, this primes the students to begin recalling any relevant ideas or thought processes that they can apply as they learn. In addition, concept activation can stimulate curiosity or interest in students' minds as they recollect ideas, which they already know and understand, and begin to form questions or wonder how it applies to what is to come. Jody Paul, the professor of a college-level CS Principles pilot at the Metropolitan State College of Denver, found that activating a concept and engaging curiosity encouraged “the cognitive processing necessary to connect new concepts into a student's existing knowledge framework” (2012).

Consider this example: Imagine we were attempting to explain the three states of matter—solid, liquid, and gas—to a young child. We can take a linear route of presenting necessary information by

1. Telling the child that there are three states of matter.
2. Giving the child definitions of each state.
3. Presenting an example to tie concepts together and complete the lesson.

However, immediately starting off with the foreign idea of “states of matter”, which the child does not understand or possibly does not even care about, may cause them to become confused or apathetic due to no desire to understand what is being taught. A more effective approach that uses concept activation would proceed as follows:

1. Ask the child if they ever left ice in a cup on a table for too long, and what happened.
2. Ask the child what they notice happens to puddles after it rains and the sun comes out.
3. Ask the child to think about a time when their parents bought freeze-pops. Right out of the box they look like colored juice but out of the freezer they are now cold and hard. Ask them to think about why that happens.

Here, before exposing the child to how water changes forms and new words such as evaporation, condensation, liquefaction, and solidification, the basic concept that water can have different forms in the first place is activated in the child’s mind. They recall phenomena they may have already seen or experienced, and even though the child the may not yet understand the reasons or significance behind these processes, the curiosity about how these things happen has been peaked. The child can now be eased into the scientific details of why this all happens, and this gradual introduction makes them more likely to understand and stay interested in the topic.

Still, the process of concept activation does have its limitations. Paul found in her class that using a numerical example too early in the lesson as part of concept activation resulted in some students becoming alienated from said concepts, probably due to the so-called “math phobia” that plagues some students and discourages the learning of STEM concepts. Paul states, “far more effective was to introduce and activate a concept using a non-numerical example” (2012). Because of the benefits of utilizing concept activation versus jumping right into the lesson material, incorporating an activity that activates student’s perceptions of relatable ideas at the beginning of each of my AP CSP modules is another approach shared between all three suggested lesson plans.

4.3 Connecting Computing - Impact

Another theme that was chosen to be included in each module was the Big Idea of Impact. Other than increasing student knowledge and the understanding of computing concepts, one of the issues AP CSP intends to tackle is the lack of interest among high school students in computing-related careers. In order to stimulate interest, not only are these lesson plans designed to provide students with a more interactive learning experience, they also include discussions and concrete examples about how some of the ideas they have learned and explored are applicable to various fields of study and affect society. Students will transfer their understanding of the topics in order to actively participate in these meaningful discussions.

Seeing how broadly computing can be applied to numerous fields and subjects may aid in more students gaining interest in pursuing computing related careers. These discussions and examples involving the impact of computing are placed towards the conclusion of each module because I believe that making connections between concepts from the module and the real world will have greater validity and meaning for the students *after* they have explored and gained a deeper understanding of said concepts.

5 CREATIVITY MODULE

The process of assessing creativity, especially within a classroom setting, can initially appear to be purely subjective. However, work has been done regarding how creativity can be assessed based on specific quality indicators. Although creativity cannot be judged in a binary sense of creative vs. not creative, the quality of specific characteristics of a creative work can be judged on a spectrum. In other words, measuring qualifying characteristics exhibited from the creator and within the creative expression in question can assess creativity.

For example, one such quality indicator stems from the idea of synthesis of ideas. Most often creativity is assessed based on the quality indicator of whether or student a student can come up with multiple ideas, and build upon existing ideas to form new ones. However, we can also evaluate how well these ideas were synthesized into some form of structure.³ Given certain quality indicators that an instructor would like to assess for creativity, student activities and tasks can be intentionally designed to have them work on these indicators (Miller, 2013). The creation of this module was guided by the not only by specific learning goals directly from the AP CSP course framework, but given that the focus is creativity this lesson plan incorporates space for quality indicators to be assessed. Borrowing from an educational model developed by the Institute of Creativity and Educational Innovations (INCIE), this module takes the following quality indicators into consideration.

1. Imagination
2. Ability to formulate new approaches and exhibit mental flexibility
3. Capacity for synthesis and organization of ideas

These quality indicators are later discussed in Section 5.2.8.

³ The term “structure” here is not to be confused with a sense of generic organization or conforming to specific terms, ideas we tend to associate with the word “structure” but not “creativity”. It is important to remember that structure can itself be original and creative. Many pieces of art follow some form of structure set by the artist, yet it doesn’t make the artwork any less original or creative.

5.1 Creativity: Analysis

Creativity was selected to be a main topic for one of the 3 modules because StarLogoTNG’s video game like environment would be very suitable for a game programming activity. In terms of creativity, a game programming module where students design, create, test, debug, play, and present their very own games would be suitable for having students experience the key concepts presented in Table 1 by participating in activities that allow them to display the learning objectives listed below. Game programming address the key concept that computing fosters the creation of artifacts (Table 1 – Creativity Key Concept A, Learning Objective #1: 1a, 1b, 1c), which in this case are the games themselves, and that programming can be a very creative process (Table 1 – Creativity Key Concept C, Learning Objective #4: 4a, 4b). StarLogoTNG becomes a vehicle for creative expression and exploration (Table 1 – Creativity Key Concept B, Learning Objective #3: 3a, 3b, 3c), yet will still require students to exercise critical thinking and computation thinking skills as they evaluate their work in order to correct issues within their program and choose appropriate methods to program their game’s features (Table 1 – Creativity Key Concept A, Learning Objective #2: 2b, 2c).

CREATIVITY		
Key Concept A. Computing fosters the creation of artifacts.		
Learning Objective 1	The student can use computing tools and techniques to create artifacts.	
	<i>Evidence of Learning Objective</i>	1a. <i>Creation of a digital artifact with a practical, personal, or societal intent.</i>
		1b. <i>Selection of appropriate techniques to develop digital artifacts.</i>
		1c. <i>Use of appropriate algorithmic and information-management principles in translating one’s intention into a digital artifact.</i>
Learning Objective 2	The student can analyze computational artifacts.	
	<i>Evidence of Learning Objective</i>	2b. <i>Location of weaknesses and errors in a digital artifact.</i>
		2c. <i>Explanation of how a digital</i>

		<i>artifact functions.</i>
Key Concept B. Computing fosters creative expression.		
Learning Objective 3	The student can use computing tools and techniques for creative expression.	
	<i>Evidence of Learning Objective</i>	3a. <i>Use of appropriate computing tools and techniques for creative expression.</i>
		3b. <i>Use of new forms of expression enabled by computing.</i>
		3c. <i>Selection of appropriate computing techniques for creative exploration.</i>
Key Concept C. Programming is a creative process.		
Learning Objective 4	The student can use programming as a creative tool.	
	<i>Evidence of Learning Objective</i>	4a. <i>Creation of a program with a practical, personal, or societal intent.</i>
		4b. <i>Creation of a program that satisfies personal curiosity or expresses creativity.</i>

Table 1: A table showing the learning goals of Creativity selected for the Creativity module.

It goes without saying that an activity where students design and create their own games in StarLogoTNG will be very programming intensive. Therefore, this module will also be designed to include applicable Programming and Algorithms key concepts and learning objectives. Student will have to go further than just the creation of algorithms and use of programming, as they need to be required to debug and test their games in order to satisfy learning objectives that require evidence of the ability to explain a program, correct errors, justify decisions, or identify concerns within a program. Additionally, these processes will be necessary regardless in order for students to create working games. Table 2 and Table 3 detail the key concepts and learning objectives of Programming and Algorithms that can be incorporated into the breadth of this module.

ALGORITHMS		
Key Concept A. An algorithm is a precise sequence of instructions for a process that can be executed by a computer.		
Learning Objective 15	The student can develop an algorithm	
	<i>Evidence of Learning Objective</i>	15a. Selection of appropriate techniques -- such as sequencing, selection, iteration, and recursion -- to develop an algorithm.
		15b. Selection of appropriate combinations of algorithms to make new algorithms.
		15c. Creation of an algorithm to solve a problem.
		15d. Creation of an algorithm with a practical, personal, or societal intent.
Key Concept B. Algorithms are expressed using languages.		
Learning Objective 16	The student can express an algorithm in a language.	
	<i>Evidence of Learning Objective</i>	16a. Use of natural language, pseudo-code, or a visual or textual programming language to express an algorithm.
		16b. Explanation of how an algorithm is represented in natural language, pseudo-code, or a visual or textual programming language.
		16d. Summary of the purpose of an algorithm.
Key Concept D. Algorithms are evaluated analytically and empirically.		
Learning Objective 18	The student can evaluate algorithms analytically and empirically.	
	<i>Evidence of Learning Objective</i>	18b. Location and correction of errors in an algorithm.
		18c. Explanation of how an algorithm functions.

Table 2: A table showing the learning goals of Algorithms selected for the Creativity module.

PROGRAMMING		
Key Concept B. Programming is facilitated by appropriate abstractions.		
Learning Objective 20	The student can use abstraction to manage complexity in programs	
	<i>Evidence of Learning Objective</i>	20a. <i>Use of functions as re-usable programming abstractions.</i>
Key Concept C. Programs are developed and used by people.		
Learning Objective 21	The student can evaluate a program for correctness	
	<i>Evidence of Learning Objective</i>	21b. <i>Location and correction of errors in a program.</i>
		21c. <i>Justification of program correctness.</i>
		21d. <i>Explanation of how a program functions.</i>
Learning Objective 22	The student can develop a correct program.	
	<i>Evidence of Learning Objective</i>	22a. <i>Use of an iterative process to develop a correct program.</i>
		22c. <i>Location and elimination of errors in a program written by the student.</i>
		22d. <i>Identification of programmer and user concerns in program development.</i>

Table 3: A table showing the learning goals of Programming selected for the Creativity module.

Lastly, the topics of this module allows for Impact to be tied in through discussion of how computing has led to innovation in the game development industry, and examine specific example regarding how computing has enabled advances in various technology fields. For instance, comparing video games of the current decade with those of previous generations presents openings to discuss how innovations in computing have facilitated growth in the type of games we can play, the features of these games, and the means by which we play them. Such topics can include discussion of expansions in the fields of artificial intelligence, computer graphics, hardware development, etc.

IMPACT		
Key Concept B. Computing enables innovation in nearly every field.		
Learning Objective 29	The student can connect computing with innovations in other fields	
	<i>Evidence of Learning Objective</i>	29a. <i>Identification of the impacts of computing on innovation in other fields.</i>

Table 4: A table showing the learning goals of Impact selected for the Creativity module.

Given the learning objectives of this module, there are several problems regarding the structure of the module that will need to be addressed during the design phase. The first issue concerns how the game design process can be structured so that students will avoid design pitfalls, i.e. getting carried away with ideas or creating games with features that will be difficult or impossible to achieve with StarLogoTNG, without being so tightly structured that students' abilities to be creative are hindered.

