Computational Thinking

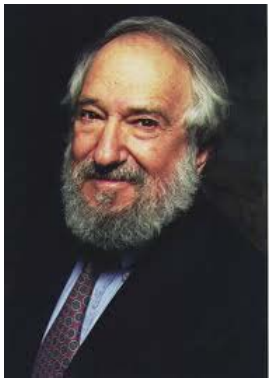
Class Overview

web site: www.cs.vt.edu/~kafura/CS6604

Virginia Tech

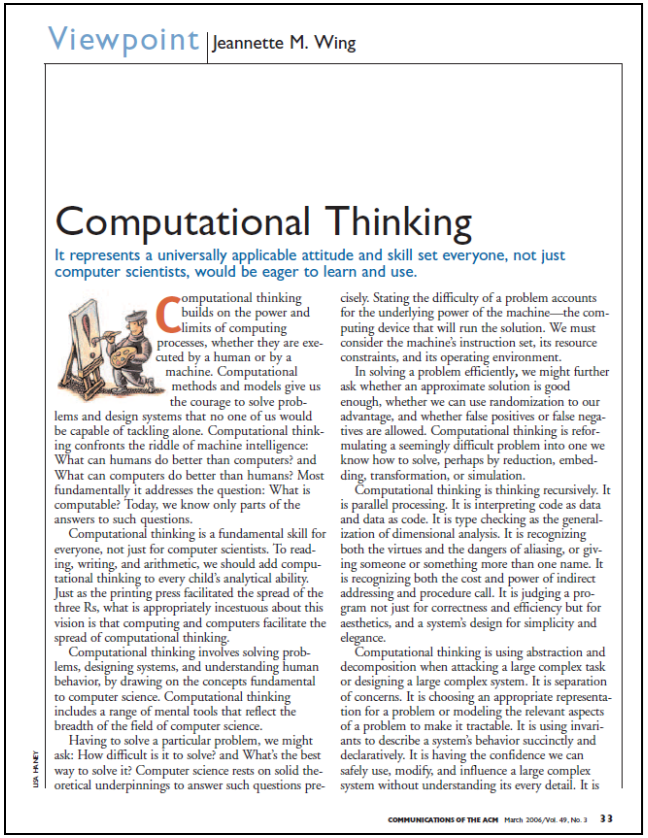
Origins

- Term first used by Seymour Papert (1996) [Snow 2012]
  - "In both cases the computer used as a tool effectively leads to a solution, but in neither does the computational representation make the mathematics more perspicuous. ... The goal is to use computational thinking to forge ideas that are at least as 'explicative' as the Euclid-like constructions (and hopefully more so) but more accessible and more powerful."



Origins

- Recent popularization by Jeannette Wing [Wing 2006]
  - "Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science. Computational thinking includes a range of mental tools that reflect the breadth of the field of computer science."



Viewpoint | Jeannette M. Wing

Computational Thinking

It represents a universally applicable attitude and skill set everyone, not just computer scientists, would be eager to learn and use.

Computational thinking builds on the power and limits of computing processes, whether they are executed by a human or by a machine. Computational methods and models give us the courage to solve problems and design systems that no one of us would be capable of tackling alone. Computational thinking confronts the riddle of machine intelligence: What can humans do better than computers? and What can computers do better than humans? Most fundamentally it addresses the question: What is computable? Today, we know only parts of the answers to such questions.

Computational thinking is a fundamental skill for everyone, not just for computer scientists. To reading, writing, and arithmetic, we should add computational thinking to every child's analytical ability. Just as the printing press facilitated the spread of the three Rs, what is appropriately incestuous about this vision is that computing and computers facilitate the spread of computational thinking.

Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science. Computational thinking includes a range of mental tools that reflect the breadth of the field of computer science.

Having to solve a particular problem, we might ask: How difficult is it to solve? and What's the best way to solve it? Computer science rests on solid theoretical underpinnings to answer such questions precisely. Stating the difficulty of a problem accounts for the underlying power of the machine—the computing device that will run the solution. We must consider the machine's instruction set, its resource constraints, and its operating environment.

