1 Example from last time

Let $f, g, \mathbb{N} \to \mathbb{R}^{\geq 0}$
Prove that if $f \in \mathcal{O}g$ then, $g \in \mathcal{O}(f)$

Proof:

Assume $f \in \mathcal{O}g$
$\exists c, n_0 \in \mathbb{R}^+, \forall n \in \mathbb{N}, n \geq n_0 \Rightarrow f(n) \leq cg(n)$

Let $n_0' = n_0 \in \mathbb{R}^+$
Let $c' = \frac{1}{c} \in \mathbb{R}^+$ since $c \in \mathbb{R}^+$
Let $n \in \mathbb{N}$ Assume $n \geq n_0' = n_0$

Then

$$f(n) \leq cg(n)$$
$$\frac{1}{c}f(n) \leq g(n)$$
$$g(n) \geq c'f(n)$$

Therefore,

$$\exists c', n + 0' \in \mathbb{R}^+, \forall n \in \mathbb{N}, n \geq n_0' \Rightarrow g(n) \geq c'f(n)$$

■

2 Algorithm Analysis

We have the math tools of: $\mathcal{O}, \Omega, \Theta$
We use these tools to achieve the goal of a "simple" f so that $\Theta(f)$ "represents the particular resource complexity.

We will focus on runtime complexity:
For Algorithm A and input x, let $RT_A(x)$ b the time to run algorithm A on input x

Say we have sorting algorithm: `def sort(l):` $RT_{sort} : Lists \to \mathbb{R}^{\geq 0}$

- Given the definition of $\mathcal{O}, \Omega, \Theta$, they cannot take lists as input
- We therefore take the measure the input. ie, length of the list - `len(l)` $\in \mathbb{N}$
- We also must output the list in some way.
  - Worst case - WC `sort(n)` = max
  - Best case - BC `sort(n)` = min

- Average Case - AC `sort(n)` = average

Define Today: $x \in \mathbb{N}$

```
1  def A(n: int) -> int: #Assume n >= 0
2      r = 0
3      for i in range(10):
4          for j in range(n * n):
5              r = r + j
6      for i in range (n//2):
7          for j in range(i * i):
8              r = r - j
9      return r
10
```

Determine $RT_A(n)$

note a step is a piece of code tht takes fixed/consistent amoutn of time to interpredtate the input. Then m steps take between $min(c)m$ and $max(c)m$ time, ie $\theta(m)$

We analyze the above program:

- Loop lines 3-4
  - Body: 1 step
  - Iterations: n^2
  - Total: $n^2 \cdot 1 = n^2$ steps
- Loop lines 2-4
  - Body: n^2 step
  - Iterations: 10
  - Total: $10n^2$ steps
- Loop lines 6-7
  - Body: i^2 steps
- Loop lines 5-7
  - Body: i^2 step
  - Iterations: $0, 1, 2, \ldots, \lfloor \frac{n}{2} \rfloor - 1$
  - Total: $0^2 + 1^2 + 2^2 + \cdots + \left(\lfloor \frac{n}{2} \rfloor - 1 \right)^2$ steps
    - $* = \sum_{i=0}^{\lfloor \frac{n}{2} \rfloor - 1} i^2 \in \theta(n^3)$

Therefore the algorithm $\in \theta(1 + 10n^2 + n^3 + 4) \in \theta(n^3)$