The second issue concerns how to guide students through the development process of their game after it has been designed. The lesson plan has to outline a path that will be taken to get from the point where students have a solid game design and the point where they have a complete and functional game. This framework needs to assist students through tasks such as developing algorithms, translating those algorithms into StarLogoTNG code, being able to correctly program these algorithms, and testing their programs. Without such a structure we risk students getting stuck during the development process with buggy code, becoming confused or frustrated attempting to achieve their original goals, and possibly never being able to complete a working game. This is a situation we want to avoid at all costs. In order for this module to be effective students must continuously feel as if they are making progress with getting parts of their game to work piece by piece until it is complete. If students feel a sense of accomplishment as they program working components of their games one by one, they will feel empowered and motivated throughout the programming process as opposed to feeling frustrated and discouraged.

Lastly, an established framework to guide students through development does not guarantee students will be successful in creating their games if they follow the set

guidelines. As in actual game development, students must be required to evaluate their work incrementally so that they have opportunities to modify their choices based on any roadblocks they encounter. In addition, students will need to have opportunities for the instructor to review their work. These will serve as checkpoints for the instructor to evaluate progress, assist with roadblocks, and point issues that need to be addressed before students can move on in the development process.

5.2 Creativity: Design

5.2.1 Concept Activation & Impact

Opening this module with a class discussion that also serves as a brainstorming session will be used as a concept activation activity for students. The lesson will begin with a class discussion about how computing relates to video games and encourages creativity. By discussing innovations in video and computer games that have occurred during their lifetime, students will recall older games they have played and realize how the kinds of games they play and the methods by which they play them has advanced due to technology. This creates an opportunity for the instructor to reinforce the impacts of computing in innovation within other fields⁴ (Table 4- Impact Key Concept B, Learning Objective #29, 29a). In doing so students begin to recollect the various types of games they have played, which then leads into an activity where the class might generate a list of different game genres and types of games.

The discussion eventually leads to focusing on 1-player games, which will be the types of games students will create in StarLogoTNG, and the theme that all of their games will have to center on is introduced. The theme selected for the module is “obtaining treasure” because such a theme is vague enough to be incorporated into many types of games, but is still something to assist students with reigning in their imaginations and ideas as they decide what their game will be. The class now discusses 1-player games of the previously listed game genres and brainstorm ways that 1-player-

⁴ These ideas can also be reiterated at the end of this module with a more in depth discussion of specific forms of advancements in technology and computer science that have contributed to innovation within the game development industry.

games can include the theme of “obtaining treasure”. From discussion of all these ideas, students’ mindsets are now primed to think about what genre of game they would like to create and what the storyline of their game might be.

5.2.2 Game Description Narrative

After students are placed into groups and decide on what kind of game they want to make, the exact details the game-play may not be solidified in their minds. Especially since it is suggested in the lesson plan that students work in teams to facilitate many aspects of the large task ahead, there may be many ideas that each team currently wants to explore for their game. Students are now asked to frame their game as a narrative, and write a story from the point of view of the main character of the game. This story will be representative of the type of events that can happen in their game. The story may be one that tells a tale of the player going from the start of the game to winning, or even from the start of the game to losing.

Even though this module is a computing activity for a computing related course, the purpose of framing a game as a narrative role-play presents a context for students to begin utilizing computational thinking skills such as Boolean logic and conditional expressions (Ingram-Goble, 2013). Consider the following example being part of a potential game narrative:

“I had collected both the red and blue jewels, so I had the ability to fly. But I had not yet collected the yellow jewel, which would be my final obstacle in my journey to save SpaceLand.”

The concept of using Boolean logic to establish certain game states is displayed. Likewise, conditional logic is exhibited within the same example narrative as we see that if-then-else logic is used to set a foundation for when specific events will occur in the game. IF the player has (red jewel AND blue jewel AND yellow jewel) THEN the player saves SpaceLand.

Upon completing their story, students are asked to go back and highlight within their anecdote the various characters and objects that appeared in the story, the different actions each character took, any interactions between characters and/or objects, and specific events that occurred when certain states of the game were either true or false. Unknowingly, the act of writing a game narrative has provided students

with a foundation for their character design process by helping them envision some of the entities that will exist in their game.

5.2.3 Character Design Process

The approach that was taken to help students decide what characters and objects they would have in their game was done by requiring students list the goals or purposes of these characters. Based on these choices, students now assign functionalities to the different game entities that will be needed for the entities to serve their purposes in the game or achieve their goals. Characters and objects translate to the breeds that will be created in StarLogoTNG, and functionalities translate to the procedures that will be written for each breed.

The act of considering a character or object's goals or purposes in the game ensure that students do not get too carried away with creating ones that may be unnecessary. Using the chosen goals or purposes as a guide to assign functionalities for each entity allows students think through all the components that they will need to program in order for their game entities to behave as desired. In addition, laying out these purposes and functionalities for each entity will allow for students to see earlier, rather than later, where it may make sense to use abstractions and re-usable functions. One such instance could be if students notice they are developing characters that tend to have the same exact functionalities that vary based on the parameters used. Here, they will either realize or be taught by the instructor that they can by combine these breeds into just one breed, and use a re-usable procedure that takes parameters to account for any relevant variations in behavior. Suggestions for when students may want to use re-usable procedures are also included in the student materials of this module, and being able to use them in their programs as abstractions allows students to display evidence of using abstractions to manage complexity in programs. (Table 3 – Programming Key Concept B, Learning Objective 20: 20a)

5.2.4 Functionality Classification

As part of their design process, students are then asked to classify the functionalities they have decided on into one of the four categories:

1. Input - Function triggered by user input

2. Automatic - Function doesn't need to be triggered by anything and is something an agent will do constantly as long as it is alive in SpaceLand.
3. Event - Function triggered by a conditional event that has to be constantly checked for.
4. Collision - Function triggered by a collision with another

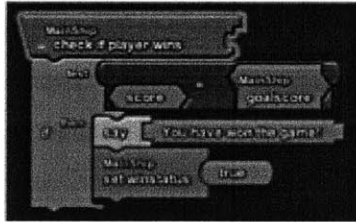
This set of classification categories stemmed from examining the possible features that can get programmed into a simple StarLogoTNG game⁵ and considering what terms and accompanying definitions could be generated to help students begin to think of the game in terms of the types of components that change the state of the game.

EVENT Functionality: This is a functionality that is triggered when a certain condition within the game occurs.

How To Code
 Event-type functionalities are implemented by creating a procedure that uses *if commands* to check for whether or not the condition has occurred, and setting the behavior that the agent needs to execute as long as the condition is currently satisfied.

The procedure gets called by the relevant breed(s) from the "Play Game" button, which is a forever button. This allows the condition to be constantly checked for so the agent can be always be informed as to whether or not to execute the specified behavior.

Example:

Functionality	Pseudo Code	StarlogoTNG Implementation
Player wins the game when the score equals the goal score.	<pre> <score> – a shared number variable <goalscore> – a shared number variable if <score> is equal to <goalscore>: player has won the game </pre>	

How To Test

Test Detail	Passing Criteria
<ol style="list-style-type: none"> 1. Setup and start game 2. Play through the game and make the event that triggers the behavior true OR Create a run once button that will set the criteria to true 3. Verify that the observed behavior of agent(s) matches the behavior that is desired 	The agent correctly executes the expected behavior as long they are alive and the event has occurred and/or is still occurring.

Figure 4: An example of how to code and test procedures that execute certain actions that only happen when a specific event occurs in the game.

⁵ In this context a simple StarLogoTNG game would be one that does not have multiple levels or cut scenes, only requires the user to user to select one pre-programmed "Setup" button and "Play Game" button to initiate game play, and has a clear and concise goal to win the game.

The reason behind classifying functionalities is that it can assist with students having the correct intuition for how to code the procedure, where to place the procedure, and how to test the procedure during development. In the accompanying course materials, there is a document that details these 3 aspects for each functionality type. The belief is that these classifications will be helpful for less experienced students who do not yet have the intuition to automatically know where to implement a procedure, to realize what StarLogoBlocks might be relevant to coding it, or to know where it should be placed in order for the desired behavior to get executed. Figure 4 is an example of the instruction that applies to the Event-type functionalities.

5.2.5 Algorithm Design

For each functionality of each entity in the game, students are asked to assign a name to the procedure that will implement it, describe its algorithm, and assign it to the relevant breed. This part of the module serves to have students use natural language and/or pseudo-code to describe their algorithms. This step requires students to think about each of their algorithms and evaluate the correctness of their plan as a team. It also allows students to realize what features of their game they may not actually know how to code, and receive assistance before they begin development. Here, students exercise evidence of being able to develop an algorithm (Table 2 – Algorithm Key Concept A, Learning Objective #15: 15a, 15b, 15c, 15d). They also exercise evidence of being able to express an algorithm and pseudo-code –and later a visual programming language—as well as summarize the purpose of each algorithm. (Table 2 – Algorithm Key Concept B, Learning Objective #16: 16a, 16d)

5.2.6 Programming & Testing

By completing all of the previous steps students have laid out all the features of their game. Referring to their design documentation when necessary, teams no go through the process of creating each breed, writing procedures, and placing the procedures where they need to be called and in the correct sequences for their game to behave as desired. Ideally, the established framework will allow for this to be a very streamlined process.

However, one challenge here is that we want students to always have a working

program, even if all of the features have not been coded. As a result, students are instructed to first implement the behavior of their “Main Player”, which is the character that the gamer controls. They test and debug each procedure as they go along and feel a sense of progress as after their first several steps they now have a character that moves around the world and can execute some of its behavior. Students are then asked to implement the behavior of the rest of their breeds, taking into consideration the idea of “bundles”. Bundles are defined in this module as a sequence of functionalities that help complete the same goal or purpose—and therefore must all be coded together before being able to test whether or not the game behaves as desired. When coding a bundle, students test the new code after all procedures or collisions in the bundle are coded. The overall outline for this process has students mimic the real world programming practices of incremental development and unit testing, and will lead to a less frustrating experience.

This process for programming and testing touches upon Key Concept C of Programming, which is that programs are developed and used by people. One learning objective of this key concept is that students can develop a correct program and evaluate a program for correctness. In order to do so, students must be able to use an iterative process—such as the one designed in this section—to develop correct programs, and locate and eliminate errors in a program they have written. (Table 3 – Programming Key Concept C, Learning Objective #22: 22a, 22c). In locating errors in their programs, students may also be specifically discovering errors in their algorithms. (Table 2 – Algorithms Key Concept D, Learning Objective #18, 18b)

5.2.7 Applying Peer & Instructor Review

Throughout the lesson, sessions of peer review as well as instructor review are included. Peer review not just helps student receive feedback on their work but can also encourage sharing of ideas. Creating an environment where teams can help each other debug or share how they implemented certain features that another team may also want to include in their game further facilitates creativity. The peer review sessions that occur during after the programming and testing stages further helps students show that they can evaluate a program for correctness. Students will explain to each other how their program functions and attempt to assist with finding, correcting, and explaining errors

within their peers code. (Table 2 – Programming Key Concept C, Learning Objective #21: 21b, 21c, 21d).

