General Information

- Office Hours: TTh 10:45 – 11:45 and T 3:20 – 4:20
- Prerequisites/background: Linear algebra, Data structures and algorithms, and Discrete math
Use of Blackboard

- Announcements
- Problem sets (aka assignments or homework)
- Solution sets
- Lecture notes
- Grades
- Check at least weekly
Lecture Basics

- Lectures: Slides + board (mostly)
- Lecture slides ⊂ lecture notes posted on BB
- Not all taught in class are in the lecture notes/slides, e.g., some example problems. So take your own notes in class
Course Organization

- Mathematical foundation
- Methods of analyzing algorithms
- Methods of designing algorithms
- Additional topics chosen from the lower bound theory, amortization, probabilistic analysis, competitive analysis, NP-completeness, and approximation algorithms
Grading

▶ About 12 problem sets: 60%
▶ Final: 40% (in-class)

Grading Policy (may be curved)

▶ [90, 100]: A or A-
▶ [80, 90): B+, B, or B-
▶ [70, 80): C+, C, or C-
▶ [60, 70): D+, D, or D-
▶ [0, 60): F
Homework Submission Policy

- Hardcopy (stapled and typeset in LaTeX) to be submitted at the beginning of class on the due date
- Extensions may be permitted for illness, family emergency, and travel to interviews/conferences. Requests must be made prior to the due date
Homework Policy

- Homework must be typeset in LaTeX. Nothing should be handwritten, including diagrams and tables.
- Empty-hand policy when you discuss homework problems with your classmates.
- List your collaborators for every homework problem.
- Cite all references that help you solve a problem.
- In no case should you copy verbatim from other people's work without proper attribution, as this is considered plagiarism.
Honor Code

- "As a member of the William and Mary community, I pledge on my honor not to lie, or steal, either in my academic or personal life. I understand that such acts violates the Honor Code and undermine the community of trust, of which we are all stewards."

- Academic honesty, the cornerstone of teaching and learning, lays the foundation for lifelong integrity. The Honor Code is, as always, in effect in this course. Please go to the "Honor System" page in the wm.edu site to read the complete honor code document.
Common functions

▶ Monotonicity: Definitions of monotonically increasing/decreasing or strictly increasing/decreasing functions.

Important note: In this course, since functions are used to represent time complexity, we restrict our attention to only increasing functions that map positive number(s) to positive number.

▶ Ceilings and floors: \( \lceil x \rceil \) and \( \lfloor x \rfloor \), where \( x \) can be any real number.

\[ x - 1 < \lfloor x \rfloor \leq x \leq \lceil x \rceil < x + 1, \ \lceil n/2 \rceil + \lfloor n/2 \rfloor = n. \]

▶ Modular arithmetic: \( a \mod n = a - \lfloor a/n \rfloor n. \ a \equiv b \mod n \) iff \( a \mod n = b \mod n. \)
Polynomials: \( p(n) = \sum_{i=0}^{d} a_i n^i \). (Note: Coefficients \( a_i \) and degree \( d \) are constants.)

Exponentials: \( a^0 = 1, \ a^{-1} = \frac{1}{a}, \ a^m \cdot a^n = a^{m+n}, \ a^m / a^n = a^{m-n} \).
\( e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots = \sum_{i=0}^{\infty} \frac{x^i}{i!} \). (Note: \( e = 2.71828 \ldots \))
\( \lim_{n \to \infty} (1 + \frac{x}{n})^n = e^x \).
Logarithms: \( \log n = \log_2 n \) or \( \log_c n \) for some \( c \) we don’t care about.

\[
\log(ab) = \log a + \log b, \quad \log\left(\frac{a}{b}\right) = \log a - \log b. \\
\log_a b = \frac{\log_c b}{\log_c a}.
\]

\[
\log_a b^n = n \log_a b \neq (\log a b)^n, \quad a^{\log_a n} = n, \quad a^{\log_c b} = b^{\log_c a}.
\]

\[
\ln(1 + x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \cdots.
\]
Factorials: \( n! = n \cdot (n-1) \cdots 2 \cdot 1. \)

\( n! = n \cdot (n-1)!, \) 0! = 1. (Recursive definition)

Sterling’s approximation: \( n! = \sqrt{2\pi n} \left( \frac{n}{e} \right)^n (1 + \Theta(\frac{1}{n})) \). (Note: \( \Theta \) means having the same order of magnitude.)

