ALGORITHMS AND 2ND GRADE STUDENTS

AN ASSESSMENT OF ALGORITHMS LESSONS USED WITH 2ND GRADE STUDENTS

Paul Gross
Graduate K-12 Fellow
Center for Engineering Education Outreach
Tufts University

October 2005

1 INTRODUCTION

A primary goal of education is to produce people who can solve general problems. To this end, students are exposed to a number of problems and the appropriate tools to use in finding solutions to those problems. In terms of mathematical tools, each can be described as a process that guides a student from a problem description to a solution. For some students, these processes are viewed as *the* methods for finding solutions to a small set of well-defined problems. Consequently, some students develop the impression that every mathematical problem has one specific solution process for finding one correct answer. To them a lack of success in memorizing these processes means they cannot "do math." This impression of mathematics and, by extension, problems involving mathematics, can be a hindrance to the students' interest in mathematics and their ability to solve more general problems. In an attempt to mitigate this impression of mathematics and promote more general problem solving, the author suggests devoting time to the study of these solution processes, also called algorithms.

Having students study algorithms can provide them with a powerful notion of problem solving. By having students *develop* processes for solving problems, rather than having them *memorize* processes to solve problems, students can examine and practice the design of solutions. Such an open ended exercise promotes creativity and deep understanding of the problems being attempted. Students can begin appreciate the "why" behind the processes they use to solve problems, rather than just the applications. Repeated exercises like this can build an abstracted process for students on how to approach and build solutions to general problems, rather than repeated standard exercises which tend to solidify the solution process to one problem and generalize poorly (e.g., addition and subtraction of positive number to addition and subtraction mixed with negative numbers). In this way, students can develop a stronger notion of how to analyze an unfamiliar problem and work towards a solution for it.

The purpose of this study is to explore younger students' ability to understand processes and design algorithms in order to solve general problems. Four classrooms of 2nd grade students were observed over seven class meetings, five lessons, and two testing sessions, on the application and design of algorithms.

2 BACKGROUND

2.1 ALGORITHM DEFINTION

In a very general sense, an algorithm can be defined as a list of instructions to solve a problem. With this definition, we can consider an algorithm as a more general description of a solution to a problem, rather than the common conception of some process in math. Thus, "everyday" processes we follow to achieve tasks can also be called algorithms (e.g., the common analogy that a recipe is an algorithm). In using algorithms with 2nd grade students, this general definition of algorithm is used.

2.2 CLASSROOMS AND INTERACTION

Over the course of one school year, the author was a GK-12 fellow in four 2nd grade classrooms at the same school. For the rest of this paper, these classes will be referred to as Red, Yellow, Green, and Blue. The size of each class and its demographics can be seen in Table 1. All of these classes are considered to be of equal ability with the exception of the Red class, which is considered the "gifted"

class, as deemed by the school. Through the course of the school year, the author visited each classroom at least one time and at most two times a week. During each visit, a lesson or activity relating to computer science and/or mathematics was conducted with the students.

Table 1. 2nd Grade Classroom Demographics

	Num Students	F	M	African American	Hispanic/ Latino	Asian American	White
Blue	23	11	12	3	2	4	14
Green	23	12	11	4	4	5	10
Yellow	23	12	11	4	2	9	8
Red	24	12	12	1	1	1	21

3 STUDY DESIGN

3.1 PRETESTING AND POSTTESTING

Pretests and posttests were administered by the author during the first and last visits to the classrooms to assess the students' ability to sequence events, describe processes, identify how programs are made, recognize computer scientists, and define an algorithm. The pretest and posttest can be seen in Appendices A and B, respectively.

The first two questions focus on student's sequencing ability by giving a sequence of events pictorially, then another sequence textually, and asking students to number them in the order they must occur. Some students were expected to sequence with a high degree of accuracy on the pretest, but an even higher percentage of students would be successful in sequencing on the posttest. The reason for expecting this performance increase is the students would have studied the ordering of events in the algorithms lessons and, consequently, would have the benefit of more applied practice in sequencing.

The third question asks students to describe the process they follow to accomplish a common task through drawing and/or writing, essentially requiring them to create an algorithm. The purpose of this exercise is to assess their ability to write algorithms before and after intervention. It is expected that this ability will improve after the lessons and, therefore, they will write "better" algorithms (as defined in section 4.1.2) on the posttest.

Questions four and five are related in that they ask questions specifically related to ideas in computer science. The fourth question asks how a word processor they commonly use at school "learned" to check spelling and the fifth question asks students to describe who a computer scientist is. These questions are for survey purposes as how computer programs are constructed and the role of computer scientists do not relate directly to understanding algorithms, but are addressed in the lessons. The interest here is in whether or not students' impressions of programs and computer science change and, if so, how.

The final question asks students to describe what an algorithm is. It is expected most students will not define an algorithm correctly or give any specific properties of algorithms on the pretest, but will offer some properties of an algorithm, if not a full definition, on the posttest. In this way, we explore how well-developed the concept of an algorithm is with the students.

3.2 LESSONS

Between the pretest and the posttest, each class received the same set of five lessons, with the exception of the Red class. The Red class had been exposed to algorithms earlier in the year at the request of the teacher, who had helped in developing algorithm lessons the previous school year. The potential consequences of this prior exposure will be discussed in section 4.

During each lesson students were encouraged to write in an observation. Many students were unfamiliar with the concept of taking notes and were told they could write or draw anything they wanted that related to the lesson. To collect data about student development, students were asked to answer questions in their journal after each lesson. The purpose of these questions was to monitor students' understanding of algorithms and collect data specific to the lesson conducted to assess its effectiveness. Each question will be described with its accompanying lesson in the following sections.

3.2.1 ALGORITHMS IN GENERAL

The main purpose of this lesson was to introduce students to examples of algorithms and, thus, give them a concise definition of an algorithm. Two examples were used: how to get dressed in the morning, and how to make breakfast. For each example, students gave instructions for completing the tasks. Emphasis was put on the idea that there are multiple algorithms for solving each problem (e.g., you may make cereal for breakfast or oatmeal, and either choice would solve the problem of making breakfast) and the proper ordering of instructions (e.g., you put on your underwear before your pants). The previously discussed definition of an algorithm was given using the two examples to highlight the required problem and the list of instructions necessary to solve the problem. At the end of the lesson, students were asked to respond in their journal to the question, "What did you like about today's lesson?"

3.2.2 APPLICATIONS OF ALGORITHMS IN ORIGAMI

This lesson focused on applying algorithms, specifically viewing instructions to make an origami object as an algorithm. Students were asked to think about examples of algorithms and share them with the class. They were then given a piece of paper with picture instructions for creating a jumping frog using origami and asked if the instructions were an algorithm. After the class discussed their answers, the class followed the instructions with guidance from the instructor and created their own jumping frogs. Following the lesson, students were asked to answer in their journal the questions: "What algorithms did we use today?" and "What makes the frog jump?"

