Day 1:

* What is computer science?
  + Based on math, logic, philosophy, engineering
  + Computers used to be humans who computed things by hand (logs, sqrts, sin, cos, etc.)
  + Today it affects many areas:
    - Weather forecasting
    - Art, Music
    - Commerce (online shopping)
    - Science, medicine
    - Security
    - Question for students: what computational problems brought you here? What computational problems are you interested in?
  + How do we automate these things?
    - Alan Turing pioneered this
      * Developed the idea of a Turing machine:
        + A tape with zeros and ones on it

Infinite length

Can move backwards and forwards

* + - * + Something to manipulate the tape

Erase values, write values

* + - * + Something to view the tape

Decides what to do next (move the tape, erase, write, etc.)

* + - * + This simple machine can compute anything that is computable (!!!!!)
        + Great video: <https://www.youtube.com/watch?v=E3keLeMwfHY>
    - Nowadays we use laptops, cell phones, etc.
      * Show how far computing has become (Ron’s video: thirty years of computing)
  + How do we do these things efficiently?
    - Forecasting today’s weather is useless if we don’t get the information until tomorrow
    - We want our programs to be correct, but we also want them to be fast
    - What makes a program fast or slow?
      * Typically: the number of operations we need to perform
* How do people interact with computers?
  + Non programmers
    - Voice
      * This was a really hard problem to solve!
    - Keyboard / mouse (input)
    - Monitor / speakers (output)
  + Programmers:
    - Code!
    - So what is code? Consider:
      * Humans speak human languages
      * Computers speak binary (ones and zeros)
        + This is because computers are electronic

One means something is on

Zero means something is off

* + - * If humans want to be able to communicate with the computer, we need to have a way of turning human language into binary
  + Enter programming languages
    - Might be good to show a simple code example here
    - Not binary, but not really human language either - its somewhere in between
      * Tries to bridge the gap between how humans think and how computers think
  + So how do computers think?
    - Humans:
      * Are very complex
      * Are emotional
      * Are rational (mostly)
      * Can say the same words two different ways, but mean something different
        + Examples:

The children made delicious snacks

A panda eats shoots and leaves

* + - Computers
      * Are also very complex
      * Are not emotional
      * Are extremely literal
        + Take everything we tell them at face value - no room for interpretation whatsoever
    - We as humans must strive to think like a computer if we want to interact with them effectively
      * No ambiguous statement
      * No assumptions about what the computer knows
      * Must specify everything exactly, step by step
  + What is a computer program?
    - A series of instructions that a computer will perform
      * One instruction at a time
        + But very, very fast from the human’s perspective
      * We can reuse important sets of instructions
        + This is the idea of abstraction
      * We can repeat instructions
      * We can make decisions
      * We can track values without necessarily knowing what the exact value is (variables)
    - A series of instructions used to solve a particular generic form of a problem is called an algorithm
      * The generic part is important!
        + Write once, use many times
      * Algorithms are often expressed as flow charts
        + See “Algorithm” wikipedia page for a flow chart example
      * Can be thought of as a recipe for the computer
      * It must also be fast!
        + Consider: I can stir batter by hand or with a mixer - the mixer will get the job done much faster

But sometimes stirring by hand might be the right way to do it!

Correctness always trumps speed

* Introduction to pseudocode
  + Introduce a problem
    - Someone purchases something worth $x and pays for it
    - We must give them change
    - Do not have them write pseudocode for this – just write the solution in English
  + While working on this:
    - Assume $x could be anything under $20
    - We want to give back as few bills and coins as possible
    - Be as exact as possible!
  + Go through solution
    - Point out repetition, decisions, variables
  + Convert solution to pseudocode
  + Note the following:
    - Does a computer know what a quarter is?
    - Does it matter if it knows what a quarter is?
  + With humans we can assume a basic level of knowledge - we cannot assume that a computer will know anything!
  + Next time we will see a lot more examples of pseudocode