Instructor review is used as a checkpoint after certain phases to make sure students are headed in the right direction, review their work before they can move on to the next major step, and give assistance if a particular team is having a problem that neither them or their peers were able to figure out. Students also use this opportunity to discuss any changes to their design they want to make after discovering specific concerns on their own, or having an issue revealed to them during peer review. (Table 3 – Programming Key Concept C, Learning Objective #22: 22d).

The instructor review sessions that occur after major phases of the module are also an opportunity for instructors to evaluate students' understanding of their own decisions. For example, asking each member of the team to explain how one of their algorithms functions (Table 2 -Algorithms Key Concept D, Learning Objective #18, 18c) or explain how they believe they will implement the algorithm using StarLogoBlocks (Table 2 - Algorithms Key Concept B, Learning Objective #16, 16b).

5.2.8 Quality Indicators

In order to facilitate additional assessment of creativity beyond the specific evidence of learning behaviors explicitly stated in the AP CSP course framework, design aspects of this module were created to correlate to specific quality indicators.

This module has students design their games from the ground up, choose their own genre of game, and have no boundaries other than a theme to jumpstart their thought processes so that the quality of their imagination can be assessed. The instructor is able to see what ideas the student's imaginations lead them to as they rely on themselves to make their own path. Activities such as the game narrative plays a dual role in giving students a basis for the character design process, but also provides a chance for them to exercise their inventiveness and show evidence of their imagination through the quality of their story.

The peer and instructor review sessions of this activity serve to provide students with the opportunities they need to get help and feedback in order to debug issues they run into, but it also acts as a quality indicator for mental flexibility. The instructor is able to evaluate how students react when sharing ideas with their peers. Do students

exhibit the mental flexibility to consider new ideas or feedback they get from their peers, or do they display rigidity and have less capacity to change perspective? During instructor review sessions, students may also be given suggestions regarding their design based on problems the instructor may see with certain choices. Are the students able to understand these suggestions and seek alternatives? Being creative also requires a sense of fluidity and the ability and desire to revise and build upon previously generated ideas (Alonso-Geta).

Lastly, all of the different design phases and design documents that students will create throughout their lesson plan will be tangible evidence of their ability to synthesize their ideas. The structure of this module was created to aide students through the entire design, development, and testing phases of their game, but this process also relies of the students' ability to synthesize their ideas. For example, can students go from their narrative and solidify what characters and objects they need to have in their game and outline what functionalities they will need to program for them? Can students take their ideas of what they behavior they want to occur in the game and synthesize it into an outline for an algorithm? From one step to the next, students work on their ability to synthesize their ideas and organize their thought processes.

At the end of this module, the instructor is able to evaluate how imaginative the students were since they have to apply their own innovations to design their game specs, assesses how flexible students were with their ideas, and have a record that demonstrates their ability to organize their ideas from one step to another. These three aspects serve as quality indicators for creativity.

5.3 Creativity: Development

The majority of development for the Creativity module was related to creating the structure of the lesson and supporting student materials. This module was extensively evaluated after initial development, which led to many cycles of returning back to the design stage, re-designing, re-developing, and re-evaluating. In this process, the frameworks developed to guide students through the different stages of game design were continuously re-worked. Components were added, removed, or modified to make

the processes more clear, intuitive, and effective.

As the student and teacher materials were being developed for this module, it became clear that for many sections it would be useful for the teacher to have a demo to show the students. Perhaps an instructor might desire to give a demo of implementing and testing a procedure of a certain functionality type so that students better understand the concept. It may also be desirable to show students examples of what is expected of them in their design documents. Students may ask questions such as “What exactly does it mean to list purposes and functionalities for a character?” or “How detailed is the description of an algorithm supposed to be?” The best way to show students what is expected of them is to show an example. Consequently, I developed a simple pirate ship game in StarLogoTNG following the framework I created, and completed the student design documents in correlation to that game. Now instructors have a working game and its code to use for demos, as well as the corresponding design documents for the purpose of discussing examples of what each section requires.

Lastly, a starter file was created to go along with this module that contained some already programmed procedures that may be helpful to students so that they are not starting completely from scratch and have some common functions created for them. The idea of having a starter file was based off of previously developed StarLogoTNG lessons where it was helpful for students to have some form of initial facilitator for creating their games. The starter file is also useful by including procedures students might want to use so that they do not have to program them. For example, the file contains a “rectify collisions” procedure that uses a recursive call to another procedure that changes the position of an agent if it happens to be generated on a coordinate with another agent. This function is useful because having an unwanted collision between two agents, although rare, can occur during setup and will produce undesired effects or glitches in the game.⁶

⁶ This problem was discovered while developing my pirate ship demo game. Upon setup, an EnemyShip agent happened to be generated to be on top of the main player, instantly killing the MainShip and ending the game as soon as the game began. Alternatively, at times an EnemyShip and Rock agent occupied the same position, making it impossible for the main player to obtain the treasure that would be dropped at that same coordinate, since the MainShip will die attempting to get the treasure since a Rock is also in that spot.

5.4 Creativity: Implementation

A modified version of this lesson plan was implemented through a workshop with eight male students. Table 5 shows the distribution of ages and prior experience of each attendee. Some attendees were younger than desired, which was still beneficial in judging whether or not students of varying ages could grasp the concepts in this lesson. Three attendees had experience with both programming and TNG, three more had experience with some form of programming but not with TNG, and two attendees had never been exposed to either.

	Programming Experience		StarLogoTNG Experience	
	YES	NO	YES	NO
11 years old	✓			✗
11 years old	✓			✗
12 years old		✗		✗
13 years old	✓			✗
13 years old	✓		✓	
13 years old	✓		✓	
14 years old	✓		✓	
14 years old		✗		✗

Table 5: A table showing the relative experience and ages of test subjects for the Creativity module

Not only would the varying age ranges allow me to judge the suitability of the lesson for middle-school and younger high-school aged students, but the varying experience of the attendees will allow me to pay attention specifically to the following issues.

1. Would students who already had experience with programming and StarLogoTNG find the framework of the lesson unnecessary or tedious? This activity has suggested pre-requisites that require exposure to some StarLogoTNG and programming concepts, and will ideally be implemented with a class of

students who have used StarLogoTNG before and who have done some basic programming. The experience of the 3 students who had experience with both StarLogoTNG and programming can be expected to be similar to those that this module is meant for.

2. Would the created design framework and idea of incorporating demos help students who have experience with programming but not with StarLogoTNG understand the software and basic capabilities well enough and quickly enough to be able to be successful with this lesson plan? Many pilot AP CSP courses use many different programming tools, so it is possible that this lesson may be implemented with students who might have learned some programming already, but are using StarLogoTNG for the first time.
3. Would the created design framework be able to help guide students who even had no programming experience and no experience with StarLogoTNG? The performance of such students during the workshop could provide some measure of how effective the designed framework may or may not actually be.

The format of the 5-hour workshop tested specific aspects of the module, as opposed to the entire lesson plan, in order to fit into the allotted time. In addition, the potential needs of students who had no prior experience with StarLogoTNG or with programming would have to be addressed, else they would not be able to participate. The workshop was broken up into two major sections. The first section was conducted in a group and served as a tutorial and introduction to StarLogoTNG, as well as a presentation of the character design framework and functionality types, and algorithm design framework.

First, students were oriented about how to use StarLogoTNG and then introduced to a modified version of the Pirate Ship game that was created for demonstration purposes with this module. The game had all characters and objects already created, and the setup function written. No behavior was implemented, yet the goals of main character, MainShip, were filled out on the accompanying worksheets. Together as a group we brainstormed what functionalities MainShip would need to have in order to achieve its goals. The idea of functionality types was introduced and explained, and students were called upon to classify each functionality that we came up

with. The procedures were then implemented as a group. I coded specific lines of that would be useful in describing features of StarLogoTNG and explained different aspects of the software as I programmed those pieces. All other times, students were called upon to finish coding a procedure and place it in its proper location in the program. As each volunteer came up to enter code, I narrated what they were doing and re-iterated related programming and TNG concepts. This same process was repeated for the remaining characters in the Pirate ship game; as group students came up with what functionalities the other objects would need to have, classified them, and a volunteer programmed each one.

The second section of the workshop consisted of students doing game programming individually instead of as a group. Even though this module is designed to have students work in teams to facilitate sharing of ideas and lightening the workload involved, students worked individually during the workshop. One reason for this decision was that due to the varying ages and experience levels of the attendees, it was not possible to make groups where students were evenly matched. Without evenly matched teams it would be difficult to evaluate what the success or failure of any group may have been attributed to. However, students were free to still share ideas, interact with, and help each other. A copy of the Pirate Ship game created during the first section of the workshop was transferred to each machine. Students were instructed to modify this basic game to make the current basic game more difficult and/or interesting, while applying the processes used in the first section of the workshop. If students created a new breed, they were to set its purposes, decide on its functionalities, and classify each one. For each new behavior added to the game, students were to write a description of the algorithm, then code and test it.

At the end of the workshop, students were able to spend time playing each other's games as they were also interviewed one by one about their experiences.

6 DATA MODULE

In this module, students will address the Data learning objectives in Table 8 through completing data analysis activities that require them to use StarLogoTNG to process information and generate large samples of data, which they can then use to generate information and obtain the knowledge they will need to complete the tasks of the lesson plan. Simultaneously, students satisfy the Abstraction learning objectives in Table 6 through the fact that the data they obtain will be done by interacting with a simulation. Specific tasks require that students use the simulation to test hypotheses, and additions to the model. These tasks are further detailed in section 6.2.2. Finally, this module can address the Impact learning objectives in Table 7 by tying together how using simulations to generate data and further analyzing that data has led to innovations and new discoveries in other fields of science. This is further discussed in section 6.2.4.

6.1 Data: Analysis

Data was selected to be a main topic for one of the 3 modules because of the realization that a data analysis activity would be an interesting and operational approach to bridge key concepts of Abstraction and Impact, two other Big Ideas of the AP CSP course.

StarLogoTNG's effectiveness as a tool for creating models and simulations is the perfect channel to introduce students to one of the key concepts of Abstraction which, as detailed in Table 6, is the idea that models and simulations can be used to help raise and answer questions.

ABSTRACTION		
Key Concept C : Models & simulations use abstraction to raise and answer questions.		
Learning Objective 9	The student can use models and simulations to raise and answer questions	
	<i>Evidence of Learning Objective</i>	9a. <i>Use of models and simulations to generate new understanding and knowledge.</i>
		9c. <i>Use of models and simulations to formulate, refine, and test</i>

		<i>hypotheses.</i>
		9d. <i>Use of simulations to facilitate testing of models.</i>

Table 6: A table showing the learning goals of Abstraction selected for the Data module.

As evidence for one learning objective of this key concept of Abstraction, the AP CSP framework outlines that students should be able to use a simulation to generate new knowledge and formulate, test, and refine hypotheses. In contemplating what kind of simulation could be designed to achieve these learning objectives and keep the learning process stimulating, the simulation for this activity must be something that can be linked to another topic that has possible interest or meaning for the students. This can be achieved by a simulation centered on a problem that can be paralleled either to interesting real world problems or to fields of study outside of computing; the opportunity to link Abstraction to Impact is now presented. The following learning goals of Impact were then selected for this module.

