Integrating Computational Thinking into the Curriculum for Pre-Service Elementary and Middle School STEM Teachers

Rachel F. Adler

Joseph Hibdon, Jennifer Slate, Sudha Srinivas, Durene Wheeler, Hanna Kim, Scott Mayle, and Brittany Pines

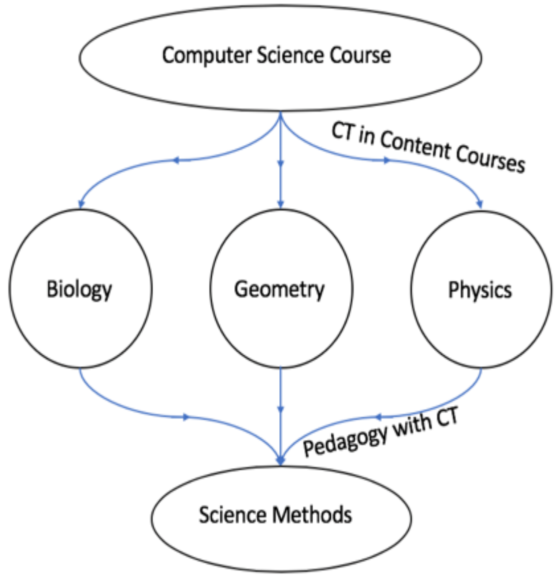
NSF #1640041

Northeastern Illinois University

Introduction

- The Math, Science, and Technology for Quality Education (MSTQE) at Northeastern Illinois University (NEIU) is an interdisciplinary undergraduate math and science content preparation program for preservice elementary and middle school teachers.
- It is a Bridge program with students from Chicago Community Colleges who take classes together with NEIU students.
- As part of an NSF STEM+C grant, we integrated computational thinking (CT) into the curriculum for our MSTQE Program.

Integration of CT into the Curriculum



- New Computer Science for All course
  - Overview of CT, Scratch, Robotics, VPython/Python

- Integration of CT into STEM Content Courses
  - Biology - NetLogo to model an epidemic
  - Physics - VPython to learn about physical concepts in mechanics (vectors, motion, forces, energy)
  - Geometry - Scratch to model geometric concepts and art

- Integration of CT into Teaching Methods Course
  - Science Methods - Robotics to simulate earthquakes

Methodology

- We used the following methods of assessment:
  - Surveys - Students completed both a pre- and post-semester survey rating their self-efficacy in CT.
  - Focus Group/Interviews - Student Focus Groups/Faculty Interviews
  - Rubric – Computational Thinking Rubric

Results - Survey

- CT Scale:
  - Loaded as one factor with loadings of .71 or higher (two items were removed).
  - High level of reliability (Cronbach’s Alpha of .92 on the pre-survey and .94 on the post).
- Student’s self-efficacy in CT improved from the beginning (M=3.19, SD=0.88) to end (M=3.99, SD=0.70) of the semester, t(79)=-6.45, p<.0001.
  - *I am able to break a complex problem into smaller, more manageable parts or components so that it can be solved using a computer.*
  - *I am able to manipulate a system's variables or components to achieve a desired result.*
  - *I am able to modify existing computer code to complete small tasks in subject areas I am familiar with.*
  - *I am able to create computer code to complete small tasks in subject areas I am familiar with.*
  - *I can analyze or interpret a program's output or data.*
  - *I understand how computational skills/tools could be applied to a variety of topics.*
  - *I understand how computers can be programmed to develop solutions to problems.*
  - *I understand how computers can be used to model phenomena.*
  - *I am confident in my ability to use computational thinking to understand or analyze problems.*

Results - Student Focus Groups / Faculty Interviews

- Focus groups with students in courses revealed that students generally expressed increases in confidence with computing.

- CS for All students expressed the most confidence in their abilities to learn computing and to teach computing.

- In faculty interviews, faculty also commented that they noticed increases in students’ confidence in computing and CT during the implementation of the modules.

Results – CT Rubric

- We also created a rubric to assess CT skills in our participants. (More on that in tomorrow's assessment session.)

Criteria	No/Limited Proficiency	Some Proficiency	Proficiency	High Proficiency
Deconstruct a problem into smaller, more manageable parts	Not able to break down the problem	Recognizes some parts of the problem but unable to identify key contributing components	Identifies the key components that contribute to the problem	Identifies the key components that contribute to the problem and describes their significance
Ability to manipulate variables/parameters for desired result	Unable to differentiate between different variables/parameters or determine their effect upon the model	Understands significance of variables/parameters, but cannot properly change them to cause variations in the model	Has good grasp of significance of variables/parameters; is able to manipulate models accordingly after initial instruction	Has excellent grasp of variables/parameters; is able to change models correctly with little guidance
Analyze and interpret program output or data	Unable to describe program output or data	Describes program output or data, but incorrectly interprets the meaning	Correctly interprets the meaning of the program output or data	Correctly interprets the meaning of the program output or data and draws conclusions within context of the program's limitations
Use algorithmic thinking to modify or construct computer code	Cannot develop steps to modify or construct computer code	Begins to develop steps to modify or construct computer code, but some steps are missing or not in a logical order	Series of steps to modify or construct computer code is complete and in logical order	Series of steps to modify or construct computer code is complete, in logical order, and efficient
Generalize to another problem or real-world situation	Cannot describe how the program could relate to another problem or situation	Can relate the program to another problem or situation, but cannot identify underlying pattern(s)	Identifies pattern(s) in the problem solving process and relates the pattern(s) to another problem or situation	Identifies and analyzes pattern(s) in the problem solving process and can justify generalization to another problem or situation

Results – CT Rubric, cont.

- The CT rubric scores from completed by instructors were high.
- Students exhibited the most difficulty with algorithmic thinking, and had the highest scores in analyzing output and data.

Criteria	No/Limited Proficiency (%)	Some Proficiency (%)	Proficiency (%)	High Proficiency (%)
Decomposition	0 (0%)	5 (11%)	25 (56%)	15 (33%)
Variables	1 (2%)	3 (7%)	26 (58%)	15 (33%)
Data	0 (0%)	2 (4%)	25 (56%)	18 (40%)
Algorithmic Thinking	1 (2%)	6 (13%)	20 (44%)	18 (40%)
Generalization	2 (4%)	4 (9%)	13 (29%)	26 (58%)

Conclusion

- Rather than modifying a single course, our approach integrates CT in a curriculum for educators.
- Our results show that at the end of the semester, students self-efficacy in CT increased.
- In addition, instructors assessment of students’ CT skills after grading student projects were high.

Future Work

- Introduce in-service teachers to the modules
- Include CT in other pre-service math courses
- Continue to collect more data from students using the modules