We want to insert $n$ elements into a BST so that they will be stored in sorted order.

\[
\text{InOrder walk: } 1 \ 2 \ 3 \ 5 \ 6 \ 7 \ 8
\]
Binary Search Trees - Built Randomly

We want to insert n elements into a BST so that they will be stored in sorted order.

InOrder walk: 1 2 3 5 6 7 8

Insertion: nothing fancy. Just read elements and insert into current tree.
Given array of elements: 3 1 8 2 6 7 5
Given array of elements: 3 1 8 2 6 7 5

3 → 3

1
Given array of elements: 3 1 8 2 6 7 5
Given array of elements: 3 1 8 2 6 7 5
Given array of elements: 3 1 8 2 6 7 5
Given array of elements: 3 1 8 2 6 7 5
Given array of elements: 3 1 8 2 6 7 5

This is a sorting algorithm.
Given the very simple BST-sort/construction algorithm

- how long can it take to build the BST?
- what shape will it have? How balanced? What depth?
Given the very simple BST-sort/construction algorithm
- how long can it take to build the BST?
- what shape will it have? How balanced? What depth?

Depends on input sequence

1 2 3 4 5 → 1

Can be bad: $O(n)$ depth
$O(n^2)$ time
$\Omega(?)$
Given the very simple BST-sort/construction algorithm
- how long can it take to build the BST?
- what shape will it have? How balanced? What depth?

Depends on input sequence

1 2 3 4 5 → 1 1 2 1 2 2 1 2 2 3 3 1 2 2 3 3 4 4 5

Can be bad:
- \(O(n)\) depth
- \(O(n^2)\) time
- \(\Omega(n \log n)\) worst-case time: sorting lower bound
Even for a balanced tree,\
$\Theta(n) \approx \frac{n}{2}$ nodes have height = $\Theta(\log n)$
so it must take $\Omega(n \log n)$ time to build.
Even for a balanced tree, \( \Theta(n) \approx \frac{n}{2} \) nodes have height = \( \Theta(\log n) \) so it must take \( \Omega(n \log n) \) time to build.

Any algorithm producing any tree shape : \( \Omega(n \log n) \) time
Even for a balanced tree, \( \Theta(n) \approx \frac{n}{2} \) nodes have height = \( \Theta(\log n) \)
so it must take \( \Omega(n \log n) \) time to build.

Any algorithm producing any tree shape : \( \Omega(n \log n) \) time

So ... if lucky, \( \Theta(n \log n) \) time
if unlucky, \( O(n^2) \) time
Even for a balanced tree, $\Theta(n) \approx \frac{n}{2}$ nodes have height = $\Theta(\log n)$

so it must take $\Omega(n \log n)$ time to build.

Any algorithm producing any tree shape: $\Omega(n \log n)$ time

So... if lucky, $\Theta(n \log n)$ time

if unlucky, $O(n^2)$ time

Sounds like...
Even for a balanced tree, \( \Theta(n) \approx \frac{n}{2} \) nodes have height = \( \Theta(\log n) \)
so it must take \( \Omega(n \log n) \) time to build.

Any algorithm producing any tree shape: \( \Omega(n \log n) \) time

So... if lucky, \( \Theta(n \log n) \) time
if unlucky, \( O(n^2) \) time

Sounds like... quicksort
Stable quicksort

1 8 2 6 7 5

use first elt to partition → 1 2 3 8 6 7 5
Stable quicksort

1 8 2 6 7 5

- Use first elt to partition
- Repeat on each side
Stable quicksort

1 3 8 2 6 7 5

• use first elt to partition

• repeat on each side

• 3rd round
Stable quicksort

1 8 2 6 7 5

• use first elt to partition

• repeat on each side

• 3rd round

• 4th round
Stable quicksort

1 8 2 6 7 5

• Use first elt to partition
• Repeat on each side
• 3rd round
• 4th round

Same tree as BST
Stable quicksort

- use first elt to partition
- repeat on each side

quicksort round 1: compare all els to 3
BST sort: 3 = root; eventually all els pass through.

3 1 8 2 6 7 5

3rd round

1 2 3 8 6 7 5

4th round

1 2 3 8 6 7 5
same tree as BST
Stable quicksort

1 8 2 6 7 5

- Use first elt to partition
- Repeat on each side

Quick sort round 1: compare all elts to 3
BST sort: 3 = root; eventually all elts pass through.

Quick sort: partitions into 2 groups
<3 & >3
Each is independent

BST sort: same

3rd round

4th round

Same tree as BST
Stable quicksort

1 8 2 6 7 5

- Use first elt to partition
- Repeat on each side

Quick sort round 1: compare all els to 3
BST sort: 3 = root; eventually all els pass through.