In solving a problem efficiently, we might further ask whether an approximate solution is good enough, whether we can use randomization to our advantage, and whether false positives or false negatives are allowed. Computational thinking is reformulating a seemingly difficult problem into one we know how to solve, perhaps by reduction, embedding, transformation, or simulation.

Computational thinking is thinking recursively. It is parallel processing. It is interpreting code as data and data as code. It is type checking as the generalization of dimensional analysis. It is recognizing both the virtues and the dangers of aliasing, or giving someone or something more than one name. It is recognizing both the cost and power of indirect addressing and procedure call. It is judging a program not just for correctness and efficiency but for aesthetics, and a system's design for simplicity and elegance.

Computational thinking is using abstraction and decomposition when attacking a large complex task or designing a large complex system. It is separation of concerns. It is choosing an appropriate representation for a problem or modeling the relevant aspects of a problem to make it tractable. It is using invariants to describe a system's behavior succinctly and declaratively. It is having the confidence we can safely use, modify, and influence a large complex system without understanding its every detail. It is

Purposes of Course

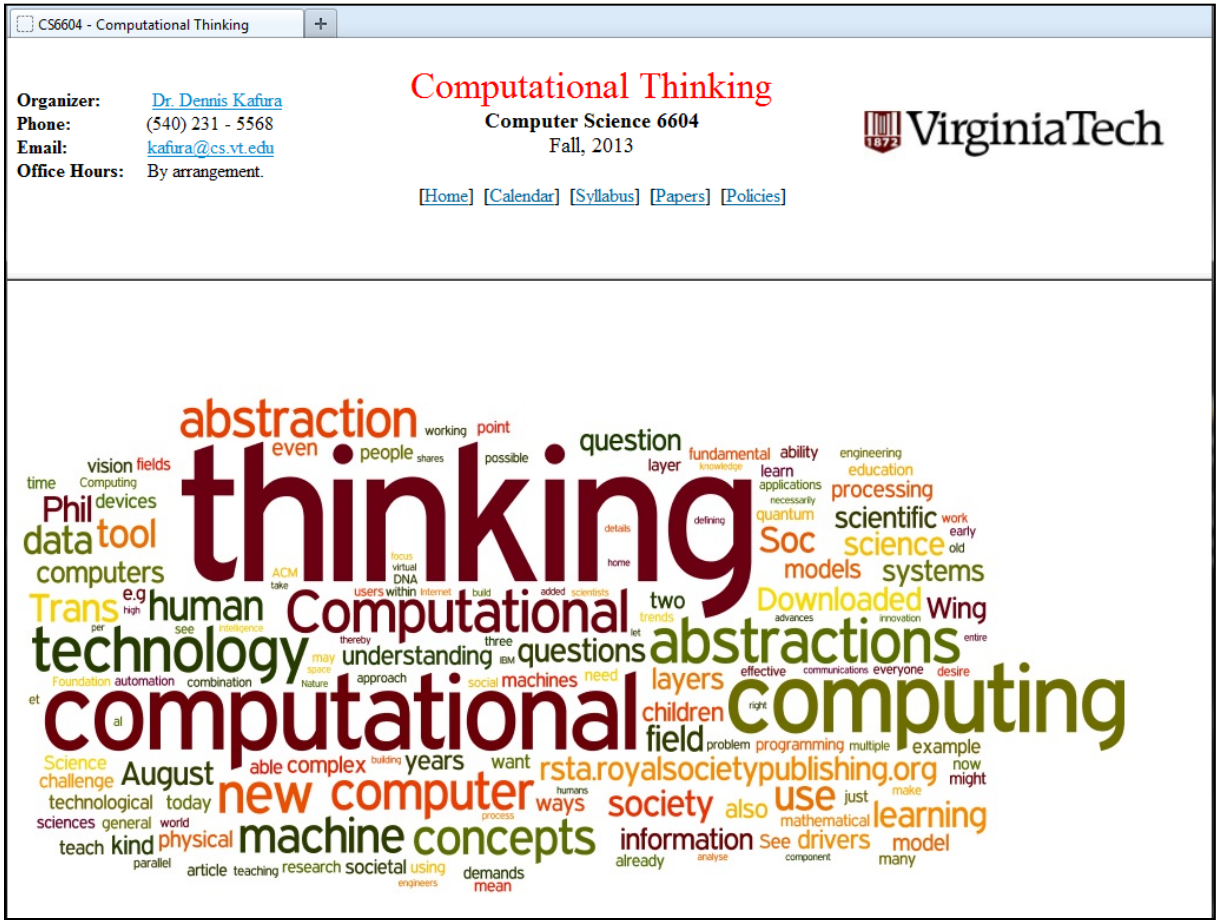
- Understand what “computational thinking” means
  - at a university level
  - specifically in the VT context
- Develop ideas
  - on courses and curriculum elements that provide a means of learning (some aspect of) computational thinking.
  - about how to measure/assess the extent to which a student has gained an ability to engage in computational thinking
- You will not learn anything “new” about computing but rather reflect on what you know and how you use that knowledge creatively