3.2.3 APPLICATIONS OF ALGORITHMS IN GAMES

The intent of this lesson was to expose students to more complex algorithms (i.e., those involving choice) and begin discussion of designing algorithms. The lesson starts with the students playing a binary search game with the instructor. The instructor selects a number between 0 and 100. The class wins if it can guess the number in seven guesses or less. With each guess, the instructor responds whether the number is greater than, less than, or equal to the number guessed. As the class guesses numbers, the instructor takes an adversarial role. The instructor does not actually chose a number, but chooses to respond "greater than" or "less than" according to which answer leaves the class with more potential numbers to guess. In this way, students only find the number in seven guesses if their algorithm for guessing is a binary search algorithm, which is optimal.

After playing this game once, the class is asked to describe the role of the process the instructor uses as an algorithm. The instructor states that they have given this algorithm to a computer, which will now play the game with them as a class (this concept is explored further in the next lesson). A screen shot of the computer program can be seen in Appendix C. Students are chosen at random to

guess a number and try to figure out the number the computer has chosen (although it has not chosen a number and plays the same adversarial role the instructor did). As students continue to guess, the instructor asks if certain guesses are better than others (e.g., if the last computer response was < 65, is 68 a good guess?). In continued playing of the game, students refine their guesses and better approach 7 guesses. Following a few rounds of the game, students answer these questions in their journals: "What algorithms did we use today?" and "How could we find the computer's number in less guesses?"

3.2.4 ALGORITHMS AND COMPUTERS

The goal of this lesson is to continue exampling applications of algorithms and connecting the concept of algorithms to computers. The lesson began with a demonstration of the spell checking feature of a word processor commonly used by the students in their computer class. The students were then asked answer yes or no to the following question, "Is ABC spell check an algorithm?", and provide one reason for their answer. After writing, students shared their answers with the class. The class then designed an algorithm for how the computer could check the spelling of a sentence typed into the word processor. It is then explained the computer "learned" how to check spelling the same way it "learned" to play the game played previously: someone input an algorithm into the computer. To this end, the students are shown the source code behind the game program used in the previous lesson to explicitly show the list of instructions used to make the computer play the game. Simple modifications are made to the source code to show the students that changing instructions changes how the computer plays the game. The class is then asked if it knows what types of people give instructions to computers to create programs. A computer scientist is one of the supplied answers either by the students or the instructor. It is further explained that computer scientists design algorithms often used in computers. Afterwards, the students respond to these questions in the journals: "What does a computer scientist do?" and "How did [the word processor] learn to check spelling?"

3.2.5 DESIGNING ALGORITHMS

The final lesson is a more of a guided work session than an activity. Students are asked to choose a problem and write an algorithm with at least 5 instructions that solves the problem as well as must draw a picture that illustrates their algorithm. Both of these are to be done in their journal, constituting a rough draft, and a final draft will later be typed and accompanied with a second illustration.

4 RESULTS

The major sources of data collected come from the pretests, posttests, answers to questions given at the end of each lesson, and students' algorithms. It is important to note that the Red class, also called the "gifted" class by the school, was exposed to algorithms (specifically, the lessons in sections 3.2.1 and 3.2.3) prior to this study and, consequently, its results will be considered separately where appropriate.

4.1 PRE- AND POST- TEST

For each test given, every question on the test was read to the students before they began, and any questions about a test were taken before the students began it. Students were encouraged to write down whatever they felt was correct, as they would not be graded; the purpose of the test was

to determine what they thought, not whether their answers were correct or incorrect. Students were told they could write "I don't know" if they felt unable to answer a question.

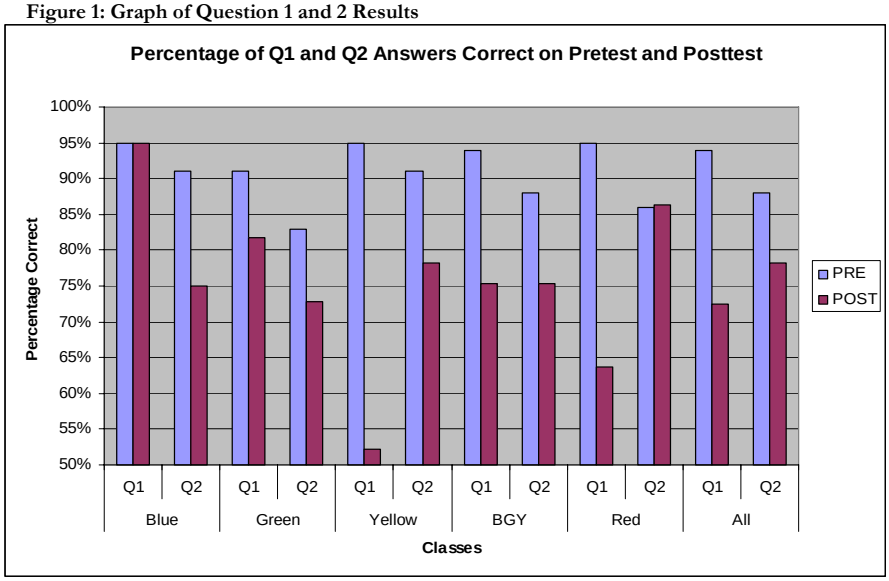
4.1.1 QUESTIONS 1 AND 2 – SEQUENCING

The results of Questions 1 and 2 for the pretest and posttest are summarized in Table 2 and Figure 1 (BGY stands for Blue, Green, Yellow, and is a combination of these classes).

Table 2. Results for Questions 1 and 2 of the Pretest and Posttest

	PRE	POST	PRE Q1		POST Q1		PRE Q2		POST Q2	
	(n)	(n)	Correct	%	Correct	%	Correct	%	Correct	%
Blue	22	20	21	95%	19	95%	20	91%	15	75%
Green	23	22	21	91%	18	82%	19	83%	16	73%
Yellow	22	23	21	95%	12	52%	20	91%	18	78%
BGY	67	65	63	94%	49	75%	59	88%	49	75%
Red	22	22	21	95%	14	64%	19	86%	19	86%
All	89	87	84	94%	63	72%	78	88%	68	78%

Figure 1: Graph of Question 1 and 2 Results



The pretest results are quite promising. They show that students handle these sequencing problems well enough, but still have some small room for improvement. However, it is clear from the results that students performed worse on the posttest questions than on the pretest questions: an unexpected occurrence. One possible reason for this result is that the questions were changed from pretest to posttest, which makes the comparison of these two questions similar to comparing apples and oranges. The process for solving the questions would be the similar, but more variance may have been introduced in changing the questions, thereby affecting the results.

Another possible reason for this result is ambiguity in the posttest questions themselves. An analysis of the incorrect answers given can be seen in Table 3. From this table, it is clear that more than 50% of the incorrect answers for each of the two questions can be attributed to two different incorrect answers. It is likely that the events taking place in the questions' were not clear, and this caused many students to answer incorrectly.

Table 3. Incorrect Answers for Q1 and Q2

Incorrect Answer		Frequency	% Incorrect Answers
POST Q1	1243	8	35%
	2413	4	17%
	2341	4	17%
Total		16	68%
POST Q2	53412	6	32%
	54312	5	26%
Total		11	58%

From this result we cannot conclude that the students' ability to solve sequencing problems has improved. However, it is further believed that their sequencing skills did not degrade during the course of the lessons, despite the indication otherwise, and that the posttest results are a reflection of both changing questions between tests and ambiguity in the posttest questions.

