Some Sum Formulas

- Please commit the following to memory (p. 76)
  - Sum of a geometric series: $\sum_{k=0}^{n} ar^k = \frac{ar^{k+1} - a}{r - 1}$
  - Sum of the counting numbers: $\sum_{k=1}^{n} k = \frac{n(n+1)}{2}$
  - Sum of the squares: $\sum_{k=1}^{n} k^2 = \frac{n(n+1)(2n+1)}{6}$

Your Turn Again

Evaluate: $\sum_{i=1}^{30} (i^2 + 3i)$

Evaluate: $\sum_{c=21}^{40} c^2$

Evaluate: $\sum_{4 \le u \le 9} 3^u$

Problems

- Not problems:
  - What is 338 + 1993?
  - Is 21789347189234817239481237481 prime?
  - Who wins chess?
- Problems:
  - Given two integers in decimal form, how do you add them?
  - Given an integer, how do you determine if it is prime?
  - Given a game with deterministic moves, how do you determine the winner?
- For us, "problem" will have this more general meaning. We will typically give our problems names with all capitals: ADD, PRIME, GAME.

Algorithm

- An algorithm is a finite sequence of steps which will solve a given problem
  - Each step must be well-defined and deterministic
  - Each step should take a finite amount of time
  - The algorithm must terminate after a finite number of steps (in a finite amount of time)

A non-Algorithm

- Problem:
  - PRIME: determine if a given input integer is prime or composite
- Algorithm:
  - Let *n* be the input integer
  - Write down all divisors of *n*
  - Check to see if there are any divisors other than 1 or *n* on the list
  - If there are, then the number is composite, otherwise it is prime
- Not an algorithm because step 2 is not well-defined and deterministic

Another non-Algorithm

- Problem:
  - MAX-ELEMENT: Find the maximum element in a finite set of integers
- Algorithm:
  - Sort the elements of the set into increasing order
  - Put this sorted list into a tuple
  - The last element of the tuple is the maximum element
- This is not an algorithm because the first step is vague and unspecified

Another non-Algorithm

- Problem:
  - CARDINALITY: Given a set of natural numbers, determine the cardinality of that set
- Algorithm:
  - 1. Let S be the set
  - 2. Let $n = 1$ and $count = 0$
  - 3. If $n \in S$, then add 1 to *count*
  - 4. Increment *n* and go back to step 3
  - 5. If *count* reaches a maximum value N and then never increases above that value, then $|S| = N$, otherwise S is countably infinite
- This is not an algorithm because it is not guaranteed to terminate in a finite number of steps