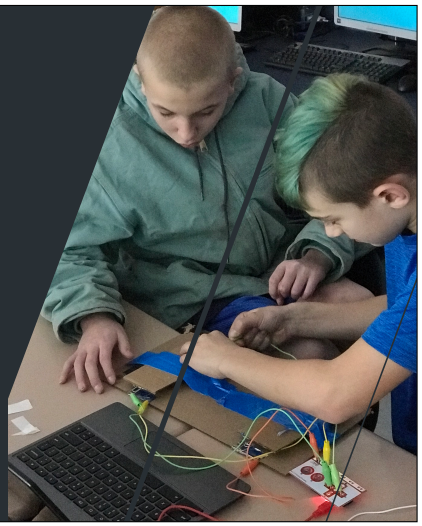
EVIDENCE-BASED INTEGRATION OF CS INSTRUCTION

Dr. Christopher Harris

COMPUTATIONAL THINKING FOR CS

- Seymour Papert (1980) coined the term Computational Thinking as a subset of and approach to Computer Science
- Jeanette Wing (2006) revived the view that CT should be foundational for all students - more conceptual and cross disciplinary than CS/coding approach
- Starting with CT led to higher scores on a programming quiz vs traditional coding approach (Duncan & Bell, 2015)



CT IMPACTS ON STUDENT SUCCESS

OLIVERIA ET AL. (2014)
ROMÁN-GONZÁLEZ ET AL. (2017)
CHEN ET AL. (2017)

CT CORRELATED WITH CORE SUBJECT SCORES

CT LINKED TO VERBAL, SPATIAL, REASONING

CT IMPACTS STUDENTS' EVERYDAY REASONING

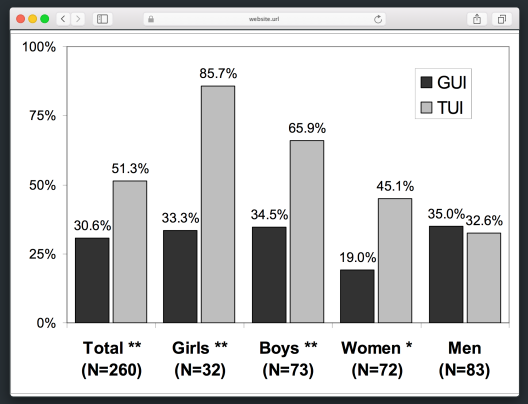
RESEARCH SHOWS YOU ALREADY HAVE MASTERY OF CT CONCEPTS

94%
ROBOTIC CODING FOR DIRECITONS

87%
IF-THEN CODING FOR CONDITIONALS

66%
OVERALL AVERAGE ON CT SKILLS TEST

Average scores of K-2 teachers on Computational Thinking Test prior to any professional development on CT or CS.



START ANALOG FOR CS SUCCESS

Horn et al. (2009) found that boys, girls, and women (i.e. most elem. teachers) had higher rates of interaction with analog/tangible (TUI) vs. digital/graphical (GUI) controls at a science museum exhibit.

STARTING ANALOG CAN HELP STUDENTS LEARN CS

- A study of 5-7 year old students found higher mastery of CS concepts using analog vs digital (Wohl et al., 2015)
- Digital learning focuses students on the tool while analog learning led to more conceptual undertstanding.
- Other studies (Horn et al., 2012; Strawhacker & Bers, 2015) found similar learning, but highlighted benefits of an analog approach for group instruction.

CT SKILLS DEFINED

Decomposition

Pattern Recognition

Abstraction

Algorithm Design

Evaluation

CT SKILLS INTEGRATED ACROSS THE ELEMENTARY CURRICULUM

DECOMPOSITION

- What are the steps in the water cycle?
- What is the main idea of this story?
- Multiplication is just adding lots of times

PATTERN RECOGNITION

- If-Then is the same as cause and effect
- How are rural and urban areas alike and different?
- Is there a trend for snowfall over the last 10 years?

ABSTRACTION

- What factors cause people to immigrate?
- Pseudocode as a writing style for early code
- Finding the question in a word problem

ALGORITHM DESIGN

- How do you multiply two fractions?
- Following steps in a science experiment
- This is the class process for a fire drill

/RABBITS AND WOLVES SIMULATION

- **Decomposition** of predator-prey interaction
- **Pattern recognition** through analysis of animal populations over time
- **Abstraction** of complex food web interaction into a simple game
- Following an **Algorithm** designed for successful completion of simulation

Rabbits and Wolves

Number of players: 2

ACTIVITY OVERVIEW

The Rabbits and Wolves simulation is a famous model for how predators and prey interact in nature. The simulation shows how populations grow and decline over time and what happens when there are either too many predators or too few.

