Intrinsic Task Value of Algorithmic Thinking in Middle-School Students

and its Effect on Attitudes Toward STEM Work

Nikole A. Kerns

East Tennessee State University

Author Note

This research was completed as part of the Ronald E. McNair Internship in April 2019

Abstract

Computational thinking (CT) in middle school has been of interest since the phrase was introduced by Papert in the 1980s. Research shows that increased ability in CT correlates with significant educational outcomes, including success in a range of specific subjects and the likelihood that a student will finish college. Differences in subjective value and expectancy for success in CT may help explain gender and racial differences in choosing math-intensive coursework, college majors, and careers. CT in middle school is typically introduced using game-based programming tasks, confounding engagement with CT with other factors, such as storytelling, art, and the cultural power of digital game play. Evidence shows that one aspect of CT, algorithmic thinking, is itself intrinsically valued by people as early as high school. This study hypothesized that scaffolded teaching of algorithmic thinking would engage middle-school students, have intrinsic and utility task value, and would change middle-school students’ attitudes toward STEM courses and careers. To test this, I proposed a benchtop study exploring (a) the ability of middle-school students to think algorithmically, (b) the intrinsic task value of such work, and (c) its effect on their expectancy for success in mathematics and in academic tasks generally. In a 1 hour during school session, I asked students to approach a typical middle-school math problem, finding the common divisor between two numbers, first using their own methods, then following the simple algorithm set out by Euclid. To allow students to come to their own methods of following the algorithm, I avoided directly teaching the steps. My role was to scaffold competence in algorithmic thinking and to present challenges that illustrate the unexpected learning possible through the algorithms and the increased efficiency in attacking problems that are meaningful in students' academic work. The intrinsic value of the tasks was assessed through a brief questionnaire before and after the session. Expectancy for success in mathematics-related work and in academic tasks was pretested with a brief questionnaire before and after the session as well.

Intrinsic Task Value of Algorithmic Thinking in Middle-School Students

and its Effect on Attitudes Toward STEM Work

In school, students react in different ways to computer science. Some are overwhelmed because it looks so different from anything they have seen, but some are intrigued by how they can manipulate data. Experience and how students are introduced to this subject play a role on how they view it. If we can get students interested this may have wonderful results.

Research shows that intrinsic value has positive outcomes for students (Ryan & Deci, 2000). Based on the expectancy-value theory (Eccles, Barber, Updegraff, & O’Brien, 1998), it is argued that if there could be internal motivation for students to engage in academic work that requires algorithmic thinking, then students will feel that they are increasing their abilities to think logically and to solve problems they did not know they could work on. This engagement might increase their motivation to engage in for academic tasks generally and math-related work specifically. The aim of this proposal is to extend prior emphasis of previous research on the influence algorithmic thinking has in developing higher level skills to explore whether this higher level of thinking is interesting and enjoyable to middle-school students and whether that can influence what careers and courses students choose to take.

Literature Review

Pushing computational thinking (CT) down into middle school has been on the educational agenda since it was introduced by Papert in the 1980s (Papert, 1980). Papert coined the phrase “computational thinking” to mean the art of deliberately thinking like a computer, for example, how a computer proceeds in a step-by-step, literal, and mechanical procedure. This review will address: (a) the importance of teaching computational thinking in developing students higher-level skills; (b) how emphasizing algorithmic thinking specifically helps us understand if students value thinking abstractly on its own; (c) the influence intrinsic value has on choices made by students when it comes to courses and careers; and (d) a teaching style that allows the student to build their own abilities for a difficult task and to watch themselves grow in competency.

**Importance of** **Teaching Computational Thinking**

Papert (Papert, 1980) claimed our culture is poor in systematic procedures and was most interested in the educational advantage of teaching CT. Analysis of this way of thinking and how it is different from other kinds of thinking “can result in a new level of intellectual sophistication” in those doing so (p. 27). Most important to Papert was that, through these experiences, children would be thinking articulately about thinking. He claims the educational environment for children does not allow thinking about thinking to be brought out into the open and so children cannot learn to talk about their thoughts and test their ideas.

