Announcements

❖ I don’t have add codes
  ▪ CS Advisors closed registration on Monday; if you’re a senior and don’t have an add code yet ... let us know ASAP
  ▪ Otherwise, 373 is closed 😭

❖ I got my first anonymous feedback!
  ▪ Can you please clarify what is “too fast”?

❖ HW1 is due, HW2 is released
  ▪ Late policy: ~5% first day, ~10% second day, 20% third day
  ▪ Survey coming out soon

❖ Hint for HW2’s confusingTest(): commenting out resize() is not the answer
Questions from Reading Quiz

❖ I’m still confused how N(N-1)/2 was derived

❖ It’s not easy to find the exact count/cost, and it’s especially easy to be off by one. Will we have to calculate the exact count at some point?

❖ I found it interesting how selection sort has no best or worst case

❖ Are there any better ways to sort? N^2 seems like a pretty bad runtime.
Derivation: $N(N-1)/2$

$C = 1 + 2 + 3 + \cdots + (N - 3) + (N - 2) + (N - 1)$

$+ C = (N - 1) + (N - 2) + (N - 3) + \cdots + 3 + 2 + 1$

$2C = N + N + N + \cdots + N + N + N$

$2C = (N - 1)N$

$\therefore C = N(N - 1)/2$
Lecture Outline

❖ Intro to Deques

❖ Asymptotic Analysis Formalisms: O, Θ, and Ω

❖ Case Analyses: Best, Worst, Overall

❖ Asymptotic Analysis Case Study: PrintParty
Deques

Deque ADT. A collection storing an ordered sequence of elements.
- Each element is accessible by a zero-based index.
- A deque has a size defined as the number of elements in the deque.
- Elements can be added to the front or back.
- Optionally, elements can be removed from the front or back.

List ADT. A collection storing an ordered sequence of elements.
- Each element is accessible by a zero-based index.
- A list has a size defined as the number of elements in the list.
- Elements can be added to the front, back, or any index in the list.
- Optionally, elements can be removed from the front, back, or any index in the list.
Deques

❖ What series of List ADT methods or Deque ADT methods can build this sequence:
  ▪ Deque has addFirst, addLast
  ▪ List has those two, plus addAt(idx, val) and reserve(capacity)

❖ Example:
  ▪ addFirst(D), addFirst(C), addFirst(B), addFirst(A)
  ▪ Works for either Deque or List
Lecture Outline

- Intro to Deques

- Asymptotic Analysis Formalisms: $\Theta$, $O$, and $\Omega$

- Case Analyses: Best, Worst, Overall

- Asymptotic Analysis Case Study: PrintParty
Review: Algorithm Analysis: Our Destination

- “The worst-case order-of-growth for dup1’s runtime is quadratic (parabolic)”
- “The best-case order-of-growth for dup1’s runtime is constant”

```java
boolean dup1(int[] a)
```

- Best case:
  - Array contains a duplicate at front
    - Constant time: \( \Theta(1) \)
  - Array contains no duplicate items
    - Quadratic time: \( \Theta(N^2) \)
- Worst case:
  - Array contains a duplicate at front
    - Constant time: \( \Theta(1) \)
  - Array contains no duplicate items
    - Quadratic time: \( \Theta(N^2) \)
Big-O: Intuition

- Suppose we have a function $R(N)$ with order of growth $f(N)$. In Big-O notation, we write this as:

$$R(N) \in O(f(N))$$

- Big-O can informally be thought of as something like “less-than or equals”

<table>
<thead>
<tr>
<th>Function</th>
<th>Big-O</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N^3 + 3N^4$</td>
<td>$O(N^4)$</td>
</tr>
<tr>
<td>$(1 / N) + N^3$</td>
<td>$O(N^3)$</td>
</tr>
<tr>
<td>$Ne^N + N$</td>
<td>$O(Ne^N)$</td>
</tr>
<tr>
<td>$40 \sin(N) + 4N^2$</td>
<td>$O(N^2)$</td>
</tr>
</tbody>
</table>
Big-O: Intuition

❖ ... and also ...