4.1.2 QUESTION 3 – WRITING AN ALGORITHM

The qualitative nature of Question 3 required the development of a scoring rubric in order to interpret the results. Algorithms are scored in three dimensions: completeness, coherence, and order. These dimensions were determined in reading a great number of 2nd grade algorithms and attempting to judge their quality. From these dimensions, a scoring rubric was developed, and can be seen in Table 4. The rubric allows for a minimum score of 0 and a maximum score of 9. Example algorithms from students' final assignments and their scoring can be seen in Appendix D.

Table 4. Scoring Rubric for Evaluating Student Algorithms

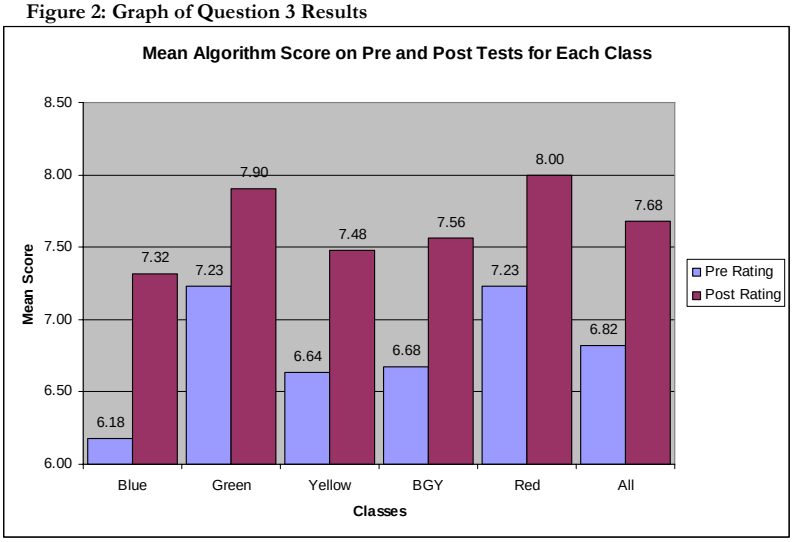
Dimension	Score	Criteria
Completeness	3	Algorithm solves problem stated completely
	2	Algorithm mostly solves the problem; a few issues raised by instructions are not handled
	1	Algorithm mostly does not solve the problem, most issues raised by instructions are not handled
	0	Algorithm does not solve the problem to any degree
Coherence	3	All instructions of the algorithm work toward a solution
	2	Most instructions work toward a solution, few are either not instructions or inapplicable to the stated problem
	1	Most instructions do not work toward a solution, most statements are not instructions or are inapplicable to the stated problem
	0	No instructions work toward solving the problem
Order	3	All instructions follow logically towards a solution
	2	Most instructions follow logically to a solution, few are out of order but mostly follow one another
	1	Most instructions do not follow logically to a solution, most are out of order or do not apparently follow one another
	0	No instructions follow logically towards a solution, instructions appear randomly ordered

In order to reduce bias in the study, evaluation of the algorithms using the rubric was not done by the author. The evaluation was conducted by an independent evaluator who evaluated both the pretest algorithms and the posttest algorithms (to ensure consistency in the application of the rubric) with no knowledge of who the students were or of which algorithms were from the posttest or pretest.. The results of the pretest and posttest algorithm evaluations can be seen in Table 5 and Figure 2. In viewing these results, it is important to note that the Red class's prior exposure to algorithms is a clear bias of its pretest results. It is suggested that the reader consider the BGY (Blue, Green, and Yellow) category to judge the overall improvement of the 2nd grade students rather than the All category because of this bias.

Table 5. Results of Evaluating Algorithms from Question 3

	Test	n	Mean Score	Std. Dev.	t-test n	p-value
Blue	Pre	22	6.18	2.11	16	0.004
	Post	18	7.32	1.36		
Green	Pre	19	7.23	1.15	19	0.069
	Post	21	7.90	1.09		
Yellow	Pre	22	6.64	2.11	21	0.015
	Post	23	7.48	1.36		
BGY	Pre	63	6.68	1.67	56	3.12×10^{-5}
	Post	62	7.56	1.20		
Red	Pre	22	7.23	1.48	21	0.125
	Post	22	8.00	1.21		
Total	Pre	85	6.82	1.63	77	1.39×10^{-5}
	Post	84	7.68	1.21		

Figure 2: Graph of Question 3 Results



The data show that students' algorithm scores improved from pretest to posttest. Using a 2-tailed paired t-test, the results are shown to have statistical significance ($p < .05$) for each group examined except the Red and Green classes; thus, it is highly probable the improvement is not a statistical anomaly. It is possible that one algorithm was easier to write or describe than another and this caused the increase in performance, but both problems were chosen to be common, everyday tasks that all students would have experience with (i.e., brushing their teeth and washing their hands). Also, since both tasks involve using a sink and then doing some action, this bias is likely mitigated.

From this improvement in the quality of their algorithms, it can be concluded that the lessons did improve the students' ability to design processes.

4.1.3 QUESTIONS 4 AND 5 – COMPUTERS AND COMPUTER SCIENTISTS

Although these questions do not deal directly with outcomes related to students' understanding of computer science, these ideas were addressed in lessons. It is interesting to explore the students' development of these ideas as well. For each question, the students' responses were coded into categories of related responses. The percentage of occurrence for each response category in each question is given in the following results. The results of Question 4, "How did [the word processor] learn to check your spelling?" can be seen in Table 6 and Figure 3. The results to Question 5, "In your own words, a computer scientist is a person who…" can be seen in Table 7 and Figure 4.

Table 6. Percentage of Category Occurrence for Each Class in Answering Question 4

Categories	Blue		Green		Yellow		BGY		Red		All	
	Pre	Post	Pre	Post	Pre	Post	Pre	Post	Pre	Post	Pre	Post
No response	5%	-	4%	5%	-	-	3%	2%	-	-	2%	1%
Don't know	55%	15%	70%	-	65%	9%	64%	8%	32%	5%	56%	7%
Indecipherable	5%	15%	-	-	-	-	1%	5%	-	-	1%	3%
Response does not answer question	9%	5%	-	-	4%	-	4%	2%	5%	-	4%	1%
Install/Download	-	5%	-	14%	-	-	-	6%	-	14%	-	8%
Internal Dictionary/ Microchip/Technology	9%	20%	4%	27%	9%	4%	7%	17%	23%	18%	11%	17%
External Person/ Company made it so	5%	25%	-	9%	4%	17%	3%	17%	18%	5%	7%	14%
Programmed/ Algorithm	-	10%	4%	14%	9%	43%	4%	23%	9%	45%	6%	29%
Describes Spell Check	14%	-	17%	27%	4%	17%	12%	15%	14%	9%	12%	14%
Other	-	5%	-	5%	-	9%	-	6%	-	5%	-	6%