Wing (2006) in an opinion piece for the Association for Computing Machinery points out that CT has tended to transform all the many disciplines it has been introduced into and compares computers to the printing press: “Just as the printing press facilitated the spread of the three R’s... computing and computers facilitate the spread of computational thinking” (p. 33). Wing writes passionately about teaching CT throughout schools as something everyone “would be eager to learn” but relates it to computer science: “Computational thinking builds on the power and limits of computing processes... [draws on] the concepts fundamental to computer science” (p. 33). To Wing, CT meant thinking like a computer scientist, not just a computer programmer. She highlights among other things, thinking at multiple levels of abstraction, working within the constraints of devices that have a limited way of doing things, reformulating new problems to make use of old solutions, thinking differently about efficiency, and focusing on thinking ahead to consequences and back to what is limiting solving a problem.

Wing goes on to describe how useful it would be to transfer thinking in these concepts to many situations and disciplines: solving a problem using constrained resources, asking when a solution is good enough, reforming problems as problems that have already been worked through, thinking recursively and using many parallel processes, thinking about prevention, protection, and recovery, looking for and correcting errors, and looking at data as a way to speed up work.

Wing goes on to target younger students and the population generally: “We especially need to reach the pre-college audience, including teachers, parents, and students... We should expose pre-college students to computational methods and models... we should look to inspire the public’s interest in the intellectual adventure of the field. We’ll thus spread the joy, awe, and power of computer science, aiming to make computational thinking commonplace.” (Wing, 2006, p. 35)

Buitrago Flórez, et al. (2017) systematically reviewed the research literature on how CT has been brought into schools and what the results are. This study defines CT as “a way of reasoning that compiles several high-level skills and practices that are at the heart of computing but applicable to many areas far beyond computer science.” Their review found that CT enhances thinking ability by allowing students to handle abstract problems much like a computer programmer has to do when they are developing an algorithm, and that CT is very useful and rewarding to students in numerous fields such as biology, chemistry physics, medicine, engineering, arts, music, and social sciences. However, that review also found problems in teaching and learning programming. The first one is motivation. The students believe success is beyond their control regardless of understanding or time spent on the program. The second issue is congruence between skills learned in introductory courses and ability to write specific programs in an efficient way. Finally, there is a methodological problem in teaching programming. The professors assume students can learn on their own just from examples shown in class.

Papert (1980) conducted a pioneering experiment on how K-8 students can be engaged in CT. In this experiment students are asked to form all the possible combinations of beads of assorted colors. He found that these children were unable to do this systematically and accurately until they are in the fifth or sixth grades. Papert went on to invent “turtle graphics” and the *Logo* programming language as ways for middle-school children to “think like a computer”.

Werner, et al. (Werner, Denner, Bliesner, & Rex, 2009) refer back to Papert’s experiment in an article on teaching programming to middle schoolers using the educational software Alice to make games. They mention using middle schoolers because this was a good age group to understand how the Alice system works. An earlier article by this group on girls creating games through programming (Denner, Werner, Bean, & Campe, 2005) describes middle-school as a critical time to intervene because students (specifically girls) are exploring identities, interests, and talents. This affects pathways to participate in information technology activities. These early decisions also affect enrollment in computer classes in high school. In a more recent study (Werner, Denner, Campe, & Kawamoto, 2012), middle-schoolers are used to target a younger generation.

Shifting the emphasis to algorithmic thinking

An algorithm is a sequence of operations for solving a specific type of problem (Knuth, 1968). In order to develop these sequences of operations, students must be able to think abstractly. Some examples of how algorithms appear in middle school include long division, finding the least common multiple, adding fractions with unlike denominators, and breaking numbers into place-value systems (Tennessee Department of Education, 2017).

Computational thinking in middle school is typically introduced and assessed using game-based computer programming tasks. These programming tasks are promising as a reliable strategy to introduce computer science concepts to a wide array of students (Werner et al., 2012). Games like these develop problem solving skills by breaking down logic, forcing students to think differently. However, it is an open question whether middle schoolers would feel similarly about computational thinking outside of games.

