Unplugged Robotics as a Platform for Cybersecurity Education in the Elementary Classroom

Keith Rand
krand@washoeschools.net

Washoe County School District
Reno, Nevada

Shamik Sengupta
ssengupta@unr.edu

David Feil-Seifer
dave@cse.unr.edu

University of Nevada, Reno
Reno, Nevada

Abstract – Robotics may be an ideal way to teach cybersecurity concepts to young students in the elementary classroom. Research shows robots can be an engaging experience and benefit learning in ways useful in other areas of education. Programming robots provides an ideal context for compelling demonstrations of cybersecurity concepts. Unplugged robotics activities benefit from the engaging aspect of robots but have the added advantage of bypassing hardware and making some concepts more transparent. Señor Robot is a gamified unplugged robotics activity modeled after some activities used before but specifically designed for cybersecurity education in the context of mathematics. The design and implementation of Señor Robot in a third-grade classroom is discussed along with observations and results of student assessments. Strengths and weaknesses of

Señor Robot are examined and guide a proposed revision of the game called Frogbotics. An expanded instruction set and applicability to English language arts are considered along with ways to use Frogbotics to teach specific topics in cybersecurity. A website is provided as a dissemination point for materials developed in the study.

Keywords

computer science; cybersecurity; education; elementary; Frogbotics; K-12; programming; robotics; Señor Robot; unplugged

1. INTRODUCTION

Teachers are always looking for a hook – something to catch the interest of students when the learning might otherwise be dry and difficult. What could be more dull and opaque than password security to a first grader or data integrity to a third grader? On the other hand, what could be more fascinating to the same students than robots they control? In a review of research by Lai Poh et al. on the use of robots in early childhood and lower level education, it was found that robots can provide an interactive and engaging experience [1].

The review also reports that robots and robotics activities can have a positive effect on collaboration and problem-solving skills, science learning, and language development [1]. Since young students are motivated to interact with and learn to program robots, and such activities can positively affect skills that enhance learning across the board, robotics seems like an ideal platform from which to teach young students cybersecurity concepts. Consider, additionally, that programming robots is intrinsically related to computer science and comprises a cyber-physical system ideal for demonstrating cybersecurity concepts.

Unplugged robotics activities, from simple to complex, have been used to teach programming concepts to students at every level (Table 1). When the availability of computers and robots is limited, unplugged activities can still be used to teach important concepts. Sometimes the concepts are more transparent to students when an unplugged approach is used. Young children are often so fascinated by robots that the mere mention of them or opportunity to role-play one can be as motivating as actually operating or interacting with one.

Table 1
Unplugged robotics activities used to teach programming concepts

Activity	**Level and purpose**
Harold the Robot [2]	Young students discover the constraints of communicating using a programming language.
Robots and Sequences [3]	Middle and high school students discover the importance of sequences in programming.
Karel J. Robot Introductory Role Play Activity [4]	High school and college students discover the nature of object-oriented programming in an introduction to the Karel J. Robot simulator.

This paper gives an account of an unplugged robotics game developed specifically for teaching cybersecurity concepts to elementary school students. The initial version of the game, Señor Robot, is described. It's use with a group of third grade students over a period of five months is discussed. After a summary of the results, a proposed revision of the game called Frogbotics is detailed along with ideas about how it could be used to teach specific cybersecurity concepts.

2. SEÑOR ROBOT

Señor Robot is a game modeled after other unplugged robot-themed activities used to teach programming concepts, particularly a classroom role-play activity designed by Joseph A. Tosh for introducing programming using Karel J. Robot [4]. Señor Robot is designed differently from other unplugged activities in two important ways.

First, Señor Robot's programming language is constructed so that the game can be easily integrated with mathematics curriculum. According to Grover and Pea, "Programming is . . . a key tool for supporting the cognitive tasks involved in [computational thinking]" [5]. It is hoped that Señor Robot will frequently bring to bear on students' math learning the benefits of these cognitive tasks. Furthermore, the number of opportunities to use the game for cybersecurity-related learning is increased by making the game adaptable to a core subject.

Señor Robot is also designed to go beyond the kinesthetic approach of similar unplugged computer science activities by instilling ownership of data that are eventually objects of cybersecurity demonstrations. Gamification of the activity by including time constraints, competitive factors, and rewards for reaching goals induces in students the attitudes necessary not only for making cybersecurity of interest to them but also for understanding how motivations shape and define cybersecurity [6].