IMPACT		
Key Concept A. Computing affects communication, interaction, and cognition.		
Learning Objective 28	The student can analyze how computing affects communication, interaction, and cognition.	
	<i>Evidence of Learning Objective</i>	28b. <i>Explanation of how widespread access to information facilitates identification of problems, development of solutions, and dissemination of results.</i>
Key Concept B. Computing enables innovation in nearly every field.		
Learning Objective 29	The student can connect computing with innovations in other fields	
	<i>Evidence of Learning Objective</i>	29a. <i>Identification of the impacts of computing on innovation in other fields.</i>
		29b. <i>Description of how computational approaches and data analysis enable innovation.</i>

Table 7: A table showing the learning goals of Impact selected for the Data module.

The component of the lesson plan that bridges the selected key concepts of Abstraction and Impact should be part of a route where students use a simulation to generate some information, make meaning of that information, and transfer that meaning in order to solve a problem that has parallels to the real world. In other words, this bridge would have to be an activity where students are analyzing data and applying and testing the knowledge they obtain from their analyses. As a result, this module would have to include key concepts of Data.

After laying out the learning objectives for this module, which initially began with the idea of utilizing the characteristic of StarLogoTNG to create abstractions, the end of this phase led to the conclusion that Data would be the main focus of the module. Data contains many learning objectives that can interlock ideas of Abstraction and Impact, and Section 6.2 explains how the module was designed to incorporate the learning objective presented in this section of Abstraction, Data, and Impact

DATA		
Key Concept A. People use computer programs to process information to gain insight and knowledge.		
Learning Objective 10	The student can use computers to process information to gain insight and knowledge.	
	<i>Evidence of Learning Objective</i>	10a. Use of computers to find patterns in, and test hypotheses about, digitally represented information.
		10b. Drawing of insight and knowledge from translating and transforming digitally represented information.
		10c. Explanation of connections between information and knowledge.
Learning Objective 11	The student can communicate how computer programs are used to process information to gain insight and knowledge.	
	<i>Evidence of Learning</i>	11b. Use of accurate and precise language, notation, or visualization

	<i>Objective</i>	<i>to describe patterns or hypotheses arising from digitally represented information.</i>
		11c. <i>Summary of insight and knowledge resulting from translating and transforming digitally represented information.</i>
Key Concept B. Computing facilitates exploration and the discovery of connections in information.		
Learning Objective 12	The student can use computing to facilitate exploration and the discovery of connections in information.	
	<i>Evidence of Learning Objective</i>	12c. <i>Use of computing tools to discover connections in information.</i>
		12d. <i>Use of computing tools to extract information and knowledge.</i>
Learning Objective 13	The student can use large datasets to explore and discover information and knowledge.	
	<i>Evidence of Learning Objective</i>	13a. <i>Use of large datasets to extract information and knowledge.</i>
		13b. <i>Explanation of how large datasets can facilitate exploration and discovery.</i>

Table 8: A table showing the learning goals of Data selected for the Data module.

6.2 Data: Design

6.2.1 Concept Activation

The idea of data stands to be a very boring concept for students. The word “data” prompts thoughts of numbers and tables, graphs and plots, equations, reports, and other reflections that may bring to a student’s mind tedious topics or tasks that they have had to complete in other courses. Students need to begin this module by engaging in an activity that will bring their minds away from any monotonous experiences they may associate with data analysis. This concept activation activity would have to peak student interest by displaying the power of data and demonstrating that interesting and

important discoveries can result from the analysis of large data sets.

The suggested choice for the concept activation activity is a video that examines the progression of the relation between average lifespan versus income for all the countries of the world between 1810-2010. The presenter in the video effectively describes and presents the data in such an interesting and useful visualization that students see various trends and interesting facts about the world emerge that they may not have known. Such a video stimulates interest in data analysis by showing that relevant, interesting, and applicable information and trends can be discovered by analyzing a seemingly boring bunch of numbers.

6.2.2 Simulation & Activity Theme

The goal of this lesson is for students to experience how models and simulations can generate data, and how that data can be analyzed to help create knowledge. In addition, students must experience how this knowledge can lead to finding unknown answers to questions and discovering solutions to problems. As another design goal, the theme of the simulation and student activities also needs to be devised in such a way that at the end of the lesson connections can be made between what students have learned and experienced, and real world applications and other fields of study. This theme must also be somewhat interesting and engaging, and contain some motivating factor for students to actually solve the problem and feel a sense of achievement.

With the popularity of zombie apocalypse movies in current-day pop culture, which center on the spread of diseases and the extinction of the human race, combines with the merit of continuous accomplishments in medical research concerning genetic markers for diseases, the idea of a turtle epidemic – which is present in one of the introductory StarLogoTNG lessons instructors may use in classrooms to get students acquainted with the software – that needs to be stopped became the chosen theme of the simulation. In this lesson students will work in teams and play the roles of scientists who must use a simulation to discover a cure for a virus epidemic. Teams will run simulations to collect data, analyze that data to discover information needed to create and test solution to the epidemic and combat problem of preserving the turtles' lifespans.

The idea is that if students discover enough information to figure out the effects

of three specific genes, they might be able to come up with a plan that will be added to the code of the simulation and utilizes “gene replacement therapy”, along with vaccination and yearly treatment of turtles, and see whether or not the results of the simulation imply that their plan will be effective in maintaining the turtle’s lifespans. In addition, each method has a certain cost value associated with it per turtle, and the potential plan must also have a projected cost value that stays within a certain budget. Using data that the edited simulation now will generate, students will have to consider trade-offs of different decisions and apply evidence-based arguments to test the model they have created to save the turtles in order to find a good solution. Students have a motivational end goal of saving the turtles and the model will be designed to have more than one valid solution, which introduces another slight motivational and competitive aspect of desiring to find a solution that is more successful than others.

Throughout this activity, teams will also engage in sharing ideas and class discussions that can help guide students toward the proper intuitions needed to successfully complete this activity.

6.2.3 Applying AMT & Peer Instruction

This lesson plan incorporates the AMT process as students acquire data by running experiments with the simulation, make meaning of that data by analyzing results, and transfer their understanding by leveraging previously attained knowledge to create a plan that will help restore the average lifespan of the turtle population. In addition, students engage in peer instruction at various steps throughout the lesson as teams will discuss their conclusions and present their supporting results with each other before the instructor explains the solutions.

6.2.4 Impact

The lesson will conclude with a discussion that relates key concepts of data to other fields of study and real world applications. The AP CSP learning objectives laid out in Table 7 are addressed through presenting students with examples of how use of models and simulations to generate and analyze data has led to phenomena such as developing technology that can help predict earthquakes to save lives, the discovery of genes that make people more susceptible to certain types of cancer which helps in them

being able to take precautionary measures, and engineering related innovations. These discussions need to be effectively facilitated by the instructor so that connections between AP CSP concepts and what the students have learned and experienced are focused on, rather than just simply giving them new information without highlighting its meaning.

6.3 Data: Development

The development of the StarLogoTNG simulation for the Data module was simple in the fact that only behavior for the two entities of Turtles and Virus needed to be simulated. Therefore, it was easy to outline what functions needed to exist in the simulation. At the same time, development became complicated in the fact that given the randomization associated with the many variables of the simulation, the right balance of parameters needed to be found such that even with results varying within a certain margins, the overall outcome each time would be the desired behavior that would allow students to come to specific deterministic conclusions.

The following details describe how the turtles in the simulation functioned:

- They are randomly assigned one of the Gene A variations, one of the Gene B variations, and one of the Gene C variations.
- Based on its Gene A variation, each turtle is given a certain lifespan.
- Based on its Gene B variation, each turtle is assigned its susceptibility value. This is the percent chance that an uninfected turtle will become infected if it gets exposed to the virus.
- Based on its Gene C variation, each turtle is assigned its recovery ability value. This is the percent chance that an infected turtle receiving treatment will recover at the end of each year.

The virus on the other hand, has much simpler attributes. The virus is either active or not active and beings as not active until the user uses the “Introduce Virus” button (see Figure 5) to infect SpaceLand.

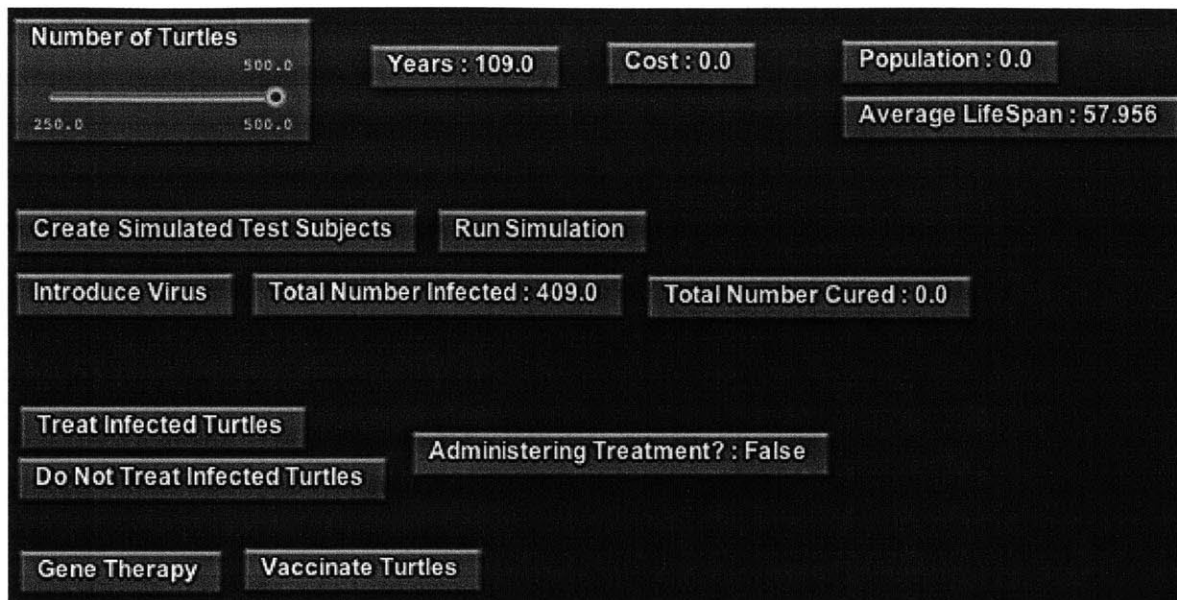


Figure 5: Buttons and monitors that students interact with during the data analysis activity

As the simulation runs turtles move around randomly living their lives, as does the virus if it is active. The simulation runs according to the following rules:

- If the virus is introduced, any turtle that collides with it is exposed to the virus
- If an uninfected turtle collides with an infected turtle, the uninfected turtle is also exposed to the virus
- When a turtle is exposed to the virus a random number between 1-100 is generated. If this number is less than their susceptibility value, the turtle becomes infected.
- Every year—designated in the code by when the clock value is an integer—infected turtles lose 1 year off of their lifespan. This means that infected turtles die twice as fast and uninfected turtles because as each year in the simulation passes, a sick turtle becomes 2 years closer to the end of its lifespan, as opposed to just 1 year.
- If turtles are being treated, every year a random number between 1-100 is generated for each infected turtle. If this number is less than their recovery ability value, the turtle is cured, gains immunity, and cannot become

infected again. If not, the turtle remains infected and loses 1 year off their lifespan since it is sick.