Relying on games as a means of getting young people to engage in computational thinking confounds the reason for their being interested in the work. Is it because they are able to think in new ways, or are they more interested in creating characters, making a storyline, controlling the action? Buitrago, et al. (Buitrago Flórez et al., 2017) focused on the benefit of CT’s emphasis on abstract thinking, and specifically on developing and using algorithms much like people actually writing computer programs. They defined algorithmic thinking as a step-by-step way to solving a problem, while they took CT as a broader term that includes algorithmic thinking, logic, abstraction, generalization, decomposition, and debugging. Wing highlighted the algorithmic thinking aspect of CT: "Computational thinking will have become ingrained in everyone's lives when words like algorithm... are part of everyone's vocabulary" (Wing, 2006, p. 34). I think it is very important and would be interesting to children below high school, thinking in and about algorithms.

**Intrinsic value of thinking algorithmically**

Evidence shows that this aspect of CT is in itself satisfying and intrinsically valued by students as early as high school. Torbert, et al., (Torbert, Vishkin, Tzur, & Ellison, 2010) demonstrates a unique way to teach parallel programming. One interesting side-effect they presented was what they called the “wow factor” when students began to understand how to think about algorithms. This effect was from the student’s realization that the approach to building algorithms in parallel was different from anything they had experienced and much more powerful and gave them a different way to think about programming. The authors concluded that “the ‘wow’ factor resulting from this discussion cannot be overstated” (p. 3).

The success in bringing middle school students to take on programming could be because they are interested in just the game aspects of middle-school computer tasks and not the abstract algorithmic thinking itself. This study wants to understand better whether middle-school kids find thinking abstractly and in steps rewarding by themselves by taking the game out of the task. If middle-schoolers can be guided to this form of thinking and if they find it rewarding, then it could allow an opportunity for students to change their attitudes toward STEM courses and careers.

Eccles and colleagues (Eccles et al., 1998) developed a complex theory to explain why people make their choices when asked to take on academic challenges like course work, college majors, or even careers. This theory claims that expectancy for success and subjective task values determine these choices. Expectancy for success has to with the students’ self-concept when it comes to academia. Subjective task values are the motivation students have to engage in an academic setting. There are four parts to subjective task value. The first one is interest-enjoyment value (enjoyment or interest in a task intrinsically), attainment value (self-concept or identity), utility value (usefulness or relevance), and relative cost (high effort demands, too much time used, too many resources used).

The focus in this study is on intrinsic task value because it is correlated to motivation, interest, choosing behaviors, and task persistence (Jacquelynne S. Eccles, Barber, Updegraff, & O’Brien, 1998). A task’s intrinsic value is one form of motivation for students in academics and is linked to attitudes toward specific academic subjects and to effort in the classroom. It leads students to be more interested in a subject and correlates to higher achievement in that subject (Chiu & Xihua, 2008). Interest and engagement in academic tasks can be from personal enjoyment and not from external rewards. An example of this around algorithmic thinking is the importance Torbert, et al., (2010) gave to the “wow factor” that resulted when they taught parallel processing. Higher achievement and higher levels of intrinsic value can ultimately lead students to choosing classes and careers (Harackiewicz, Barron, Tauer, & Elliot, 2002). There is evidence that the contribution of intrinsic value is more important than intelligence to school achievement (Spinath, Spinath, Harlaar, & Plomin, 2006).

**Teaching to allow students to experience their growing competence**

If learning algorithmic thinking is going to have intrinsic value to the students, I believe they will have to experience for themselves how well it helps them work through problems and how picking among different algorithms makes solutions more efficient. With practice, students will more quickly find a way of thinking that works for them. This mechanism of allowing the student to think freely while guiding them back when they veer off track is a form of scaffolded teaching. One rationale of this teaching is that when a student finally figures out the problem there will be intrinsic task value gained from solving it.

Bruner and Wood (Wood et al., 1976) introduced the idea of “scaffolding” to describe how someone acting as a tutor guides a child to build their own abilities in a complex task that is beyond what they can already do. Their first step is recruitment where the tutor must get the student to not only interested but taken away from a more immediately interesting task. Their second step is reduction in degrees of freedom which involves simplifying the problem by reducing the number of steps to get to the solution. The third step is direction maintenance which is where the student starts to lag in their work, and the tutor has to keep them engaged and motivated. This is the part where the tutor has to keep the student from spiraling off onto something more immediately fun. The fourth step is marking critical features which is where the tutor finds relevant features of a task to provide information about discrepancies between what a student has produced and what a correction production is. The fifth step is frustration control which is not made to be important because there is more feeling of reward when a student figures out a frustrating problem. The important thing here is there is little reliance on the tutor for direct instruction. The final scaffolding practice is modelling the solutions to a task. The tutor is imitating an attempted solution with expectation that the learner will imitate it back.