<table>
<thead>
<tr>
<th>Function</th>
<th>Big-O</th>
<th>Also Big-O</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N^3 + 3N^4$</td>
<td>$O(N^4)$</td>
<td>$O(N^5)$</td>
</tr>
<tr>
<td>$(1 / N) + N^3$</td>
<td>$O(N^3)$</td>
<td>$O(N^{423421531542})$</td>
</tr>
<tr>
<td>$Ne^N + N$</td>
<td>$O(Ne^N)$</td>
<td>$O(N*3^N)$</td>
</tr>
<tr>
<td>$40 \sin(N) + 4N^2$</td>
<td>$O(N^2)$</td>
<td>$O(N^{2.1})$</td>
</tr>
</tbody>
</table>
Big-O: Mathematical Definition

\[ R(N) \in O(f(N)) \]
means there exists a positive constant \( k \) such that

\[ R(N) \leq k \cdot f(N) \]
for all values of \( N \) greater than some \( N_0 \).

Demo: https://www.desmos.com/calculator/kaxnmhsjni
Big-Theta: Intuition

- Whereas Big-O can informally be thought of as something like “less-than or equals”, Big-Theta more closely resembles “equals”

\[ R(N) \in \Theta(f(N)) \]

<table>
<thead>
<tr>
<th>Function</th>
<th>Big-Theta</th>
</tr>
</thead>
<tbody>
<tr>
<td>( N^3 + 3N^4 )</td>
<td>( \Theta(N^4) )</td>
</tr>
<tr>
<td>( \frac{1}{N} + N^3 )</td>
<td>( \Theta(N^3) )</td>
</tr>
<tr>
<td>( Ne^N + N )</td>
<td>( \Theta(\text{Ne}^N) )</td>
</tr>
<tr>
<td>( 40 \sin(N) + 4N^2 )</td>
<td>( \Theta(N^2) )</td>
</tr>
</tbody>
</table>
# Big-Theta: Intuition

- ... but not ...

<table>
<thead>
<tr>
<th>Function</th>
<th>Big-Theta</th>
<th>Big-O (but not big-theta)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N^3 + 3N^4$</td>
<td>$\Theta(N^4)$</td>
<td>$O(N^5)$</td>
</tr>
<tr>
<td>$(1 / N) + N^3$</td>
<td>$\Theta(N^3)$</td>
<td>$O(N^{423421531542})$</td>
</tr>
<tr>
<td>$Ne^N + N$</td>
<td>$\Theta(Ne^N)$</td>
<td>$O(N*3^N)$</td>
</tr>
<tr>
<td>$40 \sin(N) + 4N^2$</td>
<td>$\Theta(N^2)$</td>
<td>$O(N^{2.1})$</td>
</tr>
</tbody>
</table>
Big-Theta: Mathematical Definition

\[ R(N) \in \Theta(f(N)) \]

means there exist positive constants \( k_1 \) and \( k_2 \) such that

\[ k_1 \cdot f(N) \leq R(N) \leq k_2 \cdot f(N) \]

for all values of \( N \) greater than some \( N_0 \).

Demo:
https://www.desmos.com/calculator/wsej7ymhtc

\[ 40\sin(N) + 4N^2 \in \Theta(N^2) \]

since \( k_1 = 3, k_2 = 5, N_0 \approx 5 \)
Big-Theta Challenge

Find a simple \( f(N) \) and corresponding \( k_1 \) and \( k_2 \).

\[
R(N) = \frac{4N^2 + 3N \ln N}{2}
\]

Remember: \( R(N) \in \Theta(f(N)) \) means there exist positive constants \( k_1 \) and \( k_2 \) such that

\[
k_1 \cdot f(N) \leq R(N) \leq k_2 \cdot f(N)
\]

for all values of \( N \) greater than some \( N_0 \).
Big-Omega: Intuition

- Big-Omega is our “greater than or equals”

\[ R(N) \in \Omega(f(N)) \]

<table>
<thead>
<tr>
<th>Function</th>
<th>Big-Omega</th>
<th>Also Big-Omega</th>
</tr>
</thead>
<tbody>
<tr>
<td>(N^3 + 3N^4)</td>
<td>(\Omega(N^4))</td>
<td>&gt; (\Omega(N^3))</td>
</tr>
<tr>
<td>((1 / N) + N^3)</td>
<td>(\Omega(N^3))</td>
<td>&gt; (\Omega(1))</td>
</tr>
<tr>
<td>(Ne^N + N)</td>
<td>(\Omega(Ne^N))</td>
<td>&gt; (\Omega(N*2^N))</td>
</tr>
<tr>
<td>(40\sin(N) + 4N^2)</td>
<td>(\Omega(N^2))</td>
<td>&gt; (\Omega(N^{1.9}))</td>
</tr>
</tbody>
</table>
Big-Omega: Mathematical Definition

\[ R(N) \in \Omega(f(N)) \]
means there exists a positive constant \( k \) such that

\[ k \cdot f(N) \leq R(N) \]
for all values of \( N \) greater than some \( N_0 \).

\[ 40 \sin(N) + 4N^2 \in \Omega(N) \]
since \( k = 30 \) and \( N_0 \approx 7 \)
Big-O, Big-Theta, Big-Omega Relationship

❖ If a function f is in Big-Theta, what does it mean for its membership in Big-O and Big-Omega? Vice versa?

