Stability

Problem: Sort a Flight Departures table by destination name, then ... by departure time

need **stable sorting**
Stability

A sorting algorithm is stable if it preserves the order of equal elements.
A sorting algorithm is **stable** if it preserves the order of **equal** elements.
Stability

Insertion sort is stable

equal items never pass each other
Stability

Insertion sort is stable
  equal items never pass each other

Selection sort is not stable
  long distance swaps
Stability

Insertion sort **is stable**
   equal items never pass each other

Selection sort **is not stable**
   long distance swaps

Merge sort **is stable**
   as long as equal items are chosen from left subarray when merging
Lower Bound for Sorting
Sorting based on Comparisons

Basic operation: compare two items
Sorting based on Comparisons

Basic operation: **compare** two items

What is a **lower bound** for the cost of sorting algorithms?
Decision Tree (sorting x, y, z)

x < y
Decision Tree (sorting x, y, z)

- If $x < y$ then go to yes
- If $y < z$ then go to $y < z$
Decision Tree (sorting x, y, z)

x < y

y < z

x y z

yes

yes
Decision Tree (sorting x, y, z)

1. If $x < y$, then yes.
2. If $y < z$, then yes.
3. If $x < z$, then yes.
4. If $x \geq y$, then no.
Decision Tree (sorting x, y, z)

- x < y
- y < z
- x < z

- Yes
- No
Decision Tree (sorting x, y, z)

- x < y
- y < z
- x < z
- x  y  z
- x  z  y
- z  x  y
- yes
- yes
- yes
- yes
- no
- no
Decision Tree (sorting x, y, z)

- If x < y:
  - If y < z:
    - If x < z:
      - yes
    - no
  - yes
  - no
- no

The final sorted order is determined by the path taken through the tree. For example, if x < y and y < z and x < z, then x, y, z is the sorted order. If x < y and y < z but x ≥ z, then z, x, y is the sorted order.
Decision Tree (sorting x, y, z)
Decision Tree (sorting x, y, z)

- If \( x < y \), then:
  - If \( y < z \), then:
    - \( x < z \): \( x y z \)
    - \( y < z \): \( x z y \)
  - \( y < z \): \( y z x \)
  - \( y < z \): \( y z x \)
- If \( x < z \), then:
  - \( y < z \): \( y z x \)
  - \( y < z \): \( y z x \)
- If \( y < z \), then:
  - \( x < z \): \( x z y \)
  - \( y < z \): \( z x y \)

The decision tree sorts x, y, z based on the relationships among them.
Decision Tree (sorting x, y, z)

```
x < y
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>y &lt; z</td>
<td>x &lt; z</td>
</tr>
<tr>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>
```

```
x < z
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>x &lt; z</td>
<td>y &lt; z</td>
</tr>
<tr>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>
```

```
x < z
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>x &lt; z</td>
<td>y &lt; z</td>
</tr>
<tr>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>
```

```
x y z
```

```
x < z
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>x &lt; z</td>
<td>y &lt; z</td>
</tr>
<tr>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>
```
Decision Tree (sorting x, y, z)

- Decision Tree for sorting three items (x, y, z)
  - Root: x < y
    - If x < y, go to next step.
    - If not, x < z.
  - If x < z, go to next step.
    - If y < z, go to next step.
    - If not, y < z.
  - If y < z, go to next step.
    - If x < z, go to next step.
    - If not, x < z.
  - If x < z, go to next step.
    - If y < z, go to next step.
    - If not, y < z.

- Final outcomes:
  - Yes: x y z, x z y, z x y
  - No: y x z, y z x, z y x
Decision Tree (sorting x, y, z)

Number of leaves:
number of all possible orderings
Decision Tree (sorting $x$, $y$, $z$)

Number of **leaves**: number of all possible orderings

Height of the decision tree: worst-case **number of comparisons**
What is the height?

Consider sorting $n$ distinct items
What is the height?

Consider sorting $n$ distinct items.

Number of leaves at least $n!$ (# permutations)
What is the height?

Consider sorting $n$ distinct items

- number of leaves at least $n!$ (# permutations)
- number of leaves at most $2^h$ (full and complete tree)
What is the height?

Consider sorting \( n \) distinct items

- number of leaves at least \( n! \) (# permutations)
- number of leaves at most \( 2^h \) (full and complete tree)

\[
2^h \geq \# \text{ leaves} \geq n!
\]

\[
h \geq \log n!
\]

\[
h \geq n \log n \quad \text{by Stirling's formula}
\]
Cost of Sorting

What is a **lower bound** for the cost of sorting algorithms?

O(n log n)

Any O(n log n) sorting algorithm is considered **optimal**