2.1 Teams and roles

The robot role-play game is designed for four teams. Each team requires a minimum of two members. Students fulfill various roles according to the size of their team (Table 2). Programmer roles are combined in teams of fewer than five. Additional roles such as co-lead programmer and programmer/debugger can be added for teams of more than five. The student who takes on the role of the robot cannot communicate with the rest of the team by any means other than the commands received as input and the resulting robot behaviors as output.

Table 2
Student roles in Señor Robot

Role	**Function**
Robot	Moves the team's robot and performs other operations as commanded by the team
Lead Programmer	Makes the final decision about how to code each command message
Programmer / Code Writer	Writes down each command message
Programmer / Uploader	Delivers each written command message to the Application
Programmer / Time Keeper	Keeps track and notifies team of start of turn and elapsed time during turn.

The Application is the teacher or another person capable of checking each command message for lexical and syntactic errors (improper specification of programming commands), robot behavior for malfunctions (response errors), and outcomes for runtime errors (illegal or illogical outcomes). Errors require the Application to restore the game to its state prior to the offending command. Game play then continues with the next team's turn. The Application signals the start and end of each team's turn and tracks elapsing time during a turn. These functions can be performed with announcements or, to create a relatively silent game environment, signs or signals and a clock or timer with a large display.

2.2 Configuration and play

At the beginning of the game, each team's robot is positioned in the starting box of a separate path upon which the robot must remain while moving through the game space. Each path leads to four different tables where a team's data cards are stacked and to a goal box. Though the paths intersect, each team's path, robot, and table stacks are designated by a unique matching color so there is no confusion (Figure 1).

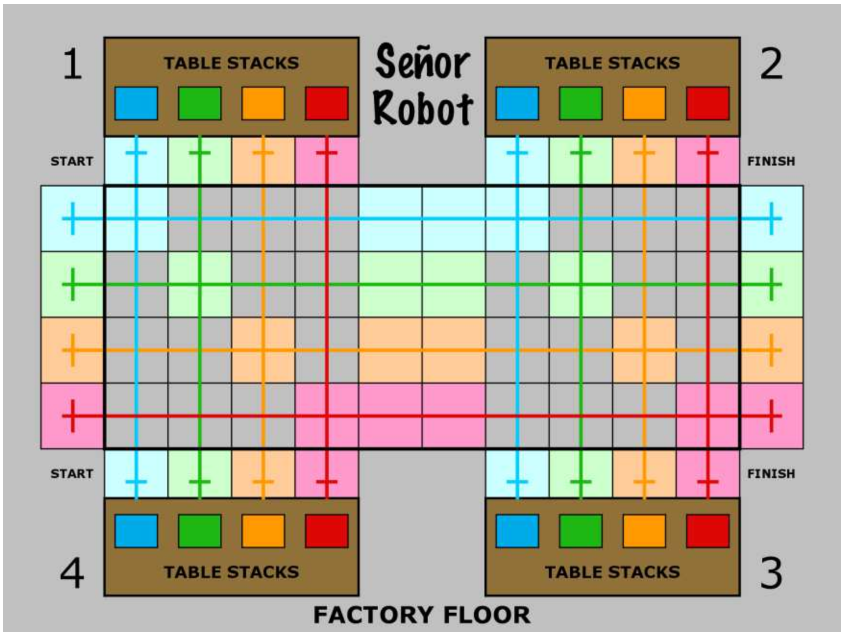


Figure 1: Game space of Señor Robot.

The separate but intersecting robot paths provide a compact way for four teams to be simultaneously engaged in the competitive activity. The number of boxes per path is the same, and rounds are used with one turn per team so there is no advantage of one path or starting order over another. Stopping or turning the robot is allowed only in boxes of the team's color (i.e., where only one color of path is present) so one team cannot hinder the progress of another. Each team's table stacks are filled with the same data cards, and the goal for all teams is the same. Ties can occur when multiple teams are successful in the same round. The result is that teams focus on achieving a goal with the fewest commands and the least mistakes rather than interfering with each other.

For grades K-2 and the beginning of 3rd grade, the game space may be laid out on the floor in the form of a play mat, and actual tables or desks and cards may be used. In such a version, students acting as robots would walk on the paths and pick up cards according to commands received. Because Señor Robot was first used with 3rd graders during the fifth month of the school year, it was deemed appropriate to use a wall version instead. Students assigned to be robots stood in line at the front of the room where they took turns receiving a command and then moving a robot graphic in a projected game space (Figure 2).