The following approximation also holds: \( n! = \sqrt{2\pi n} \left( \frac{n}{e} \right)^n e^{\alpha_n} \), where \( \frac{1}{12n+1} < \alpha_n < \frac{1}{12n} \).

\( \log n! = \Theta(n \log n). \)
Functional iteration: A function $f$ applied iteratively $i$ times to an initial argument $n$. Defined recursively, $f^{(0)}(n) = n$ and $f^{(i)}(n) = f(f^{(i-1)}(n))$ for $i > 0$. (Note: The distinction between $f^{(i)}(n)$ and $f^i(n)$.) For example, if $f(n) = 2n$ then $f^{(i)}(n) = 2^i n$.

The log star function: $\log^* n = \min \{ i \geq 0 : \log^{(i)} n \leq 1 \}$, which is a very slowly growing function. $\log^* 2 = 1$, $\log^* 4 = 2$, $\log^* 16 = 3$, $\log^* 65536 = 4$, $\log^* 2^{65536} = 5$. 
Fibonacci numbers: $F_0 = 0, F_1 = 1, F_i = F_{i-1} + F_{i-2}$ for $i \geq 2$.

$F_i = \frac{\phi^i - \hat{\phi}^i}{\sqrt{5}}$, where $\phi = \frac{1+\sqrt{5}}{2} = 1.61803\ldots$ is called the golden ratio, $\hat{\phi} = \frac{1-\sqrt{5}}{2} = -0.61803\ldots$ is the conjugate of $\phi$, and both are roots of equation $x^2 = x + 1$. 

Asymptotic notation

- Used to compare the growth rate or order of magnitude of increasing functions. “Asymptotic” describes the behavior of functions in the limit, for sufficiently large values of variables.

- $f(n) = O(g(n))$ if $\exists c, n_0$ such that $f(n) \leq cg(n)$ for $n \geq n_0$.

- $f(n) = \Omega(g(n))$ if $\exists c, n_0$ such that $f(n) \geq cg(n)$ for $n \geq n_0$.

- $f(n) = \Theta(g(n))$ if $\exists c_1, c_2, n_0$ such that $c_1 g(n) \leq f(n) \leq c_2 g(n)$ for $n \geq n_0$.

- $f(n) = o(g(n))$ if $\forall c \exists n_0$ such that $f(n) < cg(n)$ for $n \geq n_0$.

- $f(n) = \omega(g(n))$ if $\forall c \exists n_0$ such that $f(n) > cg(n)$ for $n \geq n_0$. 
Remarks:

In CLRS, the above notation is defined as sets of functions. For example, \( f(n) \in O(g(n)) \).

Comparison of growth rates of two functions: \( O(\leq), \Omega(\geq), \Theta(=), o(<), \omega(>) \).

\( f(n) = O(g(n)) \) iff \( g(n) = \Omega(f(n)) \), and \( f(n) = o(g(n)) \) iff \( g(n) = \omega(f(n)) \).

\( f(n) = \Theta(g(n)) \) iff \( f(n) = O(g(n)) \) and \( f(n) = \Omega(g(n)) \).

\( f(n) = O(g(n)) \) if \( f(n) = o(g(n)) \), and \( f(n) = \Omega(g(n)) \) if \( f(n) = \omega(g(n)) \).