This study will apply a scaffolding protocol in the teaching intervention: (a) First allow students to work independently or in small groups through the algorithm. (b) When a student gives up on the problem or veers off onto another topic, guide them back by asking them questions such as where they started to not understand and whether they follow the algorithm, and by asking them to show how they worked the algorithm. (c) Hypothesize about whether the student’s competence is being overwhelmed and then intervene to help them through the part of the problem that they do not understand and then allow them to continue the rest of the algorithm. (d) Help contain a student’s frustration to a healthy level, meaning that some frustration is good, as it motivates and rewards success but high frustration is harmful as it causes a student to give up on the problem. In general, by first letting the student try a problem they not only learn to problem-solve in their own way, but it allows them to also better engage in the problem. When they get frustrated that what they are trying is not working, they can be guided to start trying several mechanisms. This is going to get them to think about how they are thinking.

The current study

Teaching algorithms to students is a frustrating process for the student. Therefore, it is important for there to be some sort of internal motivation. This study hypothesized that scaffolded teaching of algorithmic thinking would engage middle-school students and that if this created intrinsic and utility task value for algorithmic thinking, that it would change middle-school students’ attitudes toward STEM courses and careers. Figure 1 Presents the causal model created to demonstrate how this was hypothesized to take effect.

|  |
| --- |
|  |
| *Figure 1.* Causal model demonstrating how intrinsic and utility task value would be created and how it would change attitude toward STEM courses and careers |

To test this, I proposed a benchtop study exploring (a) the ability of middle-school students to think algorithmically, (b) the intrinsic task value of such work, and (c) its effect on their expectancy for success in mathematics and in academic tasks generally. In a 1-hour session during the regular math period, I asked students to approach a typical middle-school math problem, finding the common divisor between two numbers, first using their own methods and then following a simple algorithm set out by Euclid.

Specific research questions were:

RQ1. Does scaffolded teaching of algorithmic thinking engage middle-school students?

**RQ2.** Do middle-school students find intrinsic task value in scaffolded teaching of algorithmic thinking?

**RQ3.** Does scaffolded teaching of algorithmic thinking change middle-school students’ attitudes toward STEM courses and careers?

To allow students to come to their own methods of following the algorithm, I avoided directly teaching the steps. My role was to scaffold competence in algorithmic thinking and to present challenges that illustrate the unexpected learning possible through the algorithms and the increased efficiency in attacking problems that are meaningful in students' academic work.

Engagement…The intrinsic value of the tasks was assessed through a brief questionnaire before and after each task. Attitudes toward mathematics-related school and career-work were measured with a brief questionnaire before and after the session as well.

Methods

The study used a one-shot pre- and post-test design (Figure 2) with a small volunteer sample of students. Students filled out identical questionnaires on attitudes toward STEM courses and careers before and after the class session. During the intervention, an assistant monitored the fidelity to the intervention schedule and student engagement. Students filled out brief questionnaires on intrinsic and utility task value after each task during the class session.

|  |
| --- |
|  |
| *Figure 2***.** Experimental design illustrating one shot pre and post design |

Population and sample

This targeted middle-school students (grades) typical of …. Participants were 10 middle-schoolers (6th and 8th grade) from an urban high-school in upper east Tennessee at an average age of 13 years.

Measures

The measurements used during this experiment was a combination of Likert scales, written responses to questions, and observations made by an assistant.

Fidelity of implementation. During the class session, an assistant stood aside and observed the instructor to ensure tasks were completed as planned. The assistant checked teacher behavior against the protocol checklist for intervention segments and timing.

Engagement with the academic task. The engagement in the task by the students was measured by the assistant. The assistant observed the class during the session and used a Likert scale to measure the engagement. In 10 second intervals, the assistant would log the number of students engaged in the task.

Intrinsic task value. The measure of intrinsic task value that the students gained was measured by a questionnaire the students filled out after each attempt at the algorithm. The intrinsic value was determined using a using a Likert scale that went from -5 to 5: -5 being least intrinsically valuable and 5 being most intrinsically valuable.