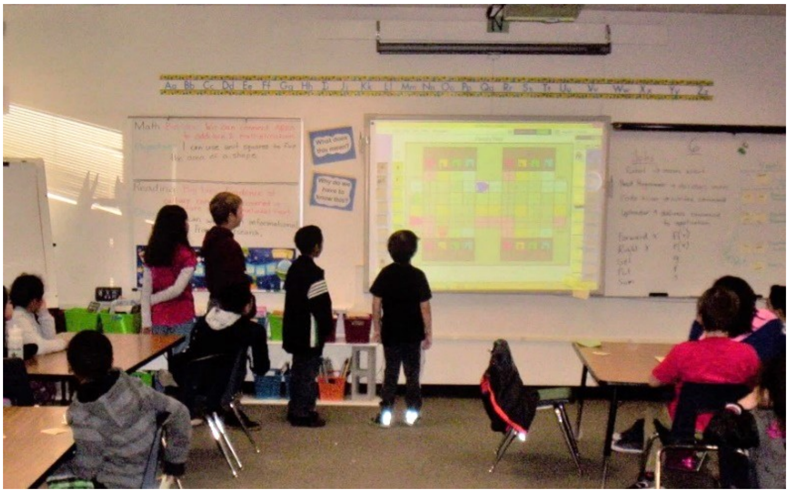


Figure 2: Students using projected wall version of Señor Robot.

The robot graphics and game space were elements of an ActivInspire™ flipchart page projected on an ActivBoard™. Data cards were Post-it™ notes applied to the board in the projected table stack spaces.

2.3 Programming language and standards-based implementation

A programming language with a low entry level was presented to students (Table 3). The parameters of *Forward* and *Right* were specified as the number of boxes and 90-degree right turns, respectively. *Put* was never used by students. *Multiply* was introduced after four months of game play.

Table 3
Señor Robot programming language

Command	**Use**	**Description**
Forward	f(x)	Move forward x boxes
Right	r(x)	Turn right 90 degrees x times
Get	g	Get a data card from top of adjacent table stack
Put	p	Put a data card on top of adjacent table stack
Sum	s	Say the sum of the numbers on data cards held
Multiply	m	Say the product of the numbers on data cards held

After initially presenting the commands and their definitions, students were questioned for approximately 15 minutes about what commands would be necessary to move a robot to certain boxes in the game space and what robot actions would result from given commands.

The next day, students were introduced to the rules and roles of the game. They practiced playing after being told the goal was to make their robot announce “six” while in the goal box at the finish. Each team’s table stacks contained data cards with the numbers 1, 3, 3, and 5. To achieve the goal the robot had to execute the *Sum* command in the goal box while holding both 3’s or the 1 and 5 (CCSS.MATH.CONTENT.3.NBT.A.2.). In all games after the practice game, each member of a winning team received points that were part of the school’s token economy and redeemable for toys at the school’s student store.

In subsequent games requiring the *Sum* command over the next two months, goals of announcing “four,” “eight,” and “eleven” were given. A time limit of 30 seconds was also imposed on each turn. If a team did not send a command to the Application and have it executed by the robot within the time limit, an error would result.

When the *Multiply* command was introduced, the goal of announcing “nine” was given to students. To achieve the goal using the fewest commands, the robot had to execute the *Multiply* command in the goal box while holding both 3’s (CCSS.MATH.CONTENT.3.OA.C.7).

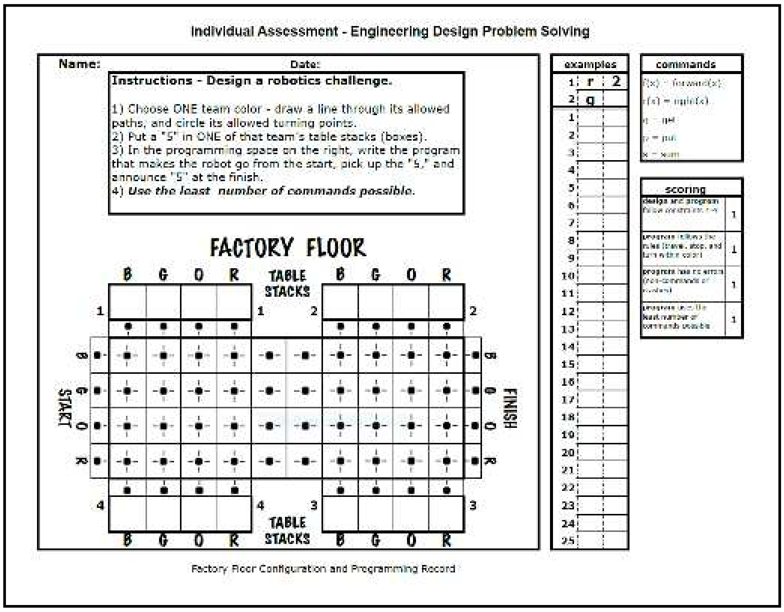
The time limit for each turn was then reduced to 15 seconds. After students learned first about multiplying three numbers together and later about multiplying a number by ten, data cards with the numbers 3, 5, 10, and 15 were used. The goal of announcing “one hundred fifty” was given. To achieve the goal using the fewest commands, the robot had to execute the *Multiply* command in the goal box while holding the 10 and 15 (CCSS.MATH.CONTENT.3.NBT.A.3 and CCSS.MATH.CONTENT.3.OA.B.5).

