Computational Thinking in K-12 and Scalable Game Design

Michael Shuffett

Virginia Tech

Shuchi Grover and Roy Pea

Computational Thinking in K-12: A Review of the State of the Field

Summary

- What and Why of Computational Thinking
- Summary of Pertinent Research on CT in K-12
  - Environments and Tools That Foster CT
  - Assessment of CT
- Computing Education in K-12
- Broadening the Scope of the Discourse and Priorities for Empirical Inquiry

"computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science" – Wing (2006)

"Computational thinking is the thought process involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent." – Wing (2011)

"their solutions can be represented as computational steps and algorithms" – Aho (2012)

Seven Big Ideas

1. **Computing** is a *creative* human activity
2. **Abstraction** reduces information and detail to focus on concepts relevant to understanding and solving problems
3. **Data** and **information** facilitate the *creation of knowledge*
4. **Algorithms** are tools for developing and expressing solutions to *computational problems*
5. **Programming** is a *creative process* that produces *computational artifacts*
6. Digital devices, systems, and the networks that interconnect them enable and foster computational approaches to solving problems
7. **Computing** enables *innovation* in other fields, including science, social science, humanities, arts, medicine, engineering, and business.

Computational thinking

vs.

computational literacy

Widely Accepted as Comprising CT

- Abstractions and pattern generalizations (including models and simulations)
- Systematic processing of information
- Symbol systems and representations
- Algorithmic notions of flow of control
- Structured problem decomposition (modularizing)
- Iterative, recursive, and parallel thinking
- Conditional logic
- Efficiency and performance constraints
- Debugging and systematic error detection

Alexander Repenning, David Webb, Andri Ioannidou

Scalable Game Design and the Development of a Checklist for Getting Computation Thinking into Public Schools

Summary

- Introduction: The Scalable Game Design Initiative
- Computational Thinking Checklist
- Discussion

"motivational concerns need to be addressed at the middle school level"

Computational Thinking Tools

1. ***has low threshold***: a student can produce a working game quickly.
2. ***has high ceiling***: a student can make a real game that is playable and exhibits sophisticated behavior, e.g., complex AI.
3. ***scaffolds Flow***: the curriculum provides stepping stones with managed skills and challenges to accompany the tool.
4. ***enables transfer***: tool + curriculum must work for both game design and subsequent computational science applications as well as support transfer between them.
5. ***supports equity***: game design activities should be accessible and motivational across gender and ethnicity boundaries.
6. ***systemic and sustainable***: the combination of the tool and curriculum can be used by all teachers to teach all students (e.g. support teacher training, standards alignment etc).