Attitude toward work in STEM. Attitudes toward STEM courses an careers were determined by asking using a Likert scale from 1 to 4: 1 being they would not take this course or do this career and 4 being they would take this course or do this career.

Procedures

Data was collected by a questionnaire used to ask specific questions to analyze the intrinsic task value of the material presented in the class session. The questionnaires were administered before and after the class session and require 20-30 minutes to complete. The questionnaires before the class session were taken home to fill out with the parents and brought back before or the day of the class session. The questionnaires after the class session were completed in the classroom immediately after the session was over.

The intervention schedule is given in Figure 3. To start the class session, 10-minutes is used to introduce the instructor and the class session. Next, approximately 10 minutes is given to introduce Euclid and algorithms generally and to engage the students in thinking about two-column addition as an algorithm. After this introduction, the main part of the class is given 45 minutes. The instructor first presents Euclid’s algorithm and then tasked the students to implement it three times: (A) using the textual version of the algorithm with given inputs M and N; (B) using the text and a flowchart with given inputs, an (C) using text and flowchart with student-selected inputs. Finally, 5 minutes is given to close out the session during which the instructor asks students to comment generally on…

|  |
| --- |
|  |
| Figure 3. Intervention schedule |

During each task-period, the instructor…After each task, students answer questions asked specifically to analyze the intrinsic and utility task value and their attitude of STEM courses and careers of the material presented in the class session.

Results

The intervention schedule was implemented with fidelity. The algorithm introduction was completed thoroughly with the allotted time. Each intervention task, using Euclid’s algorithm, was completed as set out in the schedule, but went over the allotted time slightly. Informal notes suggest the instructor avoided direct teaching and interacted with the students closely. The closing portion of the session was rushed slightly causing the final questionnaire to be rushed as well.

Research Questions

RQ1. Does scaffolded teaching of algorithmic thinking engage middle-school students?

The engagement in the task by the students was measured by the assistant who observed the class during the session using a simple moment-counting procedure. In 10 second intervals, the assistant would log the number of students engaged in the task. Table 1 illustrates the results and Figure 4 shows the distribution of engagement behaviors for each task.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Table 1  Descriptive Statistics of Counts of Academic Behaviors | | | | | | | Task | Academic M (SD) |  | Competing M (SD) |  | Management M (SD) | | A | 8.4 (2.2) |  | 1.6 (2.2) |  | 0.2 (0.1) | | B | 8.7 (1.4) |  | 1.1 (1.3) |  | 0.1 (0.4) | | C | 5.9 (3.1) |  | 3.9 (3.2) |  | 0.3 (0.6) | | Measures are mean (M) counts of students engaged in a behavior at the moment of sampling. | | | | | | |
|  |

The A, B, and C row headings are each task of the session. A represents Euclid’s written step by step algorithm using the inputs M = 73 and N = 45. B represents Euclid’s flow chart of the algorithm using the inputs M = 1225 and N = 70. C represents the students using either Euclid’s written step by step algorithm or Euclid’s flow chart of the algorithm with their choice of the inputs. Academic responses show engagement with the academic task as it is understood by the instructor. Management responses are related to the target task but not direct engagement with the task. Competing responses are responses that are not academic or management responses. M is the median and SD is the standard deviation.

This tables shows that, on average, the students mostly have academic responses to this task. The competing response was there but was very minimal in comparing to the academic response. The SD shows that the data was consistent for both academic, competing, and management responses.

|  |
| --- |
|  |
| *Figure 4***.** Distribution of engagement behaviors for each intervention segment |

For task A, the management and competing responses were very high in the beginning and were absent toward the end. The academic responses were very low and the beginning and grew as the session went on. For task B, the results were much the same as segment A. The management and competing responses were very high in the beginning and were absent toward the end. The academic responses were very low and the beginning and grew as the session went on. For task C, the management responses were very high in the beginning and were absent toward the end. However, the academic responses were not as high in the end as it was for the other segments. The competing responses also went up toward the end of this segment.