Figure 3: Cumulative Percentage Graph of Q4 Results

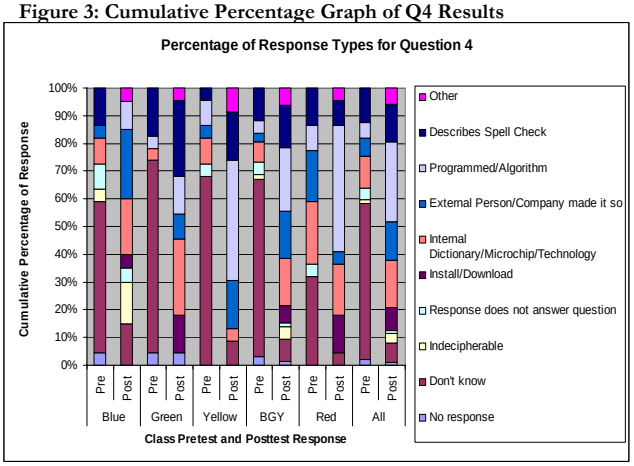


Figure 4: Cumulative Percentage Graph of Q5 Results

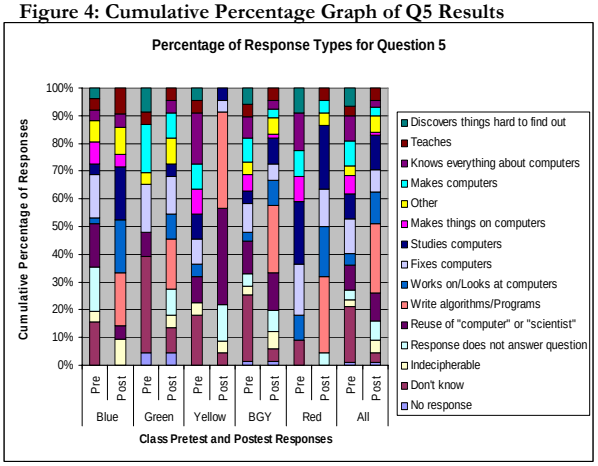


Table 7. Percentage of Category Occurrence for Each Class in Answering Question 5

Categories	Blue		Green		Yellow		BGY		Red		All	
	Pre	Post	Pre	Post	Pre	Post	Pre	Post	Pre	Post	Pre	Post
No response	-	-	4%	5%	-	-	1%	2%	-	-	1%	1%
Don't know	18%	-	35%	9%	18%	4%	24%	5%	9%	-	20%	3%
Indecipherable	5%	10%	-	5%	5%	4%	3%	6%	-	-	2%	5%
Response does not answer question	18%	-	-	9%	-	13%	4%	8%	-	5%	3%	7%
Reuse of words "computer" or "scientist"	18%	5%	9%	-	9%	35%	12%	14%	-	-	9%	10%
Write algorithms/Programs	-	19%	-	18%	-	35%	-	24%	-	27%	-	25%
Works on/Looks at computers	2%	19%	-	9%	5%	-	3%	9%	9%	18%	4%	11%
Fixes computers	18%	-	17%	14%	9%	4%	10%	6%	18%	14%	12%	8%
Studies computers	5%	19%	-	5%	9%	4%	4%	9%	23%	23%	9%	13%
Makes things on computers	9%	5%	-	-	9%	-	6%	2%	9%	-	7%	1%
Other	9%	10%	4%	9%	-	-	4%	6%	-	5%	3%	6%
Makes computers	-	-	17%	9%	9%	-	9%	3%	9%	5%	9%	3%
Knows everything about computers	5%	5%	-	5%	18%	-	7%	3%	14%	-	9%	2%
Teaches	5%	10%	4%	5%	5%	-	4%	5%	-	5%	3%	5%
Discovers things hard to find out	5%	-	9%	-	5%	-	6%	-	9%	-	7%	-

In examining the results of Question 4, it is apparent that students made a shift from not having any idea on the question to having answers based on ideas shared by students during lessons about how the word processor learned to spell check. It is also interesting to note that most students gave the "correct" answer on the posttest by stating that spell checking is an algorithm that has been programmed.

Focusing on the results of Question 5, the students clearly adopted the idea that computer scientists spend time creating algorithms and programming, as that response was not seen on pretests but was the most common response on the posttests. Also of note is a strong belief that computer scientists are students or teachers, which may be a product of the author's presence as a computer science student.

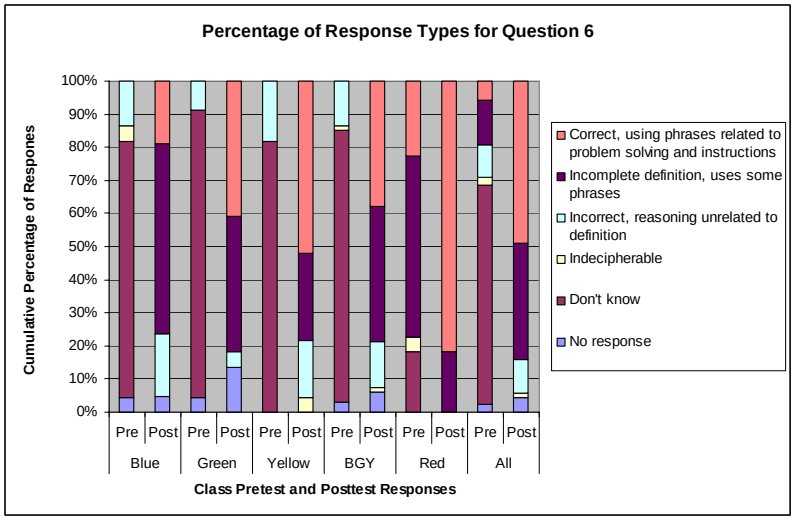
4.1.4 QUESTION 6 – DESCRIBING ALGORITHMS

The final question of the pretest and posttest was again qualitative. Thus, student responses were coded into categories that generally fit their answers to the question, "An algorithm is…" It was expected that students would not know what an algorithm is, or any components of an algorithm, on the pretest, yet would be able to name elements of an algorithm, if not the full definition, on the posttest. The pretest and posttest results can be seen in Table 8 and Figure 5. It is again important to note the bias of the Red class, as they had prior exposure to algorithm lessons. The reader is advised to consider the BGY (Blue, Green, and Yellow) group, rather than the All group in representing the change in the abilities of the 2nd grade students.

Table 8. Percentage of Category Occurrences for Each Class in Answering Question 6

Categories	Blue		Green		Yellow		BGY		Red		All	
	Pre	Post	Pre	Post	Pre	Post	Pre	Post	Pre	Post	Pre	Post
No response	5%	5%	4%	14%	-	-	3%	6%	-	-	2%	5%
Don't know	77%	-	87%	-	82%	-	82%	-	18%	-	66%	-
Indecipherable	5%	-	-	-	-	4%	1%	2%	5%	-	2%	1%
Incorrect, reasoning unrelated to definition	14%	19%	9%	5%	18%	17%	13%	14%	-	-	10%	10%
Incomplete definition, uses some phrases	-	57%	-	41%	-	26%	-	41%	55%	18%	13%	35%
Correct, using phrases related to problem solving and instructions	-	19%	-	41%	-	52%	-	38%	23%	82%	6%	49%

Figure 5: Cumulative Percentage Graph of Q6 Results



In reviewing the BGY group, the pretest clearly shows an unsurprisingly strong response of "I don't know" from the students. It is thus remarkable that, for the posttest, no students give an "I don't know" response. In fact, 79% of students in the BGY group give adequate answers using all or some elements from the definition of algorithm including phrases such as "list of instructions," and "solves problem." This

provides empirical evidence that the concept of an algorithm was learned by students through the course of these lessons.