Organization

- Topic areas:
  - Model (What is computational thinking?)
  - Pedagogy (How can it be taught?)
  - Assessment (How can it be measured?)
- Requirements
  - Active participation in discussions
  - Presentation
  - Term paper
    - one section for each topic area
    - Intermediate due dates (TBA) for first two sections
    - Final version due on December 17, 2013

Materials

- www.cs.vt.edu/~kafura/CS6604



Discussion

- Who are you?
- Why are you interested in computational thinking?
- What do you think computational thinking is?
- What experiences have you had related to computational thinking?

Abstraction

- "Thinking like a computer scientist means more than being able to program a computer. It requires thinking at multiple levels of abstraction." [Wing 2006]
- "Mental" vs. "metal" [Wing 2008]
  - "And so the nuts and bolts in computational thinking are defining abstractions, working with multiple layers of abstraction and understanding the relationships among the different layers."
  - "We operate by mechanizing our abstractions, abstraction layers, and their relationships."

Abstraction

- "The abstraction process—deciding what details we need to highlight and what details we can ignore–underlies computational thinking." [Wing 2008]
- Computational thinking abstractions:
  - Extremely general: symbolic not just numeric
  - Have to worry about edge cases and failures
  - Defining the 'right' abstraction is critical
  - Helps manage complexity
    - By reducing aspects represented
    - By layering to
      - Separate concerns
      - Allow reasoning at different levels of abstraction

Reflections

- Is abstraction *the* defining mode of computational thinking?
  - If so, what do we make of abstractions in other areas?
    - statistical models
    - paintings
    - maps
  - If not, what are the others?
  - Is it the "automatic processing" that distinguishes computational thinking abstractions?
- Is a focus on *information* (processed automatically) more fundamental?
- Is *symbolic* more basic than *information*?

Starting a framework

- Computational thinking is the cognitive ability necessary to engage in creative work using the automatable manipulation of information.
  - Cognitive – a fundamental mental ability, not just skill in tool use
  - Information+automation – what distinguishes computational thinking from other ways of thinking
- The ability is derived from sufficient mastery of a conceptual framework. The conceptual framework includes:
  - abstraction
  - …

References

- [Snow 2012] Snow, E., et al., Assessing Computational Thinking, in *NSF-CE21 Community Meeting*. 2012: Washington, D.C., USA.
- [Wing 2006] Wing, J.M., *Computational thinking.* Communication of the ACM, 2006. **49**(3): p. 33-35.
- [Wing 2008] Wing, J.M., *Computational Thinking and Thinking About Computation.* Philosophical Transactions of the Royal Society A, 2008. **366**(1881): p. 3717-3725.