- PS2? Wrapping up — will be done by the weekend
- TT2? Not sure... not close right now, but TAs are working hard — they know how important this is... Updates will be posted on Piazza.
- PS1, TT1 remarking? Working on it. Not sure yet what will be possible.

Algorithm analysis — complexity / runtime analysis

Goal: Given algorithm, find a simple function $f: \mathbb{N} \to \mathbb{R}^{\geq 0}$ such that # steps executed by the algorithm, as a function of input size, is in $\Theta(f)$.

- Input size?
  - "standard size" = total number of bits to represent entire input in memory
  - in practice, we use following conventions
  - individual integers take constant size (not always actually true!
    $x = \text{\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_ \_}$)
  - size of strings = # characters
  - size of list = sum of sizes of individual elements

    ↳ generally, $\cong$ length of list

- # steps?
  - 1 "step" = <u>any</u> block of code that executes together and whose runtime does <u>not</u> depend on input size
  - THERE IS NO ABSOLUTE MEASURE of "1 STEP"
  - what's NOT constant?
    - loops
    - function calls } CSC165
    - recursion — CSC236
    - complex data structures — CSC263, a bit in CSC148

Example:

```
def f(n: int) -> int          # Assume n ≥ 0
    r = 0
    for i in range(10):       # Loop 1
        for j in range(n * n):   # Loop 2
            r = r + j
    for i in range(n // 2):   # Loop 3
        for j in range(i * i):   # Loop 4
            r = r - j
    return r
```

Runtime, as a function of n?

WAIT! Input size?

For the purpose of practicing analysis

treat algorithms with a single integer input differently — express runtime as a function of input VALUE (instead of its size).

- Lines 1 & 8: 1 step (always execute together)
- for loops, work inside-out

step = time

- loop 2: - body takes 1 step
  - n^2 iterations
  - total time = $\underbrace{1+1+\dots+1}_{n^2} = n^2$ steps
- loop 1: - body takes n^2 steps
  - 10 iterations
  - total time = $\underbrace{n^2+n^2+\dots+n^2}_{10} = 10n^2$

- loop 4: - body = 1 step
  - # iterations = i^2 } i^2 steps

- loop 3: - body = i^2 steps
  - # iterations = $\lfloor \frac{n}{2} \rfloor$

steps changes from one iteration to the next...

$$\text{total} = \sum_{i=0}^{n//2-1} i^2 = \underbrace{0^2 + 1^2 + \ldots + \left(\lfloor \tfrac{n}{2} \rfloor - 1\right)^2}_{\Theta(n^3)}$$

$$\boxed{\sum_{i=0}^{m-1} i^2 = \frac{(m-1)\,m\,(2m-1)}{6}}$$

- Overall: $10n^2 + \Theta(n^3) \in \boxed{\Theta(n^3)}$