An alternative definition for \( f(n) = o(g(n)) \) is \( \lim_{n \to \infty} \frac{f(n)}{g(n)} = 0 \). Likewise, an alternative definition for \( f(n) = \omega(g(n)) \) is \( \lim_{n \to \infty} \frac{f(n)}{g(n)} = \infty \).
More remarks:

- Asymptotic notation ignores constant factors and lower-order terms.
- Rule of thumb: constant ≤ polylogarithmic ≤ polynomial ≤ exponential ≤ superexponential.

Example: 1, $\sqrt{\log n}$, $\ln n$, $(\log n)^2$, $\sqrt{n}$, $\sqrt{n \log n}$, $n$, $n \log n$, $n^2$, $n^{\log \log n}$, $2^n$, $n 2^n$, $n!$, $2^{2^n}$. 
Summations/Series

- Property of linearity: $\sum_{i=1}^{n}(ca_i + b_i) = c\sum_{i=1}^{n}a_i + \sum_{i=1}^{n}b_i$ and $\sum_{i=1}^{n}\Theta(f(i)) = \Theta(\sum_{i=1}^{n}f(k))$.

- Arithmetic sum/series: $\sum_{i=1}^{n}i = 1 + 2 + \cdots + n = \frac{1}{2}n(n+1)$.

- Geometric sum/series:
  $\sum_{i=0}^{n}r^i = 1 + r + r^2 + \cdots + r^n = \frac{r^{n+1} - 1}{r - 1}$ for $r \neq 1$.
  $1 + r + r^2 + \cdots = \frac{1}{1-r}$ for $|r| < 1$.

- Harmonic series: $H_n = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} = \ln n + \gamma + \frac{\varepsilon}{2n}$ for $\gamma = 0.5772156649\ldots$ (Euler’s constant) and $0 < \varepsilon < 1$.

  Example: Prove that $\ln(n+1) < H_n < \ln n + 1$.
  (Approximation by integrals)
Binomial series: \( \binom{n}{0} + \binom{n}{1} + \binom{n}{2} + \cdots + \binom{n}{n} = 2^n. \)

Other useful sums:

\[
\sum_{i=1}^{n} i^2 = \frac{1}{6} n(n+1)(2n+1). \text{ (A direct proof starting with } \sum_{j=1}^{i} (2j - 1) = i^2) \\
\sum_{i=1}^{n} i^3 = (\sum_{i=1}^{n} i)^2. \text{ (Proved by induction)} \\
\sum_{i=1}^{n} ix^{i-1} = \frac{nx^{n+1}-(n+1)x^n+1}{(x-1)^2}. \text{ (Proved by using derivatives)}
\]
Proof techniques

- Proving by contradiction:
  The following three statements are logically equivalent:

  1. If $A$ then $B$.
  2. If not $B$ then not $A$.
  3. If $A$ and not $B$ then not $C$, where $C$ is a proved fact or axiom.

*Example*: Use contradiction to prove that
(a) There are infinitely many prime numbers and
(b) $\sqrt{2}$ is irrational.
Proving by induction:

The following statements are mathematically equivalent:

1. \( P(n) \) for integers \( n \geq c \).
2. Simple integer induction: \( P(c) \) and \( P(n - 1) \rightarrow P(n) \). (What are inductive basis, inductive hypothesis, and inductive step?)
3. General integer induction: \( P(c) \) and 
   \[
   (\forall i : c \leq i \leq n - 1) P(i) \rightarrow P(n).
   \]

Example: Use induction to prove that
(a) \( \sum_{i=1}^{n} i^3 = (\sum_{i=1}^{n} i)^2 \) and
(b) Every positive composite integer can be expressed as a product of prime numbers.
Solving recurrences

- Recurrence is an equation or inequality that defines a function in terms of the function’s values on smaller inputs. For example, $T(1) = \Theta(1)$ (boundary condition) and $T(n) = 2T(n/2) + \Theta(n)$ for $n \geq 2$ (recurrence) or almost equivalently, $T(1) = 1$ and $T(n) = 2T(n/2) + n$ for $n \geq 2$.

