- lab schedule
- hw: crowdGrader
- program: submit: ext. 1 day

Array comparison by ins. sort
(worst case).

\[\#\text{comp} = 1 + 2 + 3 + \cdots + (n-1) = \frac{n(n-1)}{2}\]

\[= \Theta(n^2)\]
$$\sum \text{ sort:}$$

$$T(n) = n \lfloor g(n) \rfloor - n + 1$$

$$\text{Solve:}$$

$$T(n) = \begin{cases} 
0 & n = 1 \\
2T\left(\frac{n}{2}\right) + (n-1) & n \geq 2 
\end{cases}$$

assuming \( n \) is exact pow. of 2.

$$\text{check:}$$

$$\text{RHS} = 2T\left(\frac{n}{2}\right) + (n-1)$$

$$= 2 \left[ \frac{n}{2} \cdot \lfloor g(\frac{n}{2}) \rfloor - \frac{n}{2} + 1 \right] + (n-1)$$

$$= n \cdot \lfloor g(n) - \lfloor g(2) \rfloor \rfloor - n + 2 + n - 1$$

$$= n \lfloor g(n) - n \rfloor + 1 = T(n) = \text{LHS}$$
\[ T(n) = \Theta(n \lg(n)) \]

Fact: This is the asymptotic run time even without assumption that \( n = 2^k \).

Ins. sort: \( \Theta(n^2) \)

Merge sort: \( \Theta(n \lg(n)) \).
To analyze a recursive algorithm, we write a recurrence relation

\[ T(n) = \begin{cases} 
  c & 1 \leq n \leq n_0 \\
  aT\left(\frac{n}{b}\right) + D(n) + C(n) & n > n_0 
\end{cases} \]

Then seek asymptotic growth rate of \( T(n) \). (an asymptotic solution.)
Handout: asymptotic growth

notations: $O, \Omega, o, \omega$

Define let $f(n), g(n)$ be functions.

$O(g(n)) = \{ f(n) | \exists c > 0, \exists n_0 > 0, \forall n \geq n_0 : 0 \leq f(n) \leq cg(n) \}$

notation: write $f(n) = O(g(n))$ to mean $f(n) \in O(g(n))$. We say $g(n)$ is an asymptotic upper bound for $f(n)$.
Note: c and \( n_0 \) are not unique.

\[ 10n + 100 = \Theta(n^2 - 40n + 500) \]

Observe: \( 0 \leq 10n + 100 \leq n^2 - 40n + 500 \) for all \( n \geq 40 \) (check).

\( \therefore c = 1 \) and \( n_0 = 40 \)
In fact: \( a+b = O(cn^2 + \log n) \)

for any \( a \geq 0 \) with \( a \in \mathbb{R}^n \).

Discussion: asymptotically non-negative in the large \( n \).

\[ f(n) \equiv 0 \] for suffix large in.
In fact: if \( p(n) \), \( q(n) \) are polynomials, then

\[ p(n) = \Theta(q(n)) \quad \text{if} \quad \deg(p(n)) \leq \deg(q(n)) \]

\[ \Omega(g(n)) = \{ f(n) \mid \exists c > 0, \exists n_0 > 0, \forall n \geq n_0 : 0 \leq c g(n) \leq f(n) \} \]

Write: \( f(n) = \Omega(g(n)) \)

Say: \( g(n) \) is an asymptotic lower bound for \( f(n) \).
Fact: for polynomials $p(n), q(n)$

$$p(n) = \Omega(q(n)) \iff \deg(p(n)) \geq \deg(q(n))$$

**Picture:**

![Graph showing the relationship between $f(n)$ and $g(n)$ with a breakpoint at $n_0$.]

**Theorem**

$$f(n) = O(g(n)) \iff g(n) = \Omega(f(n))$$
Proof. Leave $(\leq)$ as exercise.

$(\Rightarrow)$

Assume $f(n) = O(g(n))$. Then there exist positive $c_1, n_1$ such that

$$\forall n \geq n_1 : 0 \leq f(n) \leq c_1 \cdot g(n). \quad \ast$$

We must show that $g(n) = \Omega(f(n))$, i.e. there exist positive $c_2, n_2$ such that

$$\forall n \geq n_2 : 0 \leq c_2 \cdot f(n) \leq g(n). \quad \ast\ast$$

Let $c_2 = \frac{1}{c_1}$ and $n_2 = n_1$.

Then obviously $\ast \Rightarrow \ast\ast$
\[ \Omega(g(n)) = \Theta(g(n)) \cap \Omega(g(n)) \]

Equivalently:

\[ f(n) = \Theta(g(n)) \text{ if and only if } \exists c_1 > 0, \exists c_2 > 0, \exists n_0 > 0, \forall n \geq n_0 : 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \]

Picture:
we say: \( g(n) \) is a tight asymptotic bound for \( f(n) \). Also: \( f(n) \) and \( g(n) \) are asymptotically equivalent.

**Exercise:**

\[ f(n) = \Theta(g(n)) \text{ iff } g(n) = \Theta(f(n)) \]

**Exercise:**

Let \( c > 0 \). Prove that

1. \( c \cdot g(n) = \Theta(g(n)) \)
2. \( c \cdot g(n) = \Omega(g(n)) \)
3. \( c \cdot g(n) = \Theta(g(n)) \)
Ex. Prove $\sqrt{n+10} = \Theta(\sqrt{n})$.

Proof: We must find pos. numbers $c_1, c_2, n_0$ st. $\forall n \geq n_0$

$$0 \leq c_1 \sqrt{n} \leq \sqrt{n+10} \leq c_2 \sqrt{n}$$

Let $c_1 = 1$, $c_2 = \sqrt{2}$, $n_0 = 10$. 