**RQ2.** Do middle-school students find intrinsic task value in scaffolded teaching of algorithmic thinking?

The measure of intrinsic task value that the students gained was measured by a questionnaire the students filled out after each attempt at the algorithm. Table 2 illustrates the results and Figure 5 shows the distribution of intrinsic value measures for each intervention segment.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table 2  Descriptive Statistics of Intrinsic Task Value Measures | | | | |
| Task | Enjoyment M (SD) |  | Interest M (SD) |  |
| A | 0.9 (2.0) |  | 2.7 (1.9) |  |
| B | 2.6 (1.8) |  | 3.1 (2.5) |  |
| C | 3.4 (2.1) |  | 3.2 (2.0) |  |
| Measures are on a scale of -5 to +5, with zero meaning “no strong feelings”. | | | | |

The enjoyment for the class session, on average, went up significantly for each segment of the task. The data was less consistent. The interest for the class session, on average, also went up. However, interest went up less significantly than the enjoyment. The data less consistent.

|  |
| --- |
|  |
| Figure **5.** Distribution of intrinsic value measures for each intervention segment |

For task A, enjoyment had almost a split result. About half disliked the activity and the other half was neutral or enjoyed it. However, most students found it interesting. For task B, enjoyment was greater than segment A. All but one was either neutral or enjoyed the activity. Interest increased slightly as well. There were more people who found the activity interesting.

For task C, there was only one person who did not find it enjoyable. Interest went up as well. No one said they found this segment to be uninteresting.

**RQ3.** Does scaffolded teaching of algorithmic thinking change middle-school students’ attitudes toward STEM courses and careers?

The measure of attitude toward STEM courses and careers was measured by a before and after class session questionnaire. Table 3 illustrates the results. Figure 6 shows the mean of the pre- and post-test attitudes toward working in STEM areas and figure 7 illustrates individual ratings. In these data, for courses in school: CSCI is computer programming, Math averages calculus and trig, STEM averages calculus, trig, physics, advanced chemistry, and computer programming. For study/work after high school: CSCI averages programming and computer science, STEM averages math, physics, chemistry, programming, and computer science.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Table 3  Descriptive Statistics of Attitudes Toward Work in STEM fields | | | | | | | | | |
| Timing | In High School | | | |  | After High School | | | |
|  | CSCI M (SD) | Math M (SD) | Physics M (SD) | STEM M (SD) |  | CSCI M (SD) | Math M (SD) | Physics M (SD) | STEM M (SD) |
| Pre | 3.1 (0.7) | 3.1 (0.6) | 3.3 (0.5) | 3.0 (0.5) |  | 2.4 (1.1) | 2.7 (0.7) | 2.6 (0.8) | 2.7 (0.7) |
| Post | 2.3 (0.8) | 3.0 (0.6) | 3.3 (0.7) | 2.8 (0.4) |  | 2.3 (0.9) | 2.8 (0.6) | 2.7 (1.0) | 2.6 (0.7) |
| Measures are on a scale of 1 (Absolutely not) to 4 (Absolutely yes) | | | | | | | | | |

On average, students’ attitudes toward STEM courses in high school went down after the class session for each course. Almost the same results for after high school careers. There were a few results that went up. The SD shows the data was consistent for both measures.

In individuals’ scorings, there are significant decreases for CSCI and STEM courses in several individuals and a few for Physic courses. There are a few that increased their attitudes toward Physic courses. Math courses either went down or stayed neutral. Most individual students had a neutral or negative attitude toward CSCI careers. However, note that students with ID’s 171 and 754 increased their attitude toward CSCI careers even though they had a negative attitude toward CSCI courses in high school. A similar result happened for Pysics careers too for students with ID’s 171, 215, and 877. There was also an increase in attitude toward math careers between students with ID’s 171 and 877. STEM careers stayed at a lower attitude rating.

|  |
| --- |
|  |
| *Figure 6.* Mean pre/post attitude toward working in STEM areas. |

|  |  |
| --- | --- |
|  |  |
| *Figure 7***.** Individual pre/post attitude toward taking STEM courses in high school | |

**Discussion**

Significance

The significance is that by finding intrinsic value gained from teaching middle-school students algorithmic thinking, they will have deeper comprehension of academic material and therefore have more expectancy for success in a wide array of options. That will then affect their achievement related choices, engagement, and performance. Finally, it will demonstrate success in the previously mentioned major fields.

Furthermore, algorithmic thinking will generally affect expectancy for success in academic tasks. I also hypothesize that this confidence will change what professions students go into. By introducing this in middle-school, it allows students to decide the profession they will reach for before they decide what classes to take in high-school.