2.4 Paper and pencil assessments

Three versions of the game space were developed for paper and pencil. Each version included an area for specifying the goal and providing teacher post-comments, a diagram of the game space, and a section for recording commands.

The first version was given as an assessment to teams after playing the game three times in the first two weeks. The form given to each team showed the data in all of the table stacks for all teams, just as the game space did when it was projected on the interactive whiteboard. However, a different robot path was highlighted by the teacher on each team's form.

The second version (Figure 4) was given to students individually as a science assessment of engineering and design standards five weeks after introducing the game.


Individual Assessment - Engineering Design Problem Solving

Name: Date:

Instructions - Design a robotics challenge.

1) Choose ONE team color - draw a line through its allowed paths, and circle its allowed turning points.
2) Put a "5" in ONE of that team's table stacks (boxes).
3) In the programming space on the right, write the program that makes the robot go from the start, pick up the "5," and announce "5" at the finish.
4) *Use the least number of commands possible.*

examples		
1	r	2
2	g	
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
12		
13		
14		
15		
16		
17		
18		
19		
20		
21		
22		
23		
24		
25		

commands
f(x) = forward(x)
r(x) = right(x)
g = get
p = put
x = [illegible]

scoring	
design and program follow directions 1-4	1
program follows the rules (travel, stop, and turn with the color)	1
program has no errors (non-commands or mistakes)	1
program uses the least number of commands possible	1

Factory Floor Configuration and Programming Record

Figure 4: Version 2 for paper and pencil – an individual assessment.

Students were asked to design their own problem (NGSS: 3-5-ETS1-1.) by tracing a robot path of their choice and then placing a 5 in one of the connected table stacks, the goal being to announce "five" in the goal box. Then students were asked to provide the best solution to the problem (NGSS: 3-5-ETS1-2.) by writing the shortest possible sequence of robot commands that would achieve the goal. Correct sequences varied according to the placement of the 5 by each student.

The third version was an assessment given to teams after three months of game play. The format was similar to version 2, except the problem description was replaced with space for writing in students' names and the goal. Instead of providing a grading rubric, more space was provided for recording commands. Each team wrote identical problem constraints (path, data, and goal) onto the form using instructions provided by the teacher. The best sequence of commands for achieving the goal was therefore the same for all groups. This version was used to see if an assessment could be quickly created and administered.

2.5 Student population, study groups, and results

The research location was a K-5 public school that received Title I funding. All students received free breakfast and lunch. The school was also designated a "Zoom" school by the state and received grant funding allocated for underperforming schools. A majority of students at the school and in each classroom in the study performed below grade level on the most recent state and national standardized math and reading tests.

Both the projected wall version and the science assessment (version 2 for paper and pencil) of Señor Robot were used in each of the six third-grade classes. In all, 100 students participated with a balance of girls and boys who were socioeconomically disadvantaged and ethnically diverse.

2.5.1 Other 3rd grade classrooms

Teachers in five of the classrooms implemented independently after discussing the activity with the researcher and observing the projected wall version of the game in play in the researcher's classroom. Each third grader was given the science assessment by his or her own teacher for a science grade. The researcher received verbal reports about the game play and assessment in the other classrooms. The other teachers reported they had no difficulty in implementing the game and that all students were enthusiastically engaged and in the activity. Results of the assessment were similar to those in the researcher's classroom.

2.5.2 Researcher's classroom

The researcher directly observed the game play and assessment in his own classroom.

When the idea of the game was introduced to students, they were immediately excited. The projected game space caught the attention of the students and acted as a visual aide when presenting the initial five programming commands. About fifteen minutes was used to explain the commands and give examples of how they would be used to control the on-screen robot. Another fifteen minutes was used to question students about commands required for certain actions and the consequences of given commands. For some questions, students demonstrated the answer by coming to the interactive whiteboard to move the on-screen robot.

