Computational Thinking,
Computer Science, and Coding
Hmm ....

What is computational thinking?

http://www.picturescraze.commovies/1528despicable+me+minion.html
What is Computational Thinking?

• A way of thinking for *logically* and *methodically* solving problems
  – E.g., *purposeful*, *describable*, *replicable*

• Includes *skills* such as
  – Decomposition
  – Pattern Recognition
  – Abstraction
  – Generalization
  – Algorithm Design
  – Evaluation
DeComposition

• Breaking down a process into a set of smaller sub-processes to allow us to describe, understand, or execute the process better
  – Dividing a task into a sequence of subtasks
  – Identifying elements or parts of a complex system
Decomposition
Examples of Decomposition

• When we taste an unfamiliar dish and identify several ingredients based on the flavor, we are decomposing that dish into its individual ingredients.

• When we give someone directions to our house, we are decomposing the process of getting from one place to another (e.g., city, interstate, etc.).

• When we break a course project into several steps, we are decomposing the task into smaller, more manageable subtasks.

• In mathematics, we can decompose a number such as 256.37 as follows: $2 \times 10^2 + 5 \times 10^1 + 6 \times 10^0 + 3 \times 10^{-1} + 7 \times 10^{-2}$.
Pattern Recognition

• Noticing or identifying similarities or common differences that will help us make predictions or lead us to shortcuts
  – We look for *patterns* when we play games to decide when to do certain things
  – Based on experience, we develop shortcuts mapping problem characteristics to solution
Examples of Pattern Recognition

- We look for patterns when choosing a registrar when we checkout.
- Drivers look for patterns in traffic to decide whether and when to switch lanes.
- People look for patterns in stock prices to decide when to buy and sell.
- Scientists look for patterns in data to derive theories and models.
- We look for patterns and learn from them to avoid repeating the same mistake.
  - “Last time we did this, it was ... let’s try something different ...”
Abstraction

• Preserving information that is relevant in a context, and forgetting or suppressing information that is irrelevant in that context to solve a problem
  – We use abstraction to **organize** things:
    • A human is a mammal, a mammal is an animal, and so on
  – A “big picture” so we can reason without thinking about the details
  – Transfer learning or learning by analogy
Abstraction

Abstraction is a fundamental concept in computer science. It involves hiding the details of implementation and providing a simplified interface. Abstraction allows us to focus on the essential aspects of a problem without getting bogged down in the details.

In computer science, abstraction is used to create a level of abstraction between the programmer and the machine. This level of abstraction is achieved through the use of programming languages, which provide a high-level description of a problem, and compilers and interpreters, which translate this high-level description into machine code.

Abstraction is also used in software engineering to organize and manage complexity. By breaking down a problem into smaller, more manageable parts, abstraction helps to reduce the complexity of the overall system.

In summary, abstraction is a powerful tool that allows us to create more effective and efficient software systems.
Examples of Abstraction

• A world map is an *abstraction* of the earth in terms of longitude and latitude, helping us describe the location and geography of a place.

• A sign of an aisle in a store—e.g., Walmart—is an *abstraction* of the items available in that aisle.

• When we write a book report, we summarize and discuss only the theme or key aspects of the book, it is abstraction.

• *When we tell a story or describe a movie to our friends, why don’t we describe every single detail of the story or movie?*
Generalization

- Identifying common or shared characteristics between two domains or problems such that models or solutions of one could be adapted or applied to the other
  - Mammals are warm blooded, give live birth, have hair, and so on. An elephant is a mammal. Therefore, it is warm blooded, give live birth, have hair ...
  - Group project A successful because of good teamwork strategy. Apply same good teamwork strategy to group project B should work too.
  - Deals with trends, norms, outliers, scalability
Generalization

Figure 1 - Features for different pattern types
Source: research data
Examples of Generalization

• Facebook tries to recommend ads to users based on what they generalize from what our friends like
• Google search identifies popular keywords in different regions at different times and suggests those keywords (in autocomplete and also correction) using a generalization-like process
• Amazon.com and Netflix model and categorize their customers, use generalization—*inferencing*—to predict what their customers are interested in, and make recommendations accordingly
• *When we don’t have complete information, we resort to generalization to make decisions (sometimes incorrectly)*
  — Think about: biases, stereotypes, superstitions
Algorithm Design

• Developing a step-by-step strategy for solving a problem
  – An algorithm is a sequence of steps that solves a problem
    • Input $\rightarrow$ output
    • Effective
  – **Algorithmic thinking** involves both *creation* and *execution* of an algorithm
Algorithm Design
THE FRIENDSHIP ALGORITHM
DR. SHELDON COOPER, PH.D

PLACE PHONE CALL

HOME? YES

"WOULD YOU LIKE TO SHARE A MEAL?"

NO

LEAVE MESSAGE

WAIT FOR CALLBACK

WHAT IS THE RESPONSE? NO

"DO YOU ENJOY A HOT BEVERAGE?"

YES

DINE TOGETHER

BEGIN FRIENDSHIP!

N = 0

"RECREATIONAL ACTIVITY? TELL ME ONE OF YOUR INTERESTS?" NO

N > 0? YES

DO I SHARE THAT INTEREST?

NO

N = N + 1

"WHY DON'T WE DO THAT TOGETHER?"