From a user interface point of view, the buttons that call the necessary functions needed to interact with the simulation are already present in the SpaceLand window, as seen in Figure 5. The only additions students make to the actual contents of the execution panel is when they are tasked with created some new graphs or plots to go along with news experiment that they will design to confirm some of their hypotheses about the turtles' genetics, such as the one shown in Figure 6.

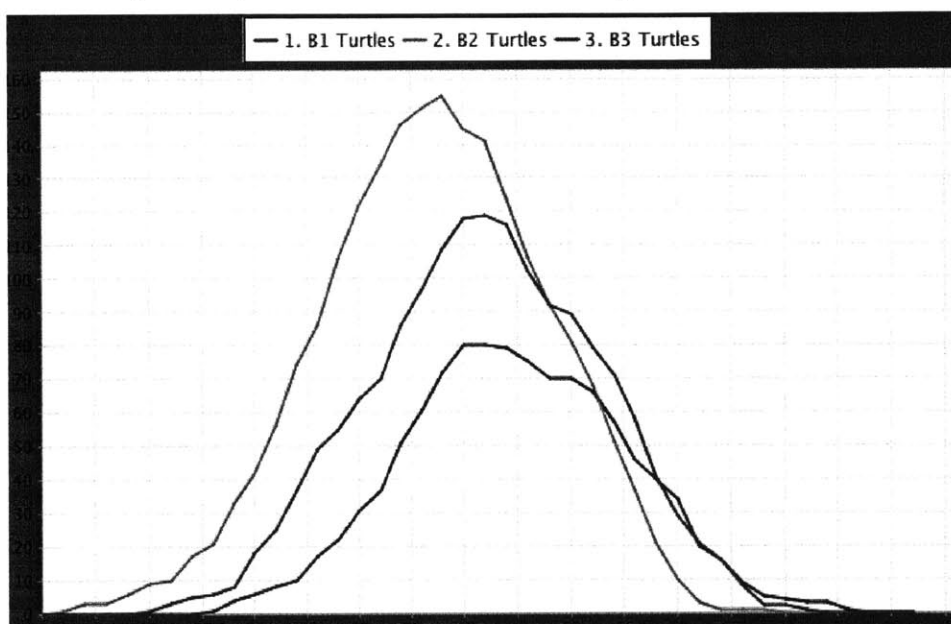


Figure 6: An example of a graph that students may add to the model using the information they are given in an attempt to plot data that clearly shows the distinctions between turtles with different Gene B variations living in an infected environment.

The framework for applying gene replacement and vaccination is also included in the code in order to create an abstraction that only needs students to focus on which genes they want to replace and which turtles they want to vaccinate. Creating a “Replace Gene” procedure that students only needed to enter parameters for and place in the “Gene Therapy” button did achieved this. In addition, a “Vaccinate” procedure was created so that in order to vaccinate turtles, students only needed to place if-commands in the “Vaccinate Turtles” button that uses the condition to designate which turtles to

vaccinate, and makes a call to “Vaccinate” in order to vaccinate those turtles.

The simulation monitors relevant values such as the current population, current year, a running measure of the average lifespan of turtles, average age infected of turtles with a specific gene, and the cost value associated with using treatment and/or vaccination and/or gene replacement therapy as part of a solution.

6.4 Data: Implementation

This lesson plan was implemented at two different stages. The first was through informal testing with one student, and then more formal testing through a workshop with 5 students and 1 teacher.

6.4.1 Data Module Implementation Phase 1

The first phase of implementation was conducted with one male 14-year-old student. This student had completed eighth grade would be entering ninth grade the following school year, placing him at an ideal age for getting a feel of how older middle school students and younger high school students might react to the lesson plan. This student also had no previous experience with StarLogoTNG or programming.

The purpose of implementing this lesson plan with such a student would be to gauge the intuitiveness of the activities. The performance of a student that is old enough to have the critical thinking skills needed for this lesson is ideal for evaluating the overall structure of the lesson. For example, were the problem statements and goals clear enough? Would the student be able to correctly analyze the results of the experiments to deduce what each turtle gene controls and what each of its variations results in? Are certain instructions too ambiguous?

Concurrently, the performance this same student who has absolutely no experience with StarLogoTNG or programming allows an evaluation of the learnability and usability of the accompanying TNG materials. Specifically, the goal was to get a sense of whether or not the tasks that required students to make additions to the code were not framed strongly enough. Although this lesson is intended for students who have had basic StarLogoTNG and programming exposure, programming is not intended to be a heavy focus of the lesson. It is only a tool needed to accomplish some of the tasks

in the lesson. Therefore, I believe this TNG materials should be flexible and intuitive enough that the addition of demos and quick tutorials by the instructor throughout the lesson can be enough for students with no experience to be able to not become completely hindered or frustrated by the minor programming aspects of this lesson.

The student was guided through the activities and given additional necessary demos and explanations regarding how to use the software. Working on his own on an activity meant to be conducted in pairs, the student was asked clarifying questions when he appeared stuck or ask to verbally explain his current through process in order to stimulate critical thinking thought processes that may have been activated he had a partner to discuss ideas with. In addition the student received programming assistance on occasion, when he was able to specifically dictate in human language what he wanted to place in the code – showing understanding of the necessary concepts – but got confused about the specifics of which TNG blocks to connect to execute his idea.

Upon completion of the activity this student was interviewed to gage intuitiveness off activity and materials. However, the transfer of certain abstraction or impact concepts were not evaluated because the student only went through the data analysis activity of the module, and the additional discussions and activities that tied together key concepts of Abstraction and Impact were excluded.

6.4.2 Phase 2

The second phase of implementation of this lesson plan was conducted with a version of the lesson that had been improved based on the results of Phase 1 implementation through a workshop. The attendees were 5 students ranging from ages 13-16, and 1 teacher. The lesson was conducted in pairs. Two male students, aged 13 and 14 who both had StarLogoTNG and programming experience were paired together. One 16-year-old male student and one 16-year-old female student who had no StarLogoTNG experience and no programing experience were paired together. The teacher, who was new to StarLogoTNG desired to do more than observe the lesson and desired to be paired with a 13-year-old male student who had both StarLogoTNG and programming experience.

The lesson plan was conducted almost in its entirety; as specific sections had to be omitted or modified in order to complete the lesson within the time frame of the

workshop. The optional peer instruction sessions of the lesson were not conducted, the analysis of Gene C was done together as a group as opposed to in pairs, and there was only enough time for teams to implement and test 1 or 2 solutions, as opposed to several. Students were asked to complete a short survey at the end of the workshop that gaged interest in the lesson plan, difficulties faces, as well as questions that tested what information or concepts they were able to take away from the lesson.

7 INTERNET

Previous exploration of course syllabi from high school AP CSP pilot courses revealed that when it came to the big idea of Internet, it was only after students were taught about the structure of the Internet and/or its elements and fundamental concepts that more interactive elements, such as making websites and participating in blogs or online forums, were introduced into the classroom. This module presents a lesson plan that serves to engage students as they learn about the nuts and bolts of the Internet, providing a medium for them to discover information on their own as opposed to being fed the information through traditional instruction.

7.1 Internet: Analysis

Internet was chosen to be a focus of one of the 3 modules because of the simple fact that the Internet is such a pervasive aspect within the lives of the current student generation. Students use the Internet for schoolwork, for entertainment, and to connect with each other through various mediums. An article from the New York Times dated in 2010 announced that the amount of time people spend on the Internet has increased by as much as 121% between 2005 and 2010 alone. However, although widely incorporated into many pieces of their lives, many students are never taught about the structure of the Internet or about how the Internet functions on a high-level. This module was intended to help teach students the basic concepts behind the Domain Name System, the resulting characteristics of the Internet, and what those characteristics contribute to.

INTERNET	
Key Concept A. The Internet is a network of autonomous systems.	
Learning Objective 24	The student can explain the abstractions in the Internet and how the Internet functions.
	<i>Evidence of Learning Objective</i>
	24a. <i>Explanation of how the Internet connects devices and networks all over the world.</i>
	24c. <i>Description of evolving standards that the Internet is built on, including those for addresses and names.</i>
	24d. <i>Identification of abstractions in</i>

		<i>the Internet and how the Internet functions.</i>
Key Concept B. Characteristics of the Internet and the systems built on it influence their use.		
Learning Objective 25	The student can explain characteristics of the Internet and the systems built on it.	
	<i>Evidence of Learning Objective</i>	25a. <i>Identification of the use of hierarchy and redundancy in the Internet.</i>
		25b. <i>Description of interfaces and protocols that enable widespread use of the Internet and systems built on it.</i>
Learning Objective 26	The student can analyze how characteristics of the Internet and systems built on it influence their use.	
	<i>Evidence of Learning Objective</i>	26a. <i>Explanation of how hierarchy and redundancy help systems scale.</i>
		26b. <i>Explanation of how interfaces and protocols enable widespread use.</i>

Table 9: A table showing the learning goals of Internet selected for the Internet module.

In order to form a lesson plan around the goals, a model that simulates the DNS lookup process will be suitable for students to discover for themselves how the process works, and be guided through activities that allow them to come to conclusions about the benefits of the structure of the Internet, as well as learn about potential weaknesses that can occur as the result of hacking. Given that students will be experimenting with a simulation to generate their understanding of Internet key concepts, this lesson also touches upon the same learning objectives of Abstraction presented in Table 6 of Section 6.1 regarding using simulations to raise and answer questions. (Table 6 – Abstraction Key Concept C, Learning Objective #9: 9a, 9c, 9d).

Lastly, a lesson that teaches the ideas behind DNS and discusses its benefits should also do justice in discussing its weaknesses. The concept of DNS hacking can be introduced by and having students experience its effects in the simulation. This would lead to being able to open up a discussion that touches upon how computing can have

harmful effects, an aspect of another key concept of Impact.

IMPACT		
Key Concept C. Computing has both beneficial and harmful effects.		
Learning Objective 30	The student can analyze the beneficial and harmful effects of computing.	
	<i>Evidence of Learning Objective</i>	30a. <i>Evaluation of legal and ethical concerns raised by computing-enabled innovations.</i>

Table 10: A table showing the learning goal of Impact selected for the Internet module

7.2 Internet: Design

7.2.1 Concept Activation

In order to stimulate interest in the subject, the beginning activity of this lesson attempts to stimulate curiosity regarding what actually happens when we surf the web. As part of a class discussion students will give their views on what they believe is involved between the point where we type in a website name into our browser, and the point where the desired webpage pops up on our screen. Some students may have no idea, and at this point the instructor doesn't mention which students may be wrong or right so that the curiosity of students who have never learned about the Internet carries on in the lesson.