4.2 ANSWERS TO POST-LESSON QUESTIONS

For each lesson taught, a question or questions were given to the students after the lesson, which were to be answered in their journals. Most of these questions directly related to assessing students' knowledge and understanding of algorithms, and only these questions will be addressed in this section.

4.2.1 ALGORITHMS IN GENERAL

The question posed to students after the first lesson was, "What did you like about today's lesson?" This question does not give an indication to student understanding of algorithms and thus will not be explored in-depth. Anecdotally, most students described the opening lesson as fun. They also mentioned "enjoying the funny parts," e.g., when the class talked about putting on underwear outside of one's pants. Many also indicated they found algorithms easy and interesting. No negative comments were made about the lesson.

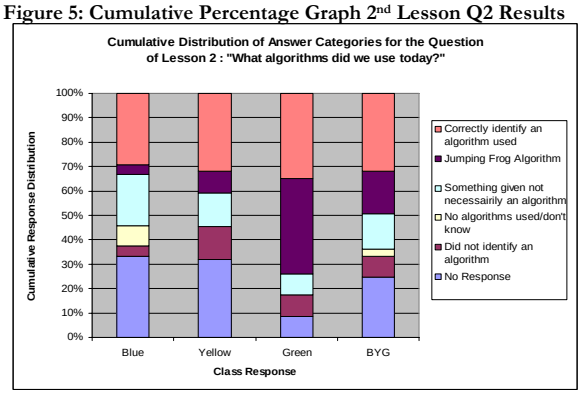
4.2.2 APPLICATIONS OF ALGORITHMS IN ORIGAMI

Following this lesson, the students were given two questions to answer: "What algorithms did we use today?" and "What makes the frog jump?" As the latter does not reflect students' conception of algorithms, it will not be addressed here. The results of the former were qualitative and categorized by content of the responses. The results of these responses can be seen in Table 9 and Figure 10. It is important to note that the Red class did not give responses to these questions, as the lesson ran long and there was not time for have them to respond.

Table 10. Percentage of Category Occurrence 2nd Lesson Q2

	Blue	Yellow	Green	BYG
No Response	33%	32%	9%	25%
Did not identify an algorithm	4%	14%	9%	9%
No algorithms used/ don't know	8%	-	-	3%
Something given not necessarily an algorithm	21%	14%	9%	14%
Jumping Frog Algorithm	4%	9%	39%	17%
Correctly identify an algorithm used	29%	32%	35%	32%

Figure 5: Cumulative Percentage Graph 2nd Lesson Q2 Results



Approximately one-third of the students responding gave a correct algorithm used during the lesson. Typical phrases included "an algorithm for making a jumping frog" or an "origami jumping frog algorithm." One interesting phrase that occurred frequently in the Green class was "jumping frog algorithm." This phrase is questionable as to whether or not it correctly identifies an algorithm, because it lacks an articulation of the algorithm. However, the author believes it was the intent of the student to describe the algorithm used in constructing the jumping frog and, thu,s concludes that a majority of the students who responded could accurately identify an algorithm. This result further implies that the students have grasped the concept of an algorithm.

4.2.3 APPLICATIONS OF ALGORITHMS IN GAMES

The questions following this lesson were "What algorithms did we use today?" and "How could we find the computer's number in less guesses?" The latter question was sparsely answered by the students and thus no analysis can be done of the responses.

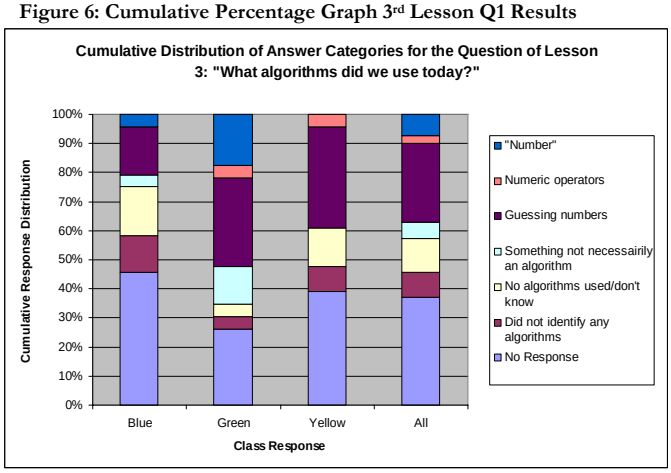
Anecdotally, of the few responses provided, some suggested counting by fives or tens to make guesses. Others suggested making "smart" guesses, which are guesses "that you really, really think are the number." A small number suggested guesstimating.

However, the former question continues the analysis of students' conceptions of algorithms and will be explored in depth. The categorized response results can be seen in Table 11 and Figure 6. As the students in the Red class had previously been exposed to this lesson, they were not asked to answer these questions in their journal. Their awareness of the optimal guessing algorithm and the fact it was an algorithm were apparent in conducting the lesson a second time.

Table 11. Percent of Cat. Occurrence 3rd Lesson Q1

	Blue	Yellow	Green	BYG
No Response	46%	26%	39%	37%
Did not identify any algorithms	13%	4%	9%	9%
No algorithms used/don't know	17%	4%	13%	11%
Something not necessarily an algorithm	4%	13%	-	6%
Guessing numbers	17%	30%	35%	27%
Numeric operators	-	4%	4%	3%
"Number"	4%	17%	0%	7%

Figure 6: Cumulative Percentage Graph 3rd Lesson Q1 Results



In interpreting the results, it is of note that 27% of students gave guessing numbers as an algorithm, which is a correct answer in this case due to the decision making involved. However, another 27% of students gave other less-correct replies, including "no algorithms used." In this case, students did not overwhelmingly identify an algorithm used during the course of the lesson. A possible reason for this is that the algorithms used were particularly abstract in making decisions about numbers, as the less-correct replies tended to reference concrete entities in the lesson (e.g., computers, less than/greater than signs, etc). This is still a surprising result when considering two algorithms were fairly well outlined during the course of the lesson (i.e., the role of the person with the chosen number, the role of the person guessing the number). In comparing the results of this question and the same question in the previous lesson, it is clear that, with the exception of the Blue class, approximately the same percentage of students gave a more-correct answer to the question, along with a higher rate of response. We can infer from this that a good number of students still display the ability to identify an algorithm, and the results of the previous question are not an anomaly.