<table>
<thead>
<tr>
<th>Function</th>
<th>Big-O</th>
<th>Big-Theta</th>
<th>Big-Omega</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N^3 + 3N^4$</td>
<td>$O(N^4)$</td>
<td>$\Theta(N^4)$</td>
<td>$\Omega(N^4)$</td>
</tr>
<tr>
<td>$(1 / N) + N^3$</td>
<td></td>
<td>$\Theta(N^3)$</td>
<td></td>
</tr>
<tr>
<td>$Ne^N + N$</td>
<td></td>
<td>$\Theta(Ne^N)$</td>
<td></td>
</tr>
<tr>
<td>$40 \sin(N) + 4N^2$</td>
<td></td>
<td>$\Theta(N^2)$</td>
<td></td>
</tr>
</tbody>
</table>
Algorithm Analysis: Our Destination

❖ “The worst-case runtime for dup1 is $\Theta(N^2)$”

❖ “The best-case runtime for dup1 is $\Theta(1)$”

❖ Unless otherwise specified, we typically mean Big-Theta of worst case
Lecture Outline

❖ Intro to Deques
❖ Asymptotic Analysis Formalisms: $O$, $\Theta$, and $\Omega$
❖ Case Analyses: Best, Worst, Overall
❖ Asymptotic Analysis Case Study: PrintParty
Case Analysis, Redux

- Asymptotic analysis describes the function’s behavior as it approaches infinity, without regard to its input.
- Case analysis looks at a specific input or class of inputs.
- We’ve seen best case and worst case.
- There is also the “all case” aka “overall case”:
  - This is the case corresponding to all possible inputs.
  - (there’s also an amortized case, which we don’t discuss in 373.)

$\begin{array}{c|c|c}
\text{best} & 0 & \Theta \\
\text{worst} & & \\
\text{overall} & \Omega & \\
\end{array}$
Overall Asymptotic Runtime Bound for \textit{dup1}

- Give overall asymptotic bounds for \textit{dup1}’s runtime
  - Reminder: \((N^2 + 3N + 2)/2\)

<table>
<thead>
<tr>
<th>Case</th>
<th>Big-O</th>
<th>Big-Theta</th>
<th>Big-Omega</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best</td>
<td></td>
<td>(\Theta(1))</td>
<td></td>
</tr>
<tr>
<td>Worst</td>
<td></td>
<td>(\Theta(N^2))</td>
<td></td>
</tr>
<tr>
<td>Overall</td>
<td></td>
<td>(\emptyset)</td>
<td></td>
</tr>
</tbody>
</table>

\(\emptyset\) does not exist, since we have different \(\Theta\)-bounds for best and worst cases

- Demo: \(\text{https://www.desmos.com/calculator/xjkrsvxus}\)

- Even though case analysis \(!=\) asymptotic analysis, we can sometimes infer the overall case from asymptotic analyses
More Practice: Mystery

boolean mystery(int[] a, int target) {
    int N = a.length;
    for (int i = 0; i < N; i += 1)
        if (a[i] == target)
            return true;
    return false;
}

<table>
<thead>
<tr>
<th>Case</th>
<th>Big-Theta</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best</td>
<td>(\Theta(1))</td>
</tr>
<tr>
<td>Worst</td>
<td>(\Theta(N))</td>
</tr>
<tr>
<td>Overall</td>
<td>(\emptyset)</td>
</tr>
</tbody>
</table>

Conditional early termination may indicate that an overall bound does not exist.
More Practice: Selection Sort

```java
void selectionSort(int[] a) {
    int N = a.length;
    for (int i = 0; i < N; i++) {
        int smallestSoFar = a[i];
        for (int j = i + 1; j < N; j++) {
            if (a[j] < a[smallestSoFar]) {
                smallestSoFar = j;
            }
        }
        int tmp = a[i];
        a[i] = a[smallestSoFar];
        a[smallestSoFar] = tmp;
    }
}
```