During the latter fifteen minutes, students were eager to raise their hands and answer questions. Students were called on at random using name cards, and eventually all students were called on. As many girls were observed raising their hands as boys. Students who were normally quiet or reserved were also anxious to answer questions. Questions were answered correctly by as many lower-performing students in math and reading as higher-performing students in those subjects, and by as many girls as boys.

Before the first game on the following day, students were told there would be no prize for winning and that it was a practice game. Students were still highly engaged. After explaining the illegal moves, the roles were listed on the board and described. Teams of four were formed by using name cards, however the students were deliberately placed to create heterogeneous groups. The teams were then told to decide who would take on each role and to send their robots to the front of the room. These methods of determining team membership and roles became standard procedure, and the groups were made different for each game.

The five commands initially presented were listed on the board with a description of their effects. In subsequent games, commands were not listed on the board. During the previous day's lesson students became aware that *r(2)* was a U-turn and that *r(3)* had the same effect as a left turn. This knowledge was retained and used by students during the practice game. Each team talked about the commands at its table and sent command messages written on Post-it™ notes to the teacher to be conveyed to its robot. The notes for each team were posted on the margins of the board to form sequences. Commands that resulted in errors were noted. After the third game, team command messages were no longer posted on the board.

During the second game, students were observed writing commands ahead of time on their own initiative. Teams lined up the Post-it™ notes in the order they were to be sent. Occasionally an error would occur and the line-up would be scrapped or adjusted. By the fourth game, at least three teams at a time in each game were observed writing sequences in advance.

In the first game that used the *Multiply* command, one team reached the goal box with the appropriate data but had trouble remembering they needed to use *m*. Another team succeeded before they figured it out because students had discovered the competitive advantage of secrecy.

The worst complication in all of the games was the high placement of the board and location of table stacks 1 and 2 in the projection. Shorter students found it difficult to reach high enough to take data cards stuck on that part of the board and to turn the robot around in the boxes adjacent to the table stacks. By far the most difficult skill challenges for students were consistently associating *r(2)* and *r(3)* with right turns rather than left and right turns with clockwise movements rather than counterclockwise. This difficulty was observed in girls and boys and higher- and lower-performing students and in the paper-and-pencil assessments as well.

The science assessment was scored on four factors: correct use of the design instructions, correct use of the programming language, using the fewest possible commands to reach the endpoint (right or wrong), and achievement of the specified programming goal. One point was awarded for each for a possible total of 4 points. Nine out of sixteen students in the author's class received 3 or 4 points. The remaining students caused runtime errors by specifying commands that led the robot off of its path. By far, the most common error was specifying *r(1)* instead of *r(3)*, i.e., a right turn instead of a left turn, or vice versa. The researcher attributes this to the difficulty of associating the correct turn when a student is tracing a horizontal path on the paper and then needs to turn toward a table stack that is either above or below (rather than to the right or left of) the tracing finger.

Teams were highly successful when completing the group assessments and always were able to reach the given goal, although occasionally not using the fewest possible commands. Evidently, one of the drawbacks of an unplugged approach is that it requires the authoritative supervision of a teacher or the savvy students in a group. If a student is working without supervision, as with an individual assessment, he or she is more susceptible to associating an incorrect behavior with a command, e.g., mistakenly turning *counterclockwise* three times to execute *r(3)*, because there is no feedback from a device that is wired or programmed to function correctly.

This indicates that an unplugged tabletop gameboard version would be problematic. On the one hand, it would not be appropriate for unsupervised practice by one student or a group of students susceptible to mistakes. On the other hand, the use of player groups that were heterogeneous by ability would lead to the same players always winning.

3. FROGBOTICS

The use of Señor Robot with third-graders has revealed strengths and weaknesses of the activity as a potential teaching tool for cybersecurity education. Frogbotics is a proposed revision of the game that incorporates the strengths of Señor Robot and corrects some of its problems.

Initially it was thought that the name Señor Robot would make the game more identifiable to ethnically diverse students. This seemed to be more of a limiting factor than a benefit. The revised activity's programming language is named FRoG after the core commands, Forward, Right, and Get. Activities using FRoG are called Frogbotics. These names have are ethnically and gender neutral.