4.2.4 ALGORITHMS AND COMPUTERS

After the completion of this lesson, the students were asked, "What does a computer scientist do?" and "How did [the word processor] learn to check spelling?" As the two post-lesson questions are addressed fully in the pretest and posttest, the student answers given in the students' journals will

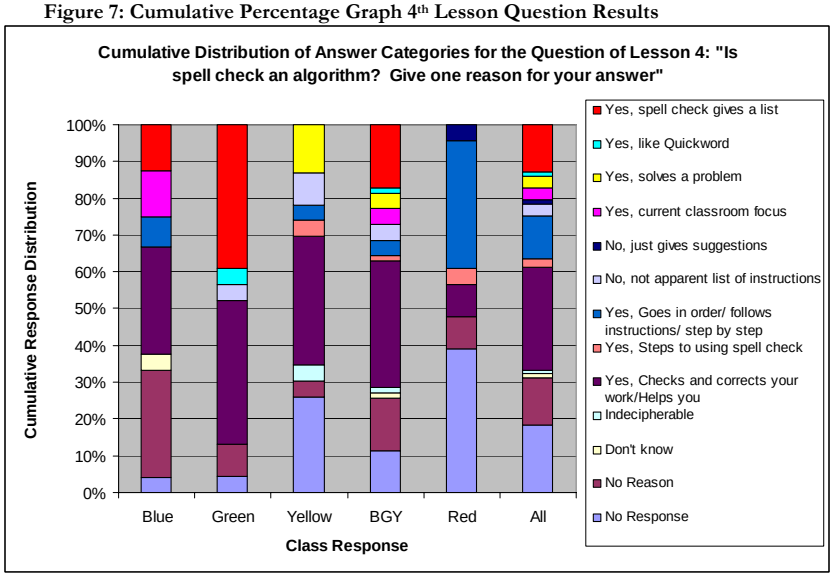
not be explored in-depth. Anecdotally, the responses do mirror that of the pretest and posttest responses, as one would expect.

During this lesson, the students were asked to answer the question, "Is ABC spell check an algorithm?", and to give one reason for that answer in their journal. The student responses to this question are categorized in Table 12 and graphed in Figure 7.

Table 12. Response Categories to Question in 4th Lesson

	Blue	Green	Yellow	BGY	Red	All
Yes	21	21	15	57	13	70
No	2	1	2	5	1	6
No Response	4%	4%	26%	11%	39%	18%
No Reason	29%	9%	4%	14%	9%	13%
Don't know	4%	-	-	1%	-	1%
Indecipherable	-	-	4%	1%	-	1%
Yes, Checks and corrects your work/Helps you	29%	39%	35%	34%	9%	28%
Yes, Steps to using spell check	-	-	4%	1%	4%	2%
Yes, Goes in order/ follows instructions/ step by step	8%	-	4%	4%	35%	12%
No, not apparent list of instructions	-	4%	9%	4%	-	3%
No, just gives suggestions	-	-	-	-	4%	1%
Yes, current classroom focus	13%	-	-	4%	-	3%
Yes, solves a problem	-	-	13%	4%	-	3%
Yes, like Quickword	-	4%	-	1%	-	1%
Yes, spell check gives a list	13%	39%	-	17%	-	13%

Figure 7: Cumulative Percentage Graph 4th Lesson Question Results



In reviewing the results of this question, it is first important to note the overwhelming 34% response from the BGY group that spell checking is an algorithm because it "helps you." This can imply that many of the students have the perception that the purpose of an algorithm is to help someone solve a problem. This is not that far from the definition, if somewhat human-centric. It also suggests a reason why students title the algorithms they create with "How to…," which will be discussed further in section 4.3. Also, from the BGY group, it is interesting to note that 29% of responses include ideas from the

definition of an algorithm (e.g., lists, solves a problem, instructions, going in order). In considering all the classes, 59% of all responses include some idea from the definition of an algorithm or the concept that an algorithm helps someone. It can be concluded from this that the students developed a grasp of the concept of an algorithm and associate it with helping a person to solve a problem.

4.3 STUDENT ALGORITHMS

The final piece of data collection for this assessment project is analyzing the algorithms produced by students in the final lesson. Students were encouraged to choose any problem they knew how to solve and give at least five instructions to solve that problem. Sample algorithms can be seen in Appendix D. Algorithms were to be written into the students' observation journals as rough drafts and later typed into final drafts. The analysis done here comes from the rough drafts at the end of the journals. Note: The students in the Red class produced algorithms earlier in the year and were asked to voluntarily make new algorithms, which they were to write in the backs of their journals.

As students were allowed to choose any problem they wanted and knew how to solve, it is interesting to consider the areas they chose for their problems. Twelve major categories were identified for the subjects of their algorithms. The breakdown of each can be seen in Table 13.

Table 13. Categories for Types of Problems Chosen by Students, F & M

Classification	n	Pct	Fem.	Pct F	Pct of F	Males	Pct M	Pct of M
Pet	12	13%	6	50%	12%	6	50%	13%
Physical/Sports	34	35%	13	38%	26%	21	62%	46%
Everyday	10	10%	7	70%	14%	3	30%	7%
Video Game	3	3%	2	67%	4%	1	33%	2%
School	3	3%	1	33%	2%	2	67%	4%
Food/Recipe	12	13%	9	75%	18%	3	25%	7%
Art	10	10%	6	60%	12%	4	40%	9%
Behavioral	3	3%	2	67%	4%	1	33%	2%
Plant	2	2%	2	100%	4%	0	-	-
Computer	1	1%	0	-	-	1	100%	2%
Building	3	3%	0	-	-	3	100%	7%
Event Planning	3	3%	2	67%	4%	1	33%	2%

For the most part, it appears students are interested in designing algorithms relating to physical activities/sports, pets and pet care, and foods/recipes. In particular, it appears males dominantly create algorithms about physical activities. Females also write many algorithms about physical activities, but tend to more normally distribute the categories of algorithms they produce.

An interesting note about these algorithms comes from their titles. Almost every algorithm's title starts with the phrase "How to…" This particular phrase was not explicitly used in the lessons, nor can the author directly attribute it to any apparent facet of the lessons. It is possible students conceptualize algorithms as lists of instructions to achieve tasks rather than more generally solve problems. In this way, they would view algorithms as being about people and telling them how to do something. This would be consistent with a theme of the lessons, starting with the algorithms in the introductory lesson (i.e., getting dressed andmaking breakfast), the algorithm for making origami frogs, describing the algorithm a person follows when playing the number game, and even further reinforced by the processes asked about on the pretest and posttest (i.e., brushing your teeth and washing your hands) which all instruct a human how to complete a task. To some degree, this result is verified by the students' response that spell checking is an algorithm because it "helps you," as is discussed in section 4.2.4.

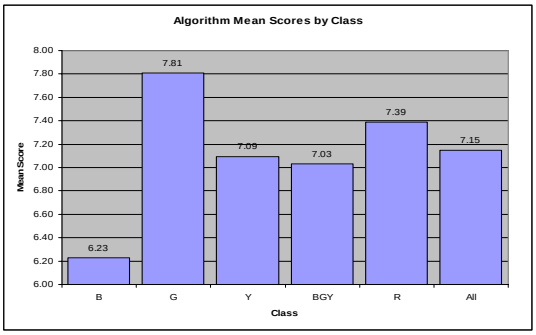
Moving from the objective inferences about the algorithms to more subjective inferences, the algorithm scoring rubric, as described in section 4.1.2, was again applied to determine the quality of the responses. Example algorithms and their scoring can be seen in Appendix D. The results of

applying the rubric and statistics on the number of steps in each algorithm can be seen in Table 14 and Figure 8.