DIRECTIONS

Setup

You will need to print out the meadow, the page with the Rabbit and Wolf tokens, and the data record sheet. Cut out the Rabbit and Wolf tokens. Players will also need a pen or pencil to record the population status on the data record sheet.

Place the meadow between the two players and give each player their stack of either Rabbit or Wolf tokens. The game starts with an empty meadow.

Before the first round of the game, the Rabbit player receives three Rabbit tokens and the Wolf player receives one Wolf token.

Each round

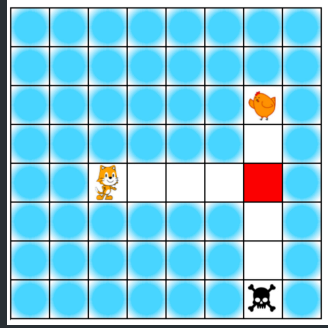
There are four steps in each round of play.

1. The Rabbit player places any Rabbit tokens that are available (either from the start of the game or a new set of rabbits) into the meadow. No overlapping!
2. The Wolf player then takes each Wolf token and tosses it into the meadow to try and "catch" rabbits. The Wolf token must be tossed from outside of the meadow.
3. Population check. Check each Wolf token in the meadow. The Wolf player looks at

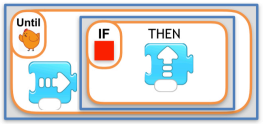
HARRIS (2018)

/TEACHER CONFIDENCE WITH CS INSTRUCTION

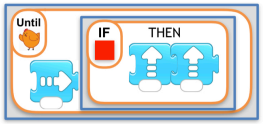
Which instructions take the cat to the chicken by the path marked out?



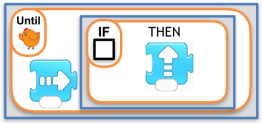
Option A



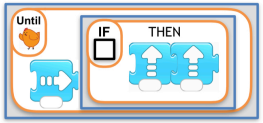
Option B



Option C



Option D



/LET'S DO SOME CS!

Computational Thinking Test by Román-González (2015) revised by Harris (2018)

CONFIDENCE IN CS IS THE CRITICAL ELEMENT

- Teacher confidence is linked to importance given to and time spent on a subject. .
- Teacher: Confidence with math is "something I need to work on because I'm trying to give something to the next generation, to give them groundwork and the interest in maths, not just literacy."

(Cohrssen et al., 2016, p. 8)

MEASURING CONFIDENCE

"The teachers we have worked with throughout this, and previous studies, were more often than not anxious about teaching CS and programming concepts, and think they are not capable of doing this well."

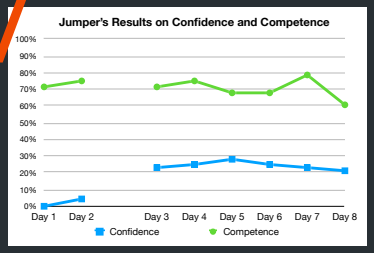
(Duncan, Bell, & Atlas, 2017. p. 5)

Elementary Teacher Computer Programming Self Efficacy Scale

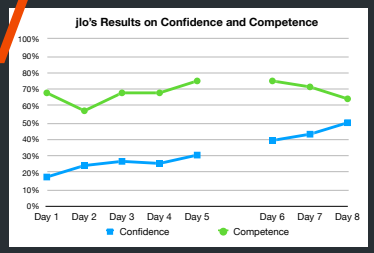
This rating scale lists different instructional scenarios where you might be teaching about computer programming. You will be asked to rate how confident you are that you can complete these instructional tasks as of now. Rate your degree of confidence by moving the slider, or clicking on the slider bar at the desired point, to indicate your confidence using the scale of 0 (Not at all Confident) to 10 (Highly Confident).

1. I can teach my students programing concepts using a tool like Scratch Jr. or Robot Turtles.
2. I can teach students how to identify a correct sequence of smaller steps to solve a larger programming task.
3. I can teach the use of If…Then…Else conditional programming statements.
4. I can teach the use of control flow in programming like loops or while statements.
5. I can teach about variables as a way to hold changing values of data.
6. I can teach students how to find repeating steps in a program that could become a function.
7. I can teach students how to package a section of code into a function to reuse it later.
8. I can teach about breaking down complex programming problems into smaller steps that can be more easily solved.
9. I can teach students to step through a program to identify and solve bugs.
10. I can teach about remixing existing code snippets into new programs to solve tasks.
11. I can teach students how to explain the reasoning behind their programming solution to a problem.
12. I can teach programming as an iterative process of incremental work to solve a problem.
13. I can teach my students how functions help make programming more efficient by addressing steps in a larger program.
14. I can teach students to identify computational problems in their daily lives.
15. I can teach students how to identify a computational problem and design a working solution to complete a task.