This discussion then becomes an opportunity to teach students about URLs and IP Addresses and introduce the idea that URLs are generalizations for those addresses. This information that students are presented with identifies one of the many abstractions in the Internet (Table 9 – Internet Key Concept A, Learning Objective #24: 24d). Instructors can now go into more detail to teach students about IP Addresses, such how they are structured and how they represented in binary code. Students will transfer their understanding of these ideas by decoding and IP Address presented to them in binary, entering the decoded address into a web browser, and see that it actually takes them to a webpage. The instructor can now also teach students about network IDs and host IDs, which gives a better picture of how machines on the Internet are connected.

However at this point in the lesson, the concept of the Internet is still somewhat of a black box. Students now have an understanding of how computers are addressed, and how devices within the same network are connected through understanding host IDs and network IDs. Yet, the actual process of how a web browser figures out what the IP Address associated with a URL is, and how that process relates to getting content from the web server with said IP address is still vague.

7.2.2 DNS Lookup Activities

In order for students to now understand how devices on the Internet connect with each other (Table 9 – Internet Key Concept A, Learning Objective #24: 24a), the simulation students will interact with in this lesson needs to have the following features:

1. Depict how computers take website names and request their IP Addresses in order to know what the address of the computer they want to contact is. This feature will allow students to discover the function of a router or a DNS server.
2. Depict how computers are connected to routers, and allow students to discover how the router functions as a messenger between their computer, a DNS server, and the rest of the Internet.
3. Connect feature #1 and feature #2 to model the process of a computer requesting an IP Address of a web server, and the two machines relaying a message back and forth.
4. Simulate the effects of caching in order for students to discover its effects and benefits.
5. Simulate a website server changing its physical location but not its IP Address and simulate a server changing its IP Address so that the students further understand how devices on machines are connected, as well as further their understanding of caching to see what role the process plays in either scenario.
6. Simulate what happens when a new website is added to the Internet.

Along the way, students will be asked apply their understanding of their discoveries to answer questions regarding what would using the Internet be like if DNS did not exist, and how structure of the Internet lends itself to making the world wide web more usable

for humans, and incredibly scalable. These assignments and the resulting discussions will address the learning objectives of being able to explain characteristics of the Internet and how those characteristics affect its use. (Table 9 – Internet Key Concept B, Learning Objective #25: 25a, 25b; Learning Objective #26: 26a, 26b).

7.2.3 Applying AMT

The AMT framework is applied in this lesson within the concept activation and simulation activities. During the concept activation section of this lesson, students are acquiring knowledge about IP Address. They learn what they are, what they do, what they mean, and how they are structured. Here, they acquire the basic idea of how devices on the Internet connect with each other, which can be summed up as

“Each device on the Internet has an IP Address, that dictates its location on the Internet. Devices on the Internet connect with each other by using each other’s IP Addresses to send messages to each other, just like humans use each other’s phone numbers to call each other, or home addresses to send mail.”

Through interaction with the simulation, students will begin to make meaning of their understanding of IP Addresses. The features of the simulation will help students visualize and understand on a high level the concepts behind routers and DNS servers and the roles they play in the process of computers using the abstraction of a website name to obtain an IP Addresses and relay messages back and forth. After gaining this deeper understanding of the roles of IP Addresses and the DNS lookup process, students will transfer their understanding by being asked to complete the code to add a new website to the Internet. The procedure to add a new website will already exist, however in order to send messages to the website, students will need to come to the realization that the IP Address of the new website must be added to the DNS server’s list of addresses.

7.2.4 Impact

After learning about the benefits of DNS, students will use the simulation to discover how hackers can exploit the system if they are able to gain access to the DNS server. The simulation will have a new feature where a malicious website is added and “poisons” the DNS server, making it return the IP Address of the malicious website.

Students will observe what happens when the malicious site is added and further transfer their understanding by coming up with an explanation for what could have happened to the DNS server. To understand the real world implications of such an event, students will learn about one such attack on MIT servers that occurred in January 22nd, 2013 and discuss the legality of the issue. For example, should hacking be considered illegal? Does it depend on the situation? Does it depend on the intent? This final discussion results in students analyzing potential harmful effects of computing (Table 10 – Impact Key Concept C, Learning Objective #30: 30a) as they explore their opinions on when, if ever, hacking should be deemed either illegal or rather just mischievous but not against the law.

7.3 Internet: Development

A previous StarLogoTNG networking simulation created by Daniel Wendel and Wendy Huang of the MIT STEP Lab heavily guided development of this simulation. The interactive program that they developed established a precedent for creating entities that were connected by links, and routed messages to each other that would propagate throughout the network. This simulation in itself had more detail than necessary for the purposes of my Internet module, as the lower level details of routing would be abstracted in this simulation in order to not detract from the big picture of how the Internet works. However, this simulation provided me with some knowledge regarding how to program the behavior of transferring messages across links.

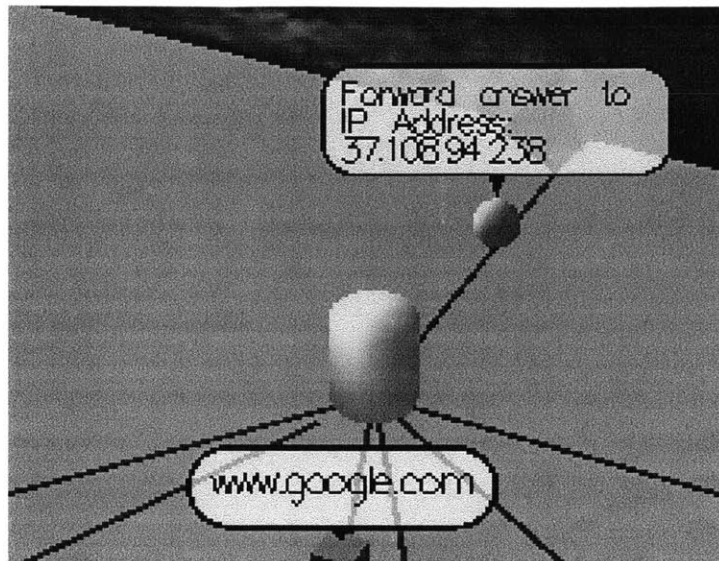


Figure 7: A message is forwarded to a website using its IP Address

The following figure depicts the initial basic setup of the simulation. An object that represented the user's home computer was connected to a router, which in turn connected to the DNS server as well as other websites. These objects were connected through other objects that served as links. Messages traveling on the network were represented as little spheres, color-coded based on what kind of message they were, and displaying the "message" they were sending, such as "Give me the IP Address of www.mit.edu." or announcing what IP Address they were currently traveling to.

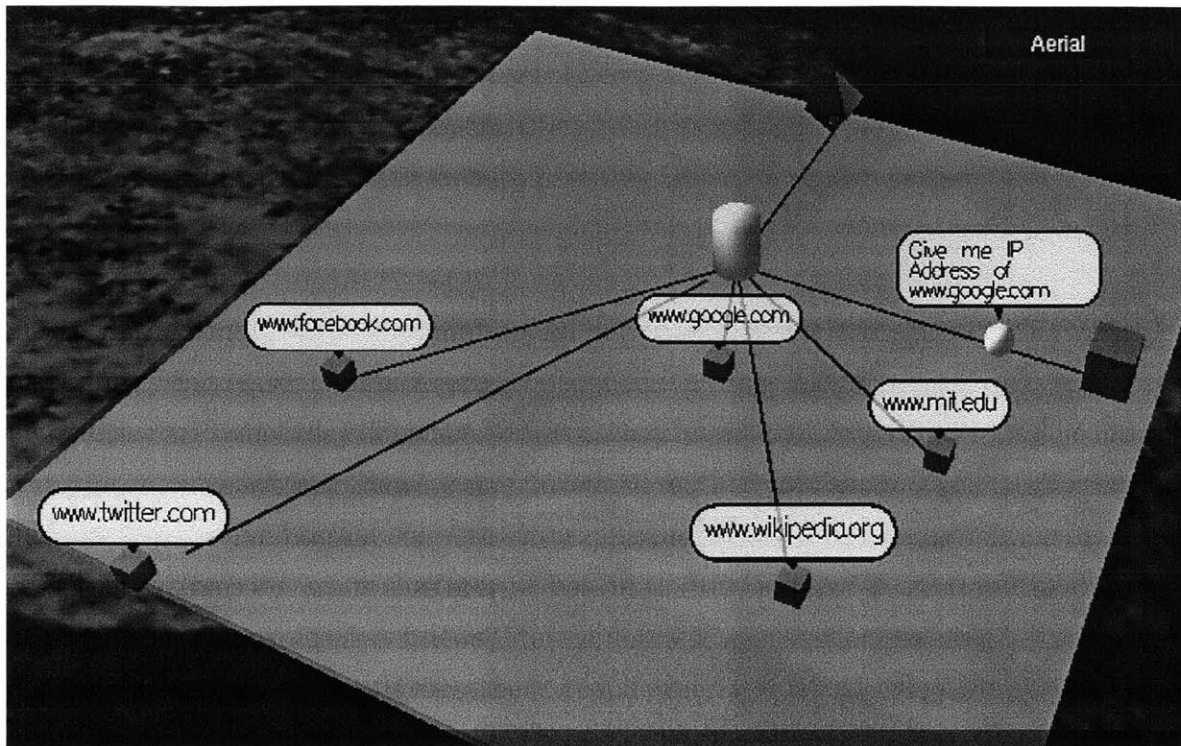


Figure 8: The home computer (diamond) is requesting from the DNS server (large cube) an IP Address of a website (small cubes). All requests are transferred through the router. (cylinder)

Buttons were used to interact with the simulation in order to send messages to a certain website, turn caching on and off, clear the cache of the home computer, change a website's IP Address or physical location, add a website, or insert a malicious website into the simulation.

7.4 Internet: Implementation

Unlike the Creativity and Data modules, the Internet module could not be implemented in the form of a workshop. In Section 8.3, the evaluation of this module is based on some general concepts that were discovered from observing how students responded to the other two modules, as well as discussing future work related to how to effectively test the module and what useful knowledge could be gained from that approach.

8 RESULTS, EVALUATION, & FUTURE WORK

8.1 Creativity

8.1.1 Results

The results of the creativity workshop revealed that the lesson should come with greater flexibility in terms of the tasks students are asked to complete during the game development process. As a group the eight boys were capable of setting the purposes of a character, coming up with the character's functionalities, classifying those functionalities, and being able to describe the algorithms for those functionalities before they coded them. However, when the students were left alone to their own devices to create their own extensions of a game, the design and planning related tasks they were asked to complete were helpful for some but proved to be of no interest to others.

The three students who had previous experience both with StarLogoTNG and programming took an approach to the designing their games that was the complete opposite of what it was intended that they do. They were very excited to start making additions to the game due to the various ideas they had thought of during the group sessions and immediately jumped into code. They each followed a pattern of taking an idea and attempting to program it. If they got stuck on the code for a particular idea they either asked for assistance from myself or from their peers, or completely scrapped the idea and tried something else. In addition, not all features that they managed to add to their game ended up in the final file they had at the end of the workshop. At times, they would succeed at programming a new feature within the game, and then decided to go a different route and try something else. Only towards the end of the workshop did these students begin to make an attempt to fill out some sections of the design documents, lackadaisically recording the additions to the game they chose to keep.