Table 14. Algorithm Analysis Results

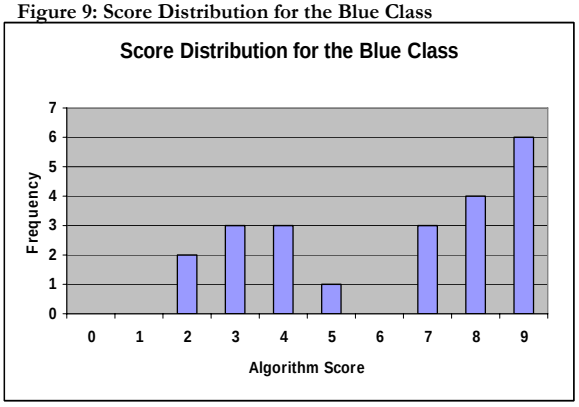
	n	Mean Score	Std. Dev.	Mean Steps	Std Dev.
B	22	6.23	2.62	6.32	2.25
G	21	7.81	0.98	7.00	2.59
Y	22	7.09	1.77	5.82	1.22
BYG	65	7.03	2.00	6.37	2.12
R	31	7.39	1.73	6.55	2.34
All	96	7.15	1.91	6.43	2.18

Figure 8: Algorithm Mean Score by Class



These results are mostly consistent with those of the earlier pretest and posttest analysis of student algorithms. Inconsistencies can be seen with the unexpectedly low average performance of the Blue class. The mean score of the Blue class must be taken into consideration with the very high standard deviation, implying some degree of parity between students in the class, which can be seen in Figure 9. One possible explanation for the difference between the two results is that fewer students in the Blue class took the posttest and could have been in the mediocre grouping shown in Figure 9. As the pretest and posttest result for the Blue class is statistically significant for the 16 students who took both the pretest and the posttest, this is a likely reason for the discrepancy.

Figure 9: Score Distribution for the Blue Class



Looking at the mean number of steps used in the students' algorithms, it should be noted that the students on average chose to use more than the required five steps in designing their solutions. This result shows that the students put effort into the algorithm assignment. This effort claim is also reinforced by the relatively high mean scores. It can be concluded from these results that students produced algorithms of decent quality with an amount of complexity greater than what was required.

5 CONCLUSION

Initial conclusions include that the students in the study have a better grasp of how computer programs are constructed, as well as what a computer scientist does. Another conclusion is that, when designing algorithms, students take an interest in problems dealing with physical activities and sports. It is possible that future lessons building from this study could better incorporate sports to increase student interest.

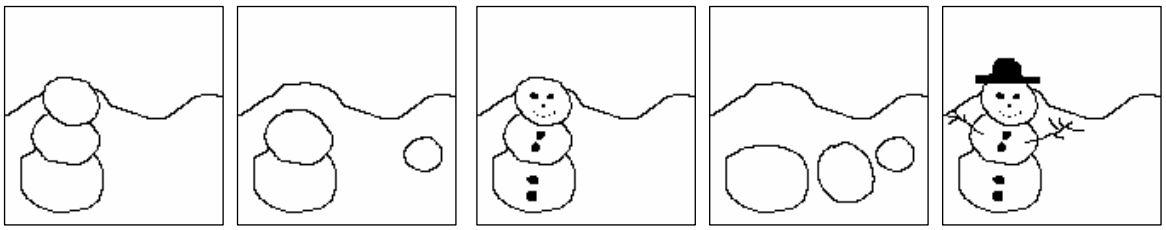
The purpose of this study was to explore younger students' ability to understand processes and design algorithms to solve general problems. As can be seen from the pretest and posttest results, students did learn the concept of an algorithm and improved their ability to design them. When

describing what an algorithm is, the majority of students either gave the definition of an algorithm or used key terms such as "list of instructions" or "solves problem," which shows the students grasped the concept of an algorithm and, thus, processes. In writing algorithms to solve everyday problems on the tests, the statistically significant result is that the students produced algorithms of higher quality after the lessons, showing that they can better design algorithms for solving problems. This is further verified by the quality and complexity of the algorithms produced by the students in the assignment given in the final lesson. Therefore, it can be concluded that 2nd grade students can understand the concept of processes, as well as produce algorithms to solve general problems.

6 ACKNOWLEDGEMENTS

Many thanks are due to the teachers and students that made this work possible. The author wholeheartedly acknowledges and appreciates the contributions of Anne C. who helped with entry and analysis of all the data contained herein. Thanks are also due to Dr. Christine Cunningham who helped in design of the pretest and posttest, as well as Dr. Kris Powers, Dr. Judah Schwartz, and Dr. Linda Beardsley, who assisted with lesson design. The author also wishes to thank Dr. Diane Souvaine for the opportunity to be a part of the GK-12 program and, especially, Brian Gravel for his continued support of this work and GK-12 in general. This material is based upon work supported by the National Science Foundation under Grant No. DGE-0230840.

1. Please put the following pictures in the order they happened from 1st to 5th.



_____ _____ _____ _____ _____

2. Please put the instructions for getting ready for school in order from 1st to 5th.

_____ Put on your backpack.

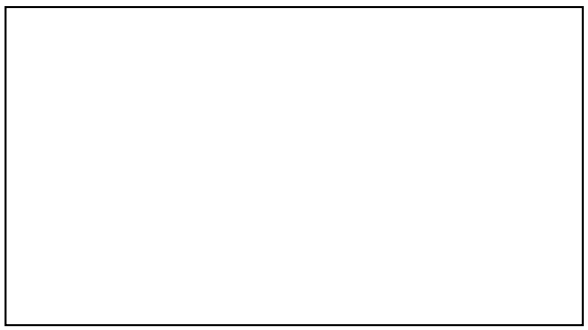
_____ Get dressed.

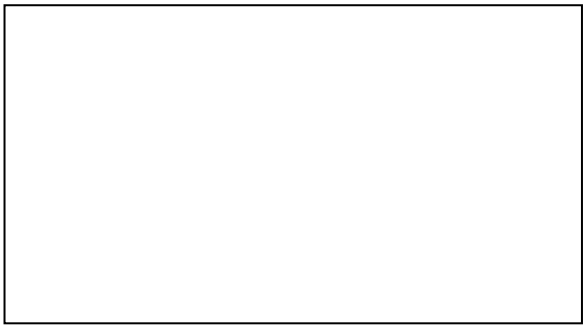
_____ Wake up and get out of bed.

_____ Put on your coat.

_____ Leave for school.

3. In each box, draw the steps you take in washing your hands. Describe each step on the lines next to the box.





4. When you use Student Writing Center, you ABC check your work to make sure that your spelling is correct. In your own words, how did Student Writing Center learn to check your spelling?

