Complexity

Recap

- The run-time of code is usually greater when operating on a larger input.
- There can be massive differences in run-time between different algorithms, even for the same problem.
- Knowing the shape of the run-time growth curve is very valuable.
  - "Can I wait long enough to get the answer for this input?"
  - "Is it even worth writing this code?"

Ways to determine run-time

- Run the code and measure run-time.
  Run it for different input sizes to determine the growth curve.
- Estimate run-time "on paper."
  Why would you want to do that?

Estimating run-time

- Benefits of estimating on paper:
  - Can pick among alternative algorithms without having to code them all!
  - Can decide if the one we pick is even worth coding up.
- If all we want to know is the shape of the growth curve, a back-of-the-envelope estimate is fine.

- Technique: pick a representative operation, and and calculate how many times it will occur.
- What makes an operation "representative"?
- Let's try analyzing some example functions.

General strategy

- Two chunks of code in sequence:
  Add the time they take.
- Loop:
  Multiply (the time one iteration takes) times (the number of iterations).
- If-elif statement:
  One approach is to determine the worst-case scenario by taking the maximum amount of time of any of the branches.

Computer Science

- Computer science is about much more than programming. Some themes:
- Recognizing commonalities among many problems and creating a general solution.
- Knowing general strategies and recognizing that a problem can be solved using one of them.
- Analyzing and comparing alternative solutions for efficiency -- in time used or space used.