<table>
<thead>
<tr>
<th>Case</th>
<th>Big-Theta</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best</td>
<td>$\Theta(N^2)$</td>
</tr>
<tr>
<td>Worst</td>
<td>$\Theta(N^2)$</td>
</tr>
<tr>
<td>Overall</td>
<td>$\Theta(N^2)$</td>
</tr>
</tbody>
</table>
Lecture Outline

- Intro to Deques
- Asymptotic Analysis Formalisms: O, Θ, and Ω
- Case Analyses: Best, Worst, Overall
- Asymptotic Analysis Case Study: PrintParty
Find an \( f(N) \) such that the runtime \( R(N) \in \Theta(f(N)) \).

A. 1  
B. \( \log N \)  
C. \( N \)  
D. \( N \log N \)  
E. \( N^2 \)  
F. I’m not sure ...

```java
void printParty(int N) {
    for (int i = 1; i <= N; i *= 2) {
        for (int j = 0; j < i; j += 1) {
            System.out.println("hello");
        }
    }
}
```

Why don’t we have multiple cases to consider?

No conditional exits! So let’s find the overall bound.
PrintParty, Geometrically

```java
void printParty(int N) {
    for (int i = 1; i <= N; i *= 2) {
        for (int j = 0; j < i; j += 1) {
            System.out.println("hello");
        }
    }
}
```

<table>
<thead>
<tr>
<th>i</th>
<th>j</th>
<th>C(N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>same as</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>17</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>18</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

The number of print statements increases geometrically as i and j increase.
PrintParty, Geometrically

```java
void printParty(int N) {
    for (int i = 1; i <= N; i *= 2) {
        for (int j = 0; j < i; j += 1) {
            System.out.println("hello");
        }
    }
}
```

N:

|   | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
|---|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|
C(N):

|   | 1 | 3 | 3 | 7 | 7 | 7 | 7 | 15 | 15 | 15 | 15 | 15 | 15 | 15 | 31 | 31 | 31 |
PrintParty, Counting

```java
void printParty(int N) {
    for (int i = 1; i <= N; i *= 2) {
        for (int j = 0; j < i; j += 1) {
            System.out.println("hello");
        }
    }
}
```

---

The function `printParty` prints the string "hello" for each multiple of 2 from 1 to `N`. The table shows the counts `C(N)` for various values of `N`.

```
N: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
C(N): 1 3 3 7 7 7 15 15 15 15 15 15 15 15 31 31 31...
```

The formula `C(N) = \sum_{i=0}^{\lfloor \log_2 N \rfloor} 2^i` counts the number of multiples of powers of 2 up to `N`. However, what does this equal?
## PrintParty, Using Examples

<table>
<thead>
<tr>
<th>N</th>
<th>C(N)</th>
<th>½N</th>
<th>2N</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0.5</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>1 + 2 + 4 = 7</td>
<td>2</td>
<td>14</td>
</tr>
<tr>
<td>7</td>
<td>1 + 2 + 4 = 7</td>
<td>3.5</td>
<td>14</td>
</tr>
<tr>
<td>8</td>
<td>1 + 2 + 4 + 8 = 15</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>27</td>
<td>1 + 2 + 4 + 8 + 16 = 31</td>
<td>13.5</td>
<td>54</td>
</tr>
<tr>
<td>185</td>
<td>... + 32 + 64 + 128 = 255</td>
<td>92.5</td>
<td>370</td>
</tr>
<tr>
<td>715</td>
<td>... + 256 + 512 = 1023</td>
<td>357.5</td>
<td>1430</td>
</tr>
</tbody>
</table>

\[
\text{let } k_1 = \frac{1}{2} \text{ and } k_2 = 2N \\
\therefore C(N) \in \Theta(N)
\]
tl;dr: Asymptotic Analysis for Iterative Problems

- Case Analysis != Asymptotic Analysis

- Memorize these summations since they’re common:
  \[1 + 2 + 3 + 4 + \ldots + (N-1) = \frac{N(N-1)}{2} \in \Theta(N^2)\]
  \[1 + 2 + 4 + 8 + \ldots + 2^{\text{floor}(\log_2 N)} = 2N - 1 \in \Theta(N)\]

- Strategies for finding an asymptotic bound:
  - Use a geometric argument / visualizations
  - Find an expression for the exact step count
  - Write out examples