5. In your own words, a computer scientist is a person who…

6. In your own words, an algorithm is…

1. Please put the following pictures in the order they happened from 1st to 4th.

_____ _____ _____ _____

2. Please put the following instructions for blowing up a balloon in order from 1st to 5th.

_____ Tie a knot in the balloon.

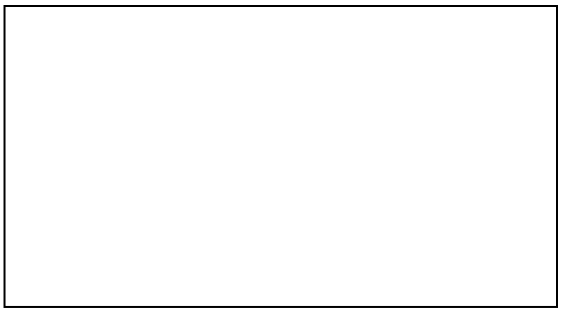
_____ Put your mouth up to the balloon.

_____ Watch the balloon get big.

_____ Pick out a balloon to blow up.

_____ Blow air into the balloon.

3. In each box, draw the steps you take in brushing your teeth. Describe each step on the lines next to the box.



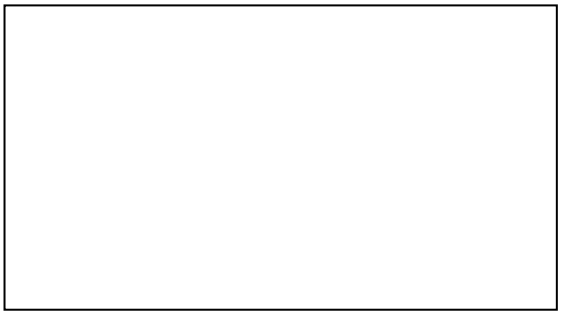
_____

_____

_____

_____

_____

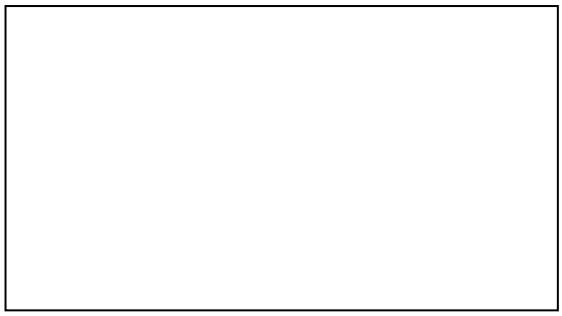


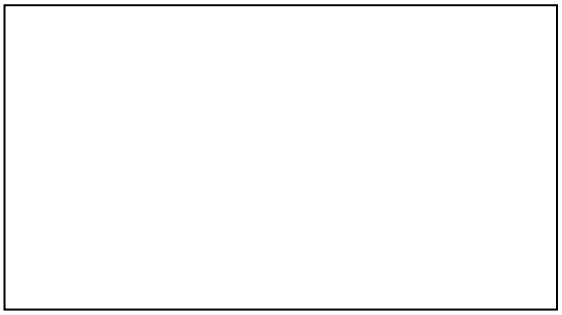
_____

_____

_____

_____



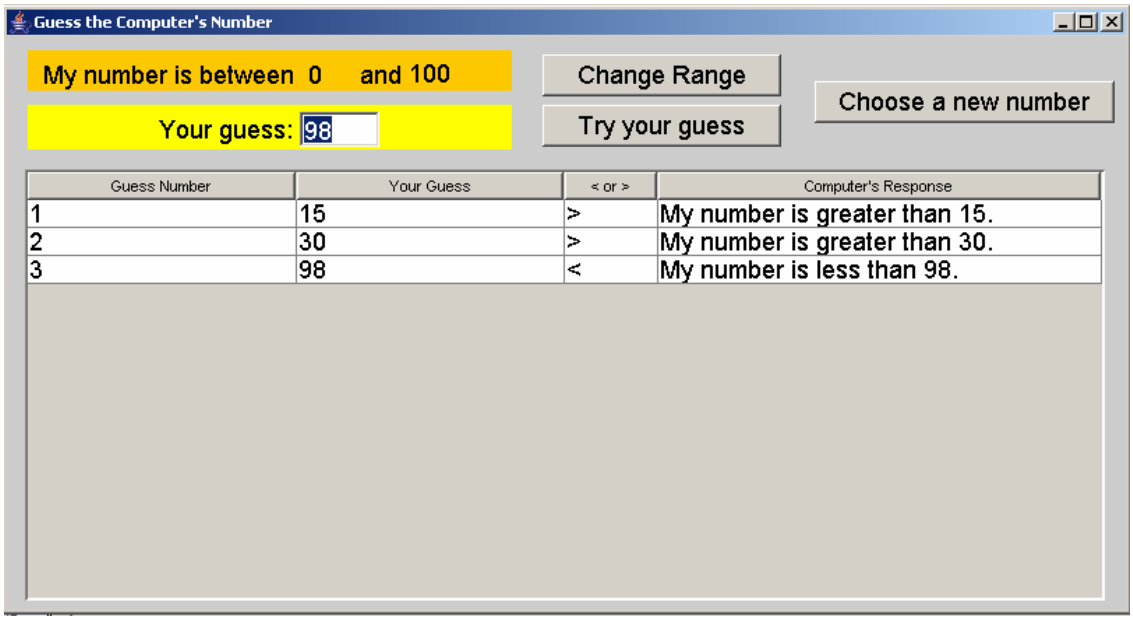


4. When you use Student Writing Center, you ABC check your work to make sure that your spelling is correct. In your own words, how did Student Writing Center learn to check your spelling?

5. In your own words, a computer scientist is a person who…

6. In your own words, an algorithm is…

APPENDIX C – BINARY SEARCH GAME SCREENSHOT



APPENDIX D – SAMPLE ALGORITHMS AND SCORING

HOW TO GIVE YOUR CAT A BATH

1. You get your cat in the bathroom.
2. Get the cat shampoo and get hot water running
3. Shut the water off and put the cat in
4. Get your cat wet then put the shampoo on the cat. Remember open the bottle first
5. Rinse the cat off with the water then put a 2nd coat on
6. Rinse off the 2nd coat on the cat.
7. Take the cat out of the bath
8. Dry the cat off, the cat may be shivering.
9. Leave the cat alone but make shure the cat is dry.

Completeness:	3
Coherence:	3
Order:	3
TOTAL	9

HOW TO PLAY HOCKEY

1. Shoot in the open space.
2. Pass to your teamate.
3. If your goalie, stay out of the crease and look at the puck.
4. You have to get up as quick as you can.
5. Pretend you doing a shot fake it, pass, and shoot.

Completeness:	1
Coherence:	1
Order:	0
TOTAL	2

HOW TO SWIM

1. First put on a bathing suit.
2. Then go in the pool.
3. Then you paddle your hands.
4. then kick you feet.
5. then you swim in a big pool.
6. Then you know how to swim.

Completeness:	3
Coherence:	2
Order:	2
TOTAL	7