YES

CASE: TEA

HAVE TEA

COFFEE

HAVE COFFEE

COCOA

HAVE COCOA

PARTAKE IN INTEREST
Examples of Algorithm Design

- When a cook writes a recipe for a dish, he or she is creating an algorithm that others can follow to replicate the dish.
- When your friend writes down the instruction to get to her house, he or she is specifying a sequence of steps—that is, an algorithm—for you to follow (See Google maps!)
- When a teacher gives a set of instructions to carry out an experiment, he or she is specifying an algorithm for you to follow to collect and analyze data.
- When you follow an installation manual to assemble a bookshelf, you are executing an algorithm.
Evaluation

• Checking to see whether a solution is good
  – Algorithm correctness
  – Requirements (meeting constraints, design principles, etc.)
  – Performance (usability, efficiency, speed, complexity, reliability, etc.)
Evaluation
How the customer explained it

How the Project Leader understood it

How the Analyst designed it

How the Programmer wrote it

How the Business Consultant described it

How the project was documented

What operations were installed

How the customer was billed

How it was supported

What the customer really needed

Examples of Evaluation

• When we cook, we taste our dishes and then adjust flavoring accordingly
• When we fold a paper airplane, we test its flight, and then revise either the design or the “execution” to make it fly better
• When we jog or bike, we keep track of our breathing, joints, etc., and decide whether to stop, go slower, or go faster
• When we carry out a physics experiment, say, to find the relationship between temperature and pressure, we check our data, investigate why it does not match the theory, redo our experimental setup, and recollect data points ...
So, Computational Thinking is...

- Decomposition
- Pattern Recognition
- Abstraction
- Generalization
- Algorithm Design
- Evaluation

Wait a minute ...
These are things that we already are capable of doing!!!

http://www.wallpapersax.com/wallpaper/cartoons-despicable-me-minion.html
What is Computer Science?

- There are many definitions.

**Computer Science** helps us practice our computational thinking better, faster, with larger and more complex problems ...
What is Computer Science?

**Science:** *We learn, model, and describe how humans think, make decisions, and solve problems ...*

**Engineering:** *We build computer solutions to support and automate human thinking, decision making, problem solving ...*
What is Computer Programming?

- A process that leads from an original formulation of a computing problem to executable programs.

“Everybody in this country should learn how to program a computer...
Many CS & Computational Thinking education/outreach resources available online

- National Center for Women & Information Technology (NCWIT) “in-a-box” kits http://ncwit.org
- Ensemble, a Portal for Computing Educators http://www.computingportal.org
- CS Education Week http://www.csedweek.org
- Google’s Computer Science for High School (CS4HS) http://cs4hs.com
- Code.org http://code.org
- ...
“CS Unplugged is a collection of free learning activities that teach Computer Science through engaging games and puzzles that use cards, string, crayons and lots of running around.”

“The activities introduce students to underlying concepts such as binary numbers, algorithms and data compression, separated from the distractions and technical details we usually see with computers.”

“CS Unplugged is suitable for people of all ages, from elementary school to seniors, and from many countries and backgrounds. Unplugged has been used around the world for over fifteen years, in classrooms, science centers, homes, and even for holiday events in a park!”

http://csunplugged.org
CS Unplugged
http://csunplugged.org

Created by Tim Bell, Ian H. Witten, and Mike Fellows, and illustrated by Matt Powell
CS Unplugged
As NCWIT’s Computer Science-in-a-Box: Unplug Your Curriculum
(http://www.ncwit.org/resources/computer-science-box-unplug-your-curriculum)

• Examples
  – Magic card show – error detection
  – Treasure island – finite state machines
  – Sorting game – sorting networks
“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.” — Jeannette Wing, CACM 2006
Finally ... Computational Thinking is ... 

- Conceptualizing, not programming
  - Computer Science is *not just* computer programming
- Fundamental, not rote skill
  - A skill needed by everyone to function
- A way that humans, not computers, think
  - Humans are clever and creative; computers take instructions from humans
- Ideas, not artifacts
  - Ideas give birth to artifacts
- It’s for everyone
  - *Not just* for computer scientist; but for everyone
Special Bond: Computational Thinking and CS

• Articulation of computational thinking skills and processes into reusable computer programs (e.g., instructing machines to do pattern recognition) via coding makes us more aware and attentive of computational thinking.

• ... and more efficient and effective in practicing computational thinking in learning, problem solving, etc.
Flavors of Computational Thinking ...

Box 1.
Computer Science Teachers Association’s Concepts of Computational Thinking:

- Formulating problems for computational solution
- Logically organizing and analyzing data
- Abstractions including models and simulations
- Algorithmic thinking
- Evaluation for efficiency and correctness
- Generalizing and transferring to other domains
- Supported by: dispositions of confidence in dealing with complexity, persistence with difficult problems, tolerance for ambiguity, open-ended problems, communication and collaboration

Box 2.
Computing at School’s Concepts of Computational Thinking:

- Logical reasoning
- Algorithmic thinking
- Decomposition
- Generalization
- Patterns
- Abstraction
- Representation
- Evaluation
- Supported by: techniques of reflecting, coding, designing, analyzing, and applying

Box 3.
ISTE’s Standards for Students in Computational Thinking:

- Leverage the power of technological methods to develop and test solutions
- Collect data
- Analyze data
- Represent data
- Decomposition
- Abstraction
- Algorithms
- Automation
- Testing
- Parallelization
- Simulation
- Supported by: empowered learner, digital citizen, knowledge constructor, designer, communicator, collaborator

References

- http://www.google.com/edu/computational-thinking
- Paul Curzon’s “So What is Computational Thinking”
“It's an energy field created by all living things. It surrounds us and penetrates us; it binds the galaxy together.”

Obi-Wan Kenobi introduced Luke Skywalker and most of us to the Force for the first time in Star Wars: A New Hope
“It's a skill practiced by all living things. It surrounds us and penetrates us; it binds problem solving (and learning, discovery, ...) together.”

Leen-Kiat Soh on the fundamentality of Computational Thinking.
“May computational thinking be with you.”