It is hypothesized that middle-school students will like the way algorithmic thinking changes the way they think, and they will find intrinsic task value in algorithmic thinking. Therefore, these students will better engage in these algorithmic thinking tasks because of their intrinsic task value. While being engaged and also learning to problem solve in their own way, students will have more confidence in their academic ability.

References

Buitrago Flórez, F., Casallas, R., Hernández, M., Reyes, A., Restrepo, S., & Danies, G. (2017). Changing a Generation’s Way of Thinking: Teaching Computational Thinking Through Programming. *Review of Educational Research*, *87*(4), 834–860. https://doi.org/10.3102/0034654317710096

Chiu, M. M., & Xihua, Z. (2008). Family and motivation effects on mathematics achievement: Analyses of students in 41 countries. *Learning and Instruction*, *18*(4), 321–336. https://doi.org/10.1016/j.learninstruc.2007.06.003

Christoph, G., Goldhammer, F., Zylka, J., & Hartig, J. (2015). Adolescents’ computer performance: The role of self-concept and motivational aspects. *Computers & Education*, *81*, 1–12. https://doi.org/10.1016/j.compedu.2014.09.004

Denner, J., Werner, L., Bean, S., & Campe, S. (2005). The girls creating games program: Strategies for engaging middle-school girls in information technology. *Frontiers: A Journal of Women Studies*, *26*(1), 90–98.

Eccles, J. S., Barber, B. L., Updegraff, K., & O’Brien, K. M. (1998). An expectancy-value model of achievement choices: The role of ability self-concepts, perceived task utility and interest in predicting activity choice and course enrollment. *Interest and Learning*, 267–280.

Fredricks, J. A., & Eccles, J. S. (2002). Children’s competence and value beliefs from childhood through adolescence: Growth trajectories in two male-sex-typed domains. *Developmental Psychology*, *38*(4), 519–533. https://doi.org/10.1037/0012-1649.38.4.519

Harackiewicz, J. M., Barron, K. E., Tauer, J. M., & Elliot, A. J. (2002). Predicting success in college: A longitudinal study of achievement goals and ability measures as predictors of interest and performance from freshman year through graduation. *Journal of Educational Psychology*, *94*(3), 562–575. https://doi.org/10.1037/0022-0663.94.3.562

Knuth, D. (1968). The art of computer programming 1: Fundamental algorithms 2: Seminumerical algorithms 3: Sorting and searching. *MA: Addison-Wesley*, *30*.

Papert, S. (1980). Mindstorms: children, computers, and powerful ideas.

Ryan, R. M., & Deci, E. L. (2000). Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being. *American Psychologist*, *55*(1), 68–78. https://doi.org/10.1037/0003-066X.55.1.68

Spinath, B., Spinath, F. M., Harlaar, N., & Plomin, R. (2006). Predicting school achievement from general cognitive ability, self-perceived ability, and intrinsic value. *Intelligence*, *34*(4), 363–374. https://doi.org/10.1016/j.intell.2005.11.004

Tennessee Department of Education. (2017). Math Standards. Retrieved August 24, 2018, from https://www.tn.gov/education/instruction/academic-standards/mathematics-standards.html

Torbert, S., Vishkin, U., Tzur, R., & Ellison, D. J. (2010). Is teaching parallel algorithmic thinking to high school students possible?: one teacher’s experience. In *Proceedings of the 41st ACM technical symposium on Computer science education* (pp. 290–294). ACM.

Werner, L., Denner, J., Bliesner, M., & Rex, P. (2009). Can middle-schoolers use Storytelling Alice to make games?: results of a pilot study. In *Proceedings of the 4th International Conference on foundations of digital games* (pp. 207–214). ACM.

Werner, L., Denner, J., Campe, S., & Kawamoto, D. C. (2012). The fairy performance assessment: measuring computational thinking in middle school. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education* (pp. 215–220). ACM.

Wing, J. M. (2006). Computational Thinking. *Communications of the ACM*, *49*(3), 33–35.

Wood, D., Bruner, J. S., & Ross, G. (1976). THE ROLE OF TUTORING IN PROBLEM SOLVING. *Journal of Child Psychology and Psychiatry*, *17*(2), 89–100. https://doi.org/10.1111/j.1469-7610.1976.tb00381.x