To make FRoG a programming language with a high ceiling as well as a low threshold, extended commands are made available for English language arts and logic in addition to mathematics. Data can be letters and words as well as numbers. The *Join* command can be used to merge multiple data cards into one in order to spell words and make sentences. *Test* can be used to return TRUE or FALSE when a data card is checked for a given value.

Commands can be used in different forms, depending upon the level of the students and the core subject learning objectives. When word or single letter commands are used, they take on the default singular use of the command (e.g., *Right* or *r* means turn right once). Parameters with positive or negative values are available for most commands and modify the function of the command in ways that allow for a variety of problem-solving approaches. For example, *g(-2)* means take a card from the top of those held by the robot, put it on top of the adjacent table stack, then repeat those actions once more.

Each letter is reserved for a specific command as before. The *Put* command is retired since *g(-1)* is equivalent, and *s( ), d( ), p( ), and q( )* represent *Sum, Difference, Product, and* Quotient, respectively. A full list of proposed commands and their definitions is freely available at frogbotics.com.

To make left and right turns less confusing for young students, a beginner game space will be developed that is rotated so the starting boxes are at the bottom. This will put the table stacks on the right and left and the goal boxes at the top. When robots move toward the goal boxes, they will face the top of the game space – left will be left and right will be right. This will also enable shorter students to more easily reach data cards and turn the on-screen robots since all of the table stacks will be to each side of the paths rather than two sets of stacks being across the top.

Different game space designs can be developed for variety and for different abilities and grades. As PDF files, these are easily imported into ActivInspire™ pages and projected on an interactive or standard whiteboard or screen. The quick use version of the final paper and pencil version of Señor Robot was successful and can be adapted to Frogbotics. Various versions for paper and pencil can be generated to match the projected versions. Frogbotics.com will be developed to freely provide all of the downloadable materials mentioned in this paper and any updates and additions as they occur.

4. CONCLUSION

Señor Robot was a successful low-threshold inducement of students into computational thinking. Its structure is conducive to discovering and learning cybersecurity concepts. It instilled in students a desire to reach given goals and an ownership of created and accessed data. Students valued this data and recognized the advantage of secrecy in a competitive environment where there could still be more than one winner. This is the beginning of a successful cybersecurity curriculum.

Frogbotics promises to continue where Señor Robot leaves off. Elements of the revised and extended game environment will provide students with an opportunity to learn about passwords and authentication, verification of data integrity, and encryption. Privacy and secrecy will be seen by students as desired aspects of message delivery where strategy and competition are factors. Cyberattacks will be game events that disrupt the delivery of command messages in a number of ways that require countermeasures on the part of teams.

With an expanded set of commands adaptable to more academic subjects and more accessible and varied game spaces, students will have a greater opportunity to be creative and grow to their potential as problem-solvers. Teachers will be able to find resources for Frogbotics at a dedicated location online and benefit from continued research on the use of the activities in cybersecurity education.

This research is supported by NSF Award #1542465.

REFERENCES

[1] Lai Poh, E. T., Causo, A., Pei-Wen Tzuo, I-Ming, C., & Yeo, S. H. (2016). A review on the use of robots in education and young children. Journal of Educational Technology & Society, 19(2), 148-163.

[2] Nelson, Richard, Jason Clutterbuck, Sebastian Höhna, Stefan Marks, and Wilson Siringoringo. "Harold the Robot." *Computer Science Unplugged*. University of Canterbury, Christchurch, n.d. Web. 29 Mar. 2018. <https://classic.csunplugged.org/harold-the-robot-2/>.

[3] Feil-Seifer, Dave, Mercedes Anderson, and Blanca Miller. "Robots and Sequences." *Unplugged Robotics*. University of Nevada, Reno, n.d. Web. 29 Mar. 2018. <https://unpluggedrobotics.blogs.unr.edu/>.

[4] Tosh, Joseph A. "Karel J. Robot Introductory Role Play Classroom Activity." *Karel J. Robot Role-Play: Introductory Role Play Classroom Activity*. Ed. Joseph Bergin. Pace University - Seidenberg School of Computer Science and

Information Systems, New York, 1 Mar. 2003. Web. 29 Mar. 2018. <https://csis.pace.edu/~bergin/KarelJava2ed/KarelRolePlay.html>.

[5] Grover, Shuchi, and Roy Pea. "Computational Thinking in K—12: A Review of the State of the Field." Educational Researcher, vol. 42, no. 1, 2013, pp. 38-43.

[6] Agresti, W. W. "The Four Forces Shaping Cybersecurity." Computer, vol. 43, no. 2, 2010, pp. 101-104.