Quick sort: partitions into 2 groups <3 & >3
Each is independent

BST sort: same
Exactly same comparisons but in different order

3rd round

4th round

Same tree as BST
We've seen \( \text{QUICKSORT} \approx \text{BST-SORT} \) (stable)

\[ \exists \text{Randomized versions have same analysis} \]

\[ E[\text{BST build}] = \Theta(n \log n) \text{ time} \]

\[ E[\text{insert node}] = \Theta(\log n) \text{ time} \]
Intuition: $\mathbb{E}[\text{depth}] = \Theta(\log n)$ so it should be \~balanced
Intuition: $E[\text{depth}] = \Theta(\log n)$ so it should be \textit{not} balanced. No.
Intuition: $E[\text{depth}] = \Theta(\log n)$ so it should be unbalanced? No.
Intuition: $E[\text{depth}] = \Theta(\log n)$ so it should be $\sim$ balanced? No

- $n - \sqrt{n}$ nodes
- $\sqrt{n}$ nodes
- $\log(n - \sqrt{n}) < \log n$ 
- Exaggerated depth
- Exaggerated number of nodes

Average depth $< \frac{1}{n} \cdot (n \cdot \log n + \sqrt{n} \cdot (\sqrt{n} + \log n))$
Intuition: \( \mathbb{E}[\text{depth}] = \Theta(\log n) \) so it should be \( \sim \) balanced? \( \text{No} \)

\[
\log(n-\sqrt{n}) < \log n
\]

\[
\sqrt{n} \quad \text{nodes}
\]

\[
\sqrt{n} \quad \text{nodes}
\]

\[
\sqrt{\frac{n}{\log n}} \quad \text{exaggerated depth}
\]

\[
\text{exaggerated # nodes}
\]

\[
\text{# nodes}
\]

Average depth \( < \frac{1}{n} \cdot \left( n \cdot \log n + \sqrt{n} \cdot (\sqrt{n} + \log n) \right) \)

\[
= \log n + 1 + \frac{\log n}{\sqrt{n}}
\]

\[
= O(\log n) \quad \text{so } \mathbb{E}[\text{depth}] \not\rightarrow \text{ balance}
\]
Expected height of randomly built BST
$H(n) = ?$
$H(n) = 1 + \ ?$
\[ H(n) = 1 + \max \{ H(k), H(n-k-1) \} \]

for some random \( k \) (\( 0 \leq k \leq n-1 \))
\[ H(n) = 1 + \max \{ H(k), H(n-k-1) \} \]

for some random \( k \) \( (0 \leq k \leq n-1) \)

If \( \frac{n}{4} < k < \frac{3n}{4} \)

\( \Leftarrow H(n) \leq ? \)
\[ H(n) = 1 + \max \{ H(k), H(n-k-1) \} \]

for some random \( k \) \( (0 \leq k \leq n-1) \)

If \( \frac{n}{4} < k < \frac{3n}{4} \)

\[ \rightarrow H(n) \leq 1 + H\left(\frac{3n}{4}\right) \]

else \( H(n) \leq ? \)
\[ H(n) = 1 + \max \{ H(k), H(n-k-1) \} \]

for some random \( k \) \( (0 \leq k \leq n-1) \)

\[ \text{if} \quad \frac{n}{4} < k < \frac{3n}{4} \]

\[ \leftrightarrow H(n) \leq 1 + H\left(\frac{3n}{4}\right) \]

\[ \text{else} \quad H(n) \leq 1 + H(n-1) < 1 + H(n) \]
\[ H(n) = 1 + \max \{ H(k), H(n-k-1) \} \]

for some random \( k \) \((0 \leq k \leq n-1)\)

\[
\begin{align*}
\text{If } & \frac{n}{4} < k < \frac{3n}{4} \\
\leftrightarrow & \quad H(n) \leq 1 + H\left(\frac{3n}{4}\right) \\
\text{else } & \quad H(n) \leq 1 + H(n-1) \\
& \quad < 1 + H(n)
\end{align*}
\]

\[ E[H(n)] \quad ? \]
\[ H(n) = 1 + \max \{ H(k), H(n-k-1) \} \]

for some random \( k \) \( (0 \leq k \leq n-1) \)