- Remark: We may neglect some technical details due to our interest in asymptotic solutions:
  - Relax the integer argument requirement on functions. For example, use $T(n/2)$ instead of $T(\lfloor n/2 \rfloor)$ or $T(\lceil n/2 \rceil)$.
  - Assume boundary condition $T(n) = \Theta(1)$ for small $n$ if not given explicitly. Asymptotically, $\Theta(1)$ is the same as any constant $c$ no matter how large it is.
  - Use $\Theta(f(n))$ or $f(n)$ at will in the recursive definition since this will have no affect on the final answer when expressed in $\Theta$. 
(1) The iteration method: Apply recurrence until a summation pattern can be figured out.

*Example:* \( T(n) = 3T\left(\frac{n}{4}\right) + n \). (Assume \( n = 4^k \).)

*Example:* Solve \( T(n) = \sqrt{n}T(\sqrt{n}) + n \) by iteration.
(2) The recursion-tree method: Similar to the iteration method, use a tree for bookkeeping. Suitable for solving recurrence in big-O, where the function appears more than once on the right-hand-side of the recursive equation.

Example: \( T(n) = T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) + n. \)

Example: Solve \( T(n) = T(\alpha n) + T((1 - \alpha)n) + n, \) where \( 0 < \alpha < 1, \) by recursion tree.
The master method:

**Theorem:** If \( T(n) = aT\left(\frac{n}{b}\right) + f(n) \) for \( a \geq 1 \) and \( b > 1 \), then

(a) if \( f(n) = O(n^{(\log_b a) - \varepsilon}) \) for some \( \varepsilon > 0 \), then
\[
T(n) = \Theta(n^{\log_b a});
\]
(b) if \( f(n) = \Theta(n^{\log_b a}) \), then \( T(n) = \Theta(n^{\log_b a} \log n) \);
(c) if \( f(n) = \Omega(n^{(\log_b a) + \varepsilon}) \) for \( \varepsilon > 0 \) and if \( af\left(\frac{n}{b}\right) \leq cf(n) \) for \( c < 1 \) and all large \( n \), then \( T(n) = \Theta(f(n)) \).

Remark: The master method does not cover all cases.

**Example:** \( T(n) = 3T\left(\frac{n}{4}\right) + n \log n \). (\( a = 3 \), \( b = 4 \), and \( f(n) = n \log n \). Case (c) applies.)

**Example:** Solve \( T(n) = 4T\left(\frac{n}{2}\right) + f(n) \) by the master theorem for \( f(n) = n, n^2, n^3 \).

**Example:** \( T(n) = 2T\left(\frac{n}{2}\right) + n \log n \). (The master theorem does not work.)
(4) The substitution method: Guess and verify.

Example: Let \( T(n) \leq cn \) for \( n \leq 49 \) and
\[
T(n) \leq T\left(\frac{n}{5}\right) + T\left(\frac{3n}{4}\right) + cn \quad \text{for} \quad n \geq 50.
\]
(Guess \( T(n) \leq 20cn \) and then prove by induction. Can the recursion tree method be used?)
Remarks for the substitution method:

- Making a good guess.

- To prove $T(n) = O(f(n))$, sometimes we use an inequality stronger than $T(n) \leq cf(n)$ in the induction, such as $T(n) \leq 20cf(n)$ in the earlier example or $T(n) \leq cf(n) - d$ which can be used for solving $T(n) = 2T\left(\frac{n}{2}\right) + 1$.

- Avoid using asymptotic notation in the inductive proof.

Example: $T(n) = T(n-1) + n$. What is wrong with the following proof?

First guess $T(n) = O(n)$.

Inductive basis: For $n = 1$, $T(1) = 1 = O(1)$.

Inductive step: Assume $T(n-1) = O(n-1)$

$$
T(n) = T(n-1) + n \\
= O(n-1) + n \\
= O(n).
$$
(5) Changing variables.

Example: \( T(n) = 2T(\sqrt{n}) + \log n. \)