Computational thinking

Conference Paper *in* ACM SIGCSE Bulletin · March 2007

DOI: 10.1145/1227310.1227378 · Source: DBLP

3 authors:

Peter B. Henderson
Butler University
144 PUBLICATIONS **584** CITATIONS
SEE PROFILE

Thomas J. Cortina
Carnegie Mellon University
13 PUBLICATIONS **164** CITATIONS
SEE PROFILE

Jeannette M. Wing
Carnegie Mellon University
234 PUBLICATIONS **15,998** CITATIONS
SEE PROFILE

Computational Thinking

Peter B. Henderson
(Moderator)
Department of Computer Science and Software Engineering
Butler University
Indianapolis, IN 46208 (USA)
(317) 940-9513
phenders@butler.edu

Thomas J. Cortina
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
(412) 268-3514
tcortina@cs.cmu.edu

Orit Hazzan
Department of Education in Technology and Science
Technion – Israel Institute of Technology
Haifa 32000, ISRAEL
(9724) 829-3107
oritha@techunix.technion.ac.il

Jeannette M. Wing
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
(412) 268-3068
wing@cs.cmu.edu

1. SUMMARY

Computational Thinking [1] is a universal metaphor of reasoning used by both mankind and machines. From this perspective it has the potential to be a comprehensive umbrella for capturing the intrinsic nature of computing and conveying this in an understandable way to students and the general public. It represents a broad spectrum of reasoning across time and disciplines. Learning to count is a beginning of human computational thinking, followed naturally by arithmetic computation and abstract levels of symbol based thinking, often starting with algebra. Counting, arithmetic, symbols and abstract thinking are fundamental to the study of computing.

Computational reasoning is the core of all modern Science, Technology, Engineering and Mathematics (STEM) disciplines and is intrinsic to all other disciplines from A to Z. It is used in our everyday lives from baking a cake, changing a tire or brushing our teeth. The human brain is wired to think computationally, as are modern computing devices. As educators, a Computational Thinking perspective can help us to convey fundamental computing ideas to all students. This special session will outline the principles of Computational Thinking, offer suggestions on ways to promote Computational Thinking at all educational levels, and provide ample time for audience participation and discussion.

Categories and Subject Descriptors

K.3 [**Computing Milieux**]: Computer and Education
K.3.2 *Computer and Information Science Education*

General Terms

Design, Algorithms

SIGCSE'07, March 7-10, 2007, Covington, Kentucky, USA.
ACM 1-59593-361-1/07/0003.

Keywords

Computational Thinking

2. OBJECTIVES

There is general consensus that the poor public perception of careers in computing impacts the attraction of talented students to core computing majors. Much of this is due to the immaturity of the discipline. If you ask 50 different computing professionals or educators to describe the discipline, then you are likely to get 50 different answers, many of which might include the word 'programming' [2, 3]. People can look at the night sky to better understand what an astronomer does, whereas using a computer, cell phone, pda, etc. provides little insight into what a computing professional does. One approach to addressing this is to educate the general public about the varied facets of careers in computing (e.g., computer science, software engineering, information technology, information systems, computer engineering, informatics, computer technician, database administrator, etc.). An alternative approach is to capture the intrinsic nature of computing in ways which are understandable to the general public by making connections to ideas and concepts they are familiar and comfortable with. Computational Thinking has this potential, especially in the education of our young people.

This special session has the following primary objectives:

- To convey the fundamental ideas underlying Computational Thinking.
- To present and discuss ways in which Computational Thinking can impact education at both the pre-college and college levels.
- To examine a prototype first course founded on the principles of Computational Thinking.
- To give concrete ideas for curriculum reform building on the Computational Thinking metaphor.

- To discuss the general concept of Computational Thinking and its potential for achieving the goals presented in this special session description.
- To discuss more effective ways to convey to the general public, perspective students and their parents the true nature of careers in computing.

3. OUTLINE OF THE SESSION

- *Computational Thinking (15 minutes)*, Jeannette Wing will present the highlights of Computational Thinking described in her CACM Viewpoint article [1].
- *Potential influence of Computational Thinking on K-12 Education (10 minutes)*, Peter Henderson will describe how Computational Thinking might be integrated into the existing K-12 curriculum to minimize disruption to the curriculum and maximize the connections and understanding of computing. Similar ideas are currently being discussed by educational policy makers in the United Kingdom as they rethink their science K-12 and undergraduate curricula.
- *Potential influence of Computational Thinking on Undergraduate Education (10 minutes)*, Orit Hazzan will present the benefits of Computational Thinking in an undergraduate curriculum. She will describe ways in which Computational Thinking might be introduced early to all undergraduate students, and used and reinforced in subsequent courses. Special emphases will be on developing abstract thinking skills.
- *A Computational Thinking based introductory course (10 minutes)*, Tom Cortina will present experiences with an evolving introductory course "Principles of Computation," for non-computing majors, which is based upon the principles of Computational Thinking. This course model can be adapted for use by high school teachers either as part of a traditional programming class or as part of a new course to illustrate the relevance of studying computing in student's lives.
- *Discussion, Q & A (30 minutes)*

Actual presentation time will be shorter than the time specified to permit clarifying questions and answers.

4. EXPECTATIONS

Defining and better understanding the discipline of computing should be of interest to all SIGCSE attendees. Computational Thinking strives to define the computing disciplines, not to redefine them.

Participants will gain a better understanding of the ideas underlying Computational Thinking and the potential these ideas have for influencing the education of students of all ages throughout the world. They will understand, discuss and debate the ways in which Computational Thinking can encompass all of overarching concepts which are fundamental to the study of computing (e.g., abstract thinking, general problem-solving, algorithmic and mathematical reasoning, cognition, and scientific and engineering based thinking, computational sciences) while at the same time making connections to relevant ideas and concepts in the traditional curriculum. Tangible ways for achieving this and for initiating curriculum reform founded on Computational Thinking will be one expectation. In addition, participants will be exposed to ideas for challenging the traditional "computing career implies programming" viewpoint.

Computational Thinking represents an alternative way of packaging, presenting, understanding and studying computing. Teaching students to think computationally is an important life skill. Accordingly, these ideas can help to define new ways of exciting future generations about the value of careers in computing.

Copies of the presentations, key papers, and the highlights of the resulting discussion will be made available on a web page for everyone to access.

5. REFERENCES

[1] Wing, J.M., Computational Thinking, CACM, Vol. 49, No. 3 (March 2006) pp 33-35.

[2] Morris, J., Programming Doesn't Begin to Define Computer Science, Pittsburgh Post-Gazette, Sunday, July 4, 2004.

[3] Denning, P., McGettrick, A., Rosenbloom, P., and Snyder, L., Re-Centering Computer Science, ACM SIGCSE Bulletin, Vol. 38, No. 1 (March 2006) pp.65-66.