RESULTS - DIGITAL

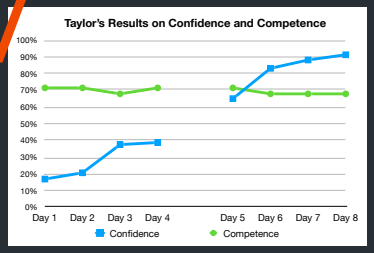


JUMPER

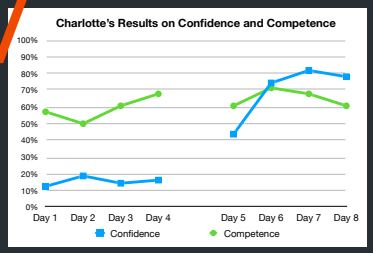


JLO

RESULTS - ANALOG



TAYLOR



CHARLOTTE

/REFERENCES

Berland, M., & Wilensky, U. (2015). Comparing virtual and physical robotics environments for supporting complex systems and computational thinking. Journal of Science Education and Technology, 24(5), 628-647.

Chen, G., Shen, J., Barth-Cohen, L., Jiang, S., Huang, X., & Eltoukhy, M. (2017). Assessing elementary students' computational thinking in everyday reasoning and robotics programming. Computers & Education, 109, 162-175. doi: 10.1016/j.compedu.2017.03.001

Cohrssen, C., Church, A., & Tayler, C. (2016). Play-based mathematics activities as a resource for changing educator attitudes and practice. SAGE Open, 6(2), 2158244016649010. doi: 10.1177/2158244016649010

Cohrssen, C., Tayler, C., & Cloney, D. (2015). Playing with maths: Implications for early childhood mathematics teaching from an implementation study in Melbourne, Australia. Education 3-13, 43(6), 641-652. doi: 10.1080/03004279.2013.848916

Duncan, C., & Bell, T. (2015). A pilot computer science and programming course for primary school students. In Proceedings of the Workshop in Primary and Secondary Computing Education (pp. 39-48). New York, NY: ACM. doi: 10.1145/2818314.2818328

Duncan, C., Bell, T., & Atlas, J. (2017). What do the teachers think?: Introducing computational thinking in the primary school curriculum. In Proceedings of the Nineteenth Australasian Computing Education Conference (pp. 65-74). New York, NY, USA: ACM. doi: 10.1145/3013499.3013506

Geist, E. (2015). Math anxiety and the "math gap": How attitudes toward mathematics disadvantages students as early as preschool. Education, 135(3), 328-336.

Grover, S., & Pea, R. (2013). Computational thinking in k-12 a review of the state of the field. Educational Researcher, 42(1), 38-43. doi: 10.3102/0013189X12463051

Harris, Christopher, "Computational Thinking Unplugged: Comparing the Impact on Confidence and Competence from Analog and Digital Resources in Computer Science Professional Development for Elementary Teachers" (2018). Education Doctoral. Paper 374. https://fisherpub.sjfc.edu/education_etd/374

Horn, M. S., Solovey, E. T., Crouser, R. J., & Jacob, R. J. K. (2009). Comparing the use of tangible and graphical programming languages for informal science education. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (pp. 975-984). New York, NY, USA: ACM. doi: 10.1145/1518701.1518851

Oliveira, O. L., Nicoletti, M. C., & del Val Cura, L. M. (2014). Quantitative correlation between ability to compute and student performance in a primary school. In Proceedings of the 45th ACM Technical Symposium on Computer Science Education (pp. 505-510). New York, NY: ACM. doi: 10.1145/2538862.2538890

Papert, S. (1980). Mindstorms: Children, Computers, and Powerful Ideas. New York, NY: Basic Books, Inc.

Román-González, M. (2015). Computational thinking test: design guidelines and content validation. In Proceedings of EDULEARN15 Conference (pp. 2436-2444). Barcelona, Spain: IATED. doi: 10.13140/RG.2.1.4203.4329

Román-González, M., Pérez-González, J.-C., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. Computers in Human Behavior, 72, 678-691. doi: 10.1016/j.chb.2016.08.047

Strawhacker, A., & Bers, M. (2015). "I want my robot to look for food": Comparing kindergartner's programming comprehension using tangible, graphic, and hybrid user interfaces. International Journal of Technology & Design Education, 25(3), 293-319. doi: 10.1007/s10798-014-9287-7

Wing, J. M. (2006). Computational thinking. Communications of the ACM, 49(3), 33-35. doi: 10.1145/1118178.1118215