Next, the other set of three students who had previous programming experience—one with Python and 2 with Scratch—but no exposure to StarLogoTNG were able to be just as active as the experienced students when it came to creating their games. They found the demonstrations during the first half of the workshop helpful as it allowed them to “learn controls and how to use it [StarLogoTNG]” and become accustomed to the software. These students had a slightly different take than the 3 more experienced students regarding the design and planning frameworks. They found that it

was helpful to them, but only to a certain extent. The student who had used Python but was new to StarLogoTNG stated that writing down descriptions for his algorithms was “helpful if [he] didn’t already know how to code it”, but not if he already knew what to do because then he just wanted to immediately program it. Similarly, one of the other students who was new to StarLogoTNG but had used Scratch before felt that it was only necessary “to write things out at the beginning” when he was coming up with his initial new functionalities for the characters and thinking of the necessary algorithms for them. As he became more comfortable and solidified his ideas regarding what he wanted to implement, he did not see the need think through future additions. His brother, who also had the same previous experience with Scratch found that all he needed to do was think through what he was going to do, i.e. what functionality to add, what code blocks he would use to create the algorithm, and where he needed to place the code for it to be executed. Essentially, he followed the framework mentally and didn’t find it necessary to record his process on the design documents.

Lastly, the two students who had no previous programming or StarLogoTNG exposure appeared to benefit the most from the structure of the design process. Unlike the other 6 students, they tended to not ask other students for help as much and instead would either ask me for assistance or review the functionality-type handouts for help pertaining to how to code a certain type of procedure after they classified it. Regarding the written tasks, like the other 6 students they did not tend to want to follow the method of clarifying design details before implementation, however their reasons differed. One of the boys said it was only helpful for him to write things out when he first started, while the other student did not take to the design documents because of slight confusion as to what exactly was expected of him, even though we had done examples as a group. This particular student who appeared more confused than others was the only student who felt that he “needed that structure in order to begin to think about things” and that it was “helpful all the way throughout”. He did not find the design sheets tedious because he found it advantageous to keep a record of what he wanted to implement and keep track of his thoughts.

Overall, all of the students felt as if they were able to be creative even though the workshop was modified to have them build upon an existing game instead of building their own games from scratch, as designed in the full version of the Creativity module.

All students, even the less experienced students, felt pleased with their projects and enjoyed having others play their game, and playing each other's games.

8.1.2 Evaluation

During the game-programming workshop the following design aspects of the Creativity module were tested:

1. Character Design Process
2. Functionality Classification
3. Algorithm Design
4. Programming & Testing

These parts of the module were designed with the intent to effectively guide students through the task of programming a game. It was meant to allow them enough room to be creative yet help them organize their creative thoughts and create a roadmap for them to be imaginative yet efficient. The structure was also designed to include tasks that satisfy some of the AP CSP learning objectives for Creativity, Algorithms, and Programming. The results from the workshop showed that this framework proved to be helpful for some students, slightly tedious for other students, and flat out undesirable for others.

The previous experience of the students factored in to how they perceived the lesson. All of the attendees with previous programming and StarLogoTNG experience did not take to the first 3 design steps listed above and just wanted to code, test, and keep what they liked. While it was exciting to watch these students be incredibly inventive and constantly come up with new ideas, completely ignoring the other design steps resulted in what could translate to a lot of wasted class time due to spending stretched of time implementing ideas they have not committed to, or running into roadblocks that could have been avoided through initially outlining an algorithm.

Other attendees who had programming experience and just had to get used to how to use StarLogoTNG and become accustomed to the StarLogoBlocks followed through with some of the design steps, but varied in which tasks of these steps they completed before attempting to program their idea. For example, the character design process was skipped if they already had a full understanding of a new character they wanted to add, and functionality classification and algorithm design were seen as

unnecessary if they believed they already knew which commands and StarLogoBlocks they would need to combine. It is possible that these students may not have felt this way if they had to start from a blank file instead of basic game that has already taken care of a huge chunk of the design process. Nevertheless, this implies that the structure of the module should have more flexibility such that design tasks can be modified by instructors based on the prior experience of their class, so that aspects that may be tedious for students who let's say, only need to decided what procedures they will need to implement but are familiar enough with programming or StarLogoBlocks to skip the description of the algorithm.

The experience of the student who had a more difficult time than others, even more so than another attendee who like him had no programming or StarLogoTNG experience, proves that the framework of this module as it exists serves it purpose as a guide for students to turn their creative visions into a functional game, by helping them outline the steps they would have to take. This student who had the most trouble believed that with more time he would have been able to figure out more ideas and get them working because the design sheets helped him outline what it was that he needed to do.

8.1.3 Future Work

Future work for this module would include re-design, re-development, and re-implementation from the character design process through the algorithm design process. Although specific tasks in these sections were included to address specific AP CSP learning goals (See Section 5.2), it is not worth making the process wearying for students who are familiar enough with StarLogoTNG or programming to not have to outline their algorithms or classify a character's functionality in order to know how to code and test the procedure that implements it. At the same time, it is not worth completely removing the structure from this module because it is useful for less experienced students.

As a result, each section of the design phases in the Creativity Module need to have specific suggestions for modifications based on how much prior programming experience the class has had, and whether or not the students will be using StarLogoTNG for the first time. Each modification suggestion would also include which

learning objectives it may skip over, so that instructors can make a more informed decision about whether or not to include or exclude certain modifications.

This module would then need to be implemented in an actual classroom setting and conducted in full. It would be useful to try the following classroom settings:

1. A class where students have no previous programming or StarLogoTNG experience, and no modifications are made to the framework for the design processes.
2. A class where students have had programming experience but are new to StarLogoTNG, using modifications such as more demos being given or cutting out the algorithm design process, which may prove tedious for students who might mentally already understand what they need to do but are more focused on which StarLogoBlocks they need to use, and therefore want to jump straight from character design to programming and testing.
3. A class where students already have StarLogoTNG and programming experience, using modifications that go from character design straight to programming, and focus even more on creativity.

Testing the module in full with larger sample size of students and with specific modifications that can be made to make the activities more suitable for their experience levels would provide a deeper look into the effectiveness of how well the framework allows students to be fully engaged in their creative expressions while still following a structure that will help them create a functioning game.

8.2 Data

8.2.1 Results – Phase 1

The first implementation of this lesson plan, which was conducted as a 1-on-1 with a 14-year old male student with no prior experience in programming or with StarLogoTNG, proved very useful. The student was able to complete the tasks of the module, although they required occasional tutorials on StarLogoTNG and assistance with translating verbal ideas such as “I want to make a graph that shows all the infected turtles of each Gene B variation” into code.

8.2.2 Evaluation – Phase 1

Other than the student revealing where certain instructions needed to be clearer, the following points sum up the most notable realizations from this test that resulted in re-development of some of the Data module materials:

1. The minimum sample size of turtles to be created in the simulation had to be increased from 100 to be at least 250. With only 100 turtles in the environment, the virus did not spread as rampantly, and effects on the lifespans of the turtles were not always distinct enough for the student to come to a clear conclusion, such as in Figure 9.
2. The lesson needed to present students with a clear list of possible choices for what health component each of the three genes could correspond to, as well as their definitions so that students do not have to try to come up with these terms such as “susceptibility” or “recovery ability” on their own. This was realized when the test subject was reading a graph that showed turtles with certain Gene B variation died off more quickly than others in an infected environment. Although the student was able to describe what he saw and realized some turtles were more affected by the virus than others, he struggled to come up with a concrete idea as to why. As I posed questions to the student attempting to steer him toward the conclusion that Gene B controlled the turtles’ susceptibility to the virus, he replied, “It could be anything” such as “something about how their organs react to the virus”. It was clear that without a specific list of health aspects, students would either get stuck on coming up with the name of an aspect of health to

attribute to the results of the data.

3. Lastly, this test phase confirmed that the simulation parameters were chosen well enough to produce results that allowed the student to come to the desired conclusions, aside from when a sample size that was too small was used.

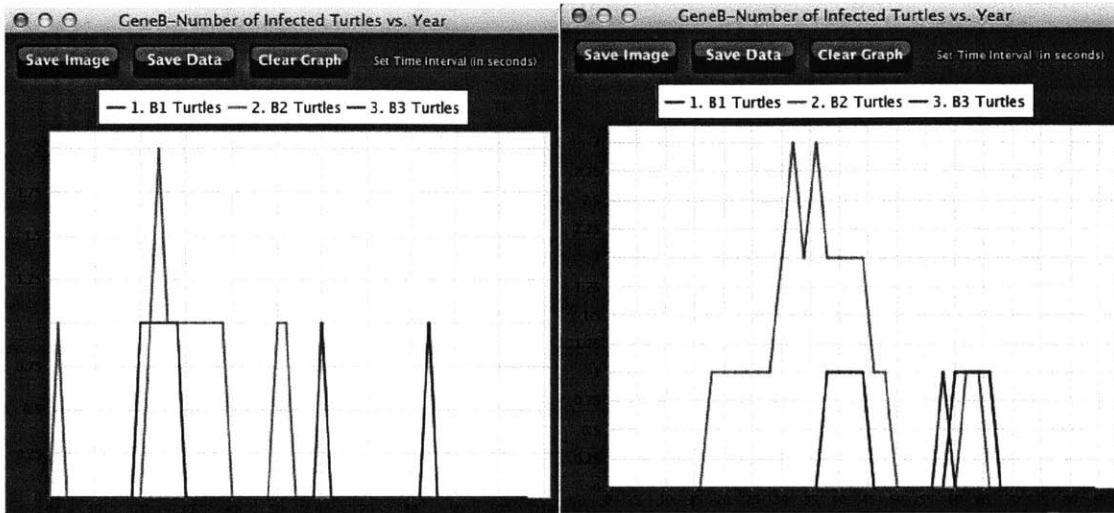


Figure 9: A diagram that shows that running the simulation with 100 turtles leads to graphs that aren't very conclusive. These side-by-side comparisons of the results of the same graph generated on two separate occasions using the same parameters do not contribute to students finding a clear trend.

8.2.3 Results – Phase 2

The students and teacher who attended the workshop decently received the test phase of the revised Data module, although there are still concerns that need to be addressed. Students described the lesson as “interactive”, “informative”, “educational”, “interesting” and “entertaining” and rated it an average of 4.2 on a scale of 1-5 for how interested they would be in doing such an activity as part of a class, with 5 being the most interested.

Age did not seem to be a factor in how well the lesson was received. The students ranged between the ages of 13-16, and each group was able to progress through conducting the experiments, recording their data, coming up with conclusions, and being able to verbally justify their decisions based on what they observed in the data. The one noticeable discrepancy would be that during the end of the lesson where

students had to combine what they knew about each gene to implement a plan to save the turtles, the team of two 16 year old students were much quicker at coming up with correct plans and testing them. Their first plan brought the average lifespan within 2 years of turtles living in a healthy environment, and adhered to the set budget. After that point, they had no trouble coming up with modifications to their initial fix to try to see what would work better. However, the team of two 13-year-old students generated a first plan that went way over budget and it took them some time to think of another one. At the end of the workshop they were still making changes to the model and running the simulation, while others had at least settled on one plan. It is unclear whether this was due to the fact that the 16 year-old students had sharper critical thinking skills due to their age, or if the two 13 year-old students were hindered by agreeing on what their next plan would be; they engaged in a lot of discussion with each other during this phase, but didn't manage to finishing implementing and testing another plan within the time allotted.

Experience with StarLogoTNG also appeared to be a non-factor in this lesson plan. The two 16-year-old students who had never been exposed to StarLogoTNG caught on quickly regarding how to use the software. Even though one of them had said that "it was hard to get used to the software". It is possible that their age also played a factor in being able to pick up how to use the software and connect blocks together to make code.

Aside from the success of the data analysis part of this lesson plan, the discussions regarding how examination of data and use of models and simulations within other fields of science and research have had an impact on society was well received. Students seemed very interested in learning that the idea discovering that just like how figuring out the effect of different turtles genes allowed them to make a plan to save them, that doctors do the same to new born babies to figure out if they are at risk for certain diseases so that the child can be appropriately cared for. The two 16 year old students seems the most interested in these discussions, as one revealed that he might be interested in programming but didn't actually know what careers programming could be applied to, and the other revealed that she had an interest in public health and biology and that it was interesting to see how technology and computing can be applicable to that field.

8.2.4 Evaluation – Phase 2

Evaluation for the second phase of implementation was heavily guided by the feedback of the teacher who attended the workshop. There still needs to be stronger scaffolding in regards to understanding the scenario presented at the beginning of the activity as well as understanding the relevant simulation variables defined for the students on a handout, and how to utilize them to make new graphs that plot new data to help them test and confirm hypotheses.

8.2.5 Future Work

Future work for this module includes reworking how the initial scenario that is presented at the beginning of the lesson in an attempt to be a more engaging way to present the students with all of the givens they need for the activity, is worked into the lesson. An approach that could be tried is to include new parts of the story as students move along in the tasks, reducing the initial cognitive load.

In addition this lesson should go through another implementation phase in an actual classroom setting of either a technology or math course where students have already been exposed to StarLogoTNG. This will allow for a larger sample size of students that will be more closely matched in age and experience. Such an experiment could be conducted in several hour-long sessions over the course of several days, so that the section of the lesson do not have to be truncated or modified to fit within a 4.5 hour workshop, and would allow for students to present their work in order to display their level of understanding by verbalizing their critical thinking process have time to debate their results with other teams and observe the effects of utilizing peer instruction in this module. This would also allow to address judge whether the scaffolding that exists in the lesson plan regarding the programming tasks that need to be done are sufficient for students who have used StarLogoTNG.

8.3 Internet

8.3.1 Future Work

Because my timeline of this project did not allow me to host a 3rd summer workshop, future work on the Internet module involves implementing the lesson plan in full with a group of middle-school to high-school aged students. Out of the 3 modules, the Internet module requires the least amount of prior programming knowledge and familiarity with StarLogoTNG. Consequently, I expect it to lead to an experience where students can really focus on the knowledge they obtain from interacting with the simulation. It would be beneficial to see how well this activity and features of the simulation allow students to grasp the key concepts of Internet chosen for this module by evaluating how effectively the lesson has succeeded in allowing students to gain a deeper understanding of the structure of the Internet and components such as DNS servers and routers. I believe more work should to be done on this module to include an additional activity that can be used to further test the transfer of knowledge before it is implemented with a class of students.

9 CONCLUSION

The increasing need for more K-12 students to gain an interest in computing and carry that interest throughout their academic lives to pursue careers in computing is the result of society moving deeper into an era where technology is incredibly pervasive within many aspects of our lives. Technology has countless impacts, both beneficial and harmful, within our society. And although computing has many applications to various career paths and fields of study, computing instruction and computational thinking is not a primary focus of the American education system. Students are also pushed away from nurturing potential interest in computing due to the subject being associated purely with programming, which can be very daunting from a student's perspective. The Advance Placement Computer Science Principles Course seeks to revise the way we teach students about technology and computing by creating a course that focuses on computer science and how it relates to the principles of creativity, abstraction, data, programming, algorithms, Internet, and societal impact. In addition this course aims to incorporate interactive methods of learning within its classrooms to motivate and inspire students.

This thesis outlines a 3-module set of lesson plans implemented in StarLogoTNG that align with the goals of the AP CSP course and are meant to assist in the instruction of certain pieces of the course's content. These modules were created by examining the learning objectives of the AP CSP course, drawing connections between the 7 different themes of the course, and creating activities that combine learning objectives of various themes into cohesive modules that can serve as the main materials for the instruction of specific AP CSP key concepts and learning objectives.

By synthesizing multiple learning objectives from different AP CSP big ideas into each lesson plan, but still having one big idea as the main focus of the module, the lesson plans become more cost effective as multiple learning objectives can be addressed through one large activity that builds upon itself. In addition, this characteristic of the modules allows students to experience how the principles they are learning are all related to each other, which helps to generate a deeper interpretation of these principles. By featuring learning frameworks that assist in teaching for understanding, the activities within these modules provide a medium for students to do more than just

acquire knowledge, but to make meaning of this knowledge and transfer their understanding of concepts to successfully complete the tasks of the lesson plan. Furthermore, the interactive and engaging nature of StarLogoTNG creates an environment where all of this learning can be done in setting that is more engaging than traditional learning approaches. By using StarLogoTNG to create games and interact with simulations, students can enjoy the learning process of even the most technical concepts, such as programming and algorithms, and be motivated to foster their computational thinking skills and become more proactive in their learning. Such an environment stands to increase the chance that more students will generate interests related to computer science and/or computing related fields.

References

- Alonso-Geta, Petra Ma Perez. "Researching, measuring and teaching creativity and innovation: a strategy for the future" Institute of Creativity and Educational Innovation, University of Valencia. Web. Accessed 21 August 2013. Referenced from ec.europa.eu/education/lifelong-learning/policy/doc/creativity/report/research.pdf
- "AP Computer Science Principles: Draft Curriculum Framework". CollegeBoard, August 2013. Web. Accessed 23 July 2013. Referenced from <http://www.csprinciples.org/home/resources>
- Beede, Doms, et al. "STEM: Good Jobs Now and For the Future." Economics and Statistics Administration, 14 Feb 2011. Web. Accessed 23 July 2013. Referenced from <http://www.esa.doc.gov/Reports/stem-good-jobs-now-and-future>
- Begel, A., Klopfer, E. StarLogoTNG: An Introduction to Game Development. 24 Dec 2004. Retrieved from <http://www.langwidge.com/starlogo.pdf>.
- Colella, Vanessa Stevens, Eric Klopfer, and Mitchel Resnick. *Adventures in Modeling: Exploring Complex, Dynamic Systems with StarLogo*. New York: Teachers College, 2001. Print.
- "Computer Science Principles: Big Ideas and Key Concepts, Learning Objectives and Evidence Statements" CollegeBoard, 5 Aug 2011. Web. Accessed 23 July 2013. Referenced from <http://www.csprinciples.org/home/resources>
- "Computer Science Principles: Big Ideas and Key Concepts, Learning Objectives and Evidence Statements" CollegeBoard, 21 Aug 2012. Web. Accessed 23 July 2013. Referenced from <http://www.csprinciples.org/home/resources>
- "Computer Science Principles: Computational Thinking Practices, Big Ideas, Key Concepts, and Supporting Concepts" CollegeBoard, 5 Aug 2011. Web. Accessed 23 July 2013. Referenced from <http://www.csprinciples.org/home/resources>
- "Computer Science Principles: Computational Thinking Practices, Big Ideas, Key Concepts, and Supporting Concepts" CollegeBoard, 21 Aug 2013. Web. Accessed 23 July 2013. Referenced from <http://www.csprinciples.org/home/resources>
- "Computer Science Principles: Course Annotations" CollegeBoard, 5 Aug 2011. Web. Accessed 23 July 2013. Referenced from <http://www.csprinciples.org/home/resources>
- "Computer Science Principles: Learning Objective". CollegeBoard, August 2013. Web. Accessed 23 July 2013. Referenced from <http://www.csprinciples.org/home/resources>

- Crouch, Catherine H., and Eric Mazur. "Peer Instruction: Ten Years of Experience and Results." *American Journal of Physics* 69.9 (2001): 970-77. Print.
- Cuny, Janice. "Transforming HS Computer Education." 18 Oct 2010. Web. Accessed 01 June 2013. Referenced from <<http://www.computingportal.org/node/3886>>
- Cutts, Q., and Simon, B. "How to Implement A Peer Instruction-Designed CS Principles Course." Inroads: Paving The Way Toward Excellence in Computing Education June 2012: 72-74. Print.
- Cutts, Q., and Simon, B. "Peer Instruction: A Teaching Method to Foster Deep Understanding." Communications of the ACM February 2012: 27-29. Print.
- Dick, W., & Carey, L. (1996). *The Systematic Design of Instruction* (4th Ed.). New York: Harper Collins College Publishers.
- Drake, L., Lacy, M.J., Merrill, M.D., Pratt, J., & ID2_Research_Group. (1996). "Reclaiming Instructional Design" *Educational Technology*, 36 (5), 5-7. Referenced from <<http://mdavidmerrill.com/Papers/Reclaiming.pdf>>
- "Employment Status of the Civilian Noninstitutional Population, 1940s to Date." *U.S. Bureau of Labor Statistics*. U.S. Bureau of Labor Statistics, 5 Feb. 2013. Web. Accessed 23 July 2013.
- Ingram-Goble, Adam. (2013) *Playable Stories: Making Programming and 3D Role-Playing Game Design Personally and Socially Relevant*. In review.
- Klopfer, E., Scheintaub, H., Huang, W., Wendel, D., Roque, R. 2009. "The Simulation Cycle - Combining games, simulations, engineering and science using StarLogoTNG." *E-Learning and Digital Media*, 6(1), 71-96. Referenced from <<http://dx.doi.org/10.2304/elea.2009.6.1.71>>
- Miller, Andrew. "Yes, You Can Teach and Assess Creativity!" *Edutopia*, 7 Mar. 2013. Web. 21 Aug. 2013. <<http://www.edutopia.org/blog/you-can-teach-assess-creativity-andrew-miller>>
- Paul, Jody. "CS Principles Pilot at Metropolitan State College of Denver." Inroads: Paving The Way Toward Excellence in Computing Education June 2012: 56-57. Print.
- "Where are the STEM Students?" *STEMconnector*. myCollegeOptions, 2012. Web. Accessed 22 July 2013.

Wiggins, Grant. "On Assessing for Creativity: Yes You Can, and Yes You Should." Web log post. *Granted, And... ~ Thoughts on Education by Grant Wiggins*. Wordpress, 3 Feb. 2012. Web. 21 Aug. 2013.
< <http://grantwiggins.wordpress.com/2012/02/03/>>

Wiggins, Grant and Jay McTighe. "What is Backward Design?" in *Understanding by Design*. 1st edition, Upper Saddle River, NJ: Merrill Prentice Hall, 2001, pp. 7-19.