If \( \frac{n}{4} < k < \frac{3n}{4} \)

\[ \iff H(n) \leq 1 + H\left(\frac{3n}{4}\right) \]

else \( H(n) \leq 1 + H(n-1) \)

\[ < 1 + H(n) \]

\[ E[H(n)] \leq 1 + \frac{1}{2} E[H\left(\frac{3n}{4}\right)] + \frac{1}{2} E[H(n)] \]
\[ H(n) = 1 + \max \{ H(k), H(n-k-1) \} \]

for some random \( k \) \( (0 \leq k \leq n-1) \)

If \( \frac{n}{4} < k < \frac{3n}{4} \)
\[ \implies H(n) \leq 1 + H\left(\frac{3n}{4}\right) \]

else \( H(n) \leq 1 + H(n-1) \)
\[ < 1 + H(n) \]

\[ \mathbb{E}[H(n)] \leq 1 + \frac{1}{2} \mathbb{E}[H\left(\frac{3n}{4}\right)] + \frac{1}{2} \mathbb{E}[H(n)] \]

\[ \frac{1}{2} \mathbb{E}[H(n)] \leq 1 + \frac{1}{2} \mathbb{E}[H\left(\frac{3n}{4}\right)] \]
\[ H(n) = 1 + \max \{ H(k), H(n-k-1) \} \]

for some random \( k \) \( (0 \leq k \leq n-1) \)

\[
\begin{align*}
\text{if } \frac{n}{4} < k < \frac{3n}{4} \\
\quad \Rightarrow H(n) \leq 1 + H(\frac{3n}{4})
\end{align*}
\]

\[
\begin{align*}
\text{else } \quad H(n) \leq 1 + H(n-1) \\
\quad \quad < 1 + H(n)
\end{align*}
\]

\[
\begin{align*}
E[H(n)] \leq 1 + \frac{1}{2} E[H(\frac{3n}{4})] + \frac{1}{2} E[H(n)] \\
\frac{1}{2} E[H(n)] \leq 1 + \frac{1}{2} E[H(\frac{3n}{4})] \\
E[H(n)] \leq 2 + E[H(\frac{3n}{4})]
\end{align*}
\]
\[ H(n) = 1 + \max \{ H(k), H(n-k-1) \} \]

for some random \( k \) \((0 \leq k \leq n-1)\)

\[ \text{if } \frac{n}{4} < k < \frac{3n}{4} \]
\[ \iff H(n) \leq 1 + H\left(\frac{3n}{4}\right) \]

\[ \text{else } H(n) \leq 1 + H(n-1) \]
\[ < 1 + H(n) \]

\[
\begin{align*}
\mathbb{E}[H(n)] &\leq 1 + \frac{1}{2} \mathbb{E}[H\left(\frac{3n}{4}\right)] + \frac{1}{2} \mathbb{E}[H(n)] \\
\frac{1}{2} \mathbb{E}[H(n)] &\leq 1 + \frac{1}{2} \mathbb{E}[H\left(\frac{3n}{4}\right)] \\
\mathbb{E}[H(n)] &\leq 2 + \mathbb{E}[H\left(\frac{3n}{4}\right)] = O(\log n)
\end{align*}
\]
$H(n) = 1 + \max \{ H(k), H(n-k-1) \}$ for some random $k \ (0 \leq k \leq n-1)$

If $\frac{n}{4} < k < \frac{3n}{4}$

\[ \Leftrightarrow H(n) \leq 1 + H\left(\frac{3n}{4}\right) \]

else $H(n) \leq 1 + H(n-1)$

\[ < 1 + H(n) \]

$E[H(n)] \leq 1 + \frac{1}{2} E[H\left(\frac{3n}{4}\right)] + \frac{1}{2} E[H(n)]$

$\frac{1}{2} E[H(n)] \leq 1 + \frac{1}{2} E[H\left(\frac{3n}{4}\right)]$

$E[H(n)] \leq 2 + E[H\left(\frac{3n}{4}\right)] = O(\log n)$

$2 \log_{4/3} n \approx 2 \cdot 2.4 \log_2 n < 5 \log n$
$H(n) = 1 + \max \{H(k), H(n-k-1)\}$

for some random $k$ ($0 \leq k \leq n-1$)

with rigorous analysis can get $\sim 3 \log n$

If $\frac{n}{4} < k < \frac{3n}{4}$

$\implies H(n) \leq 1 + H\left(\frac{3n}{4}\right)$

else $H(n) \leq 1 + H(n-1) < 1 + H(n)$

$E[H(n)] \leq 1 + \frac{1}{2} E[H\left(\frac{3n}{4}\right)] + \frac{1}{2} E[H(n)]$

$\frac{1}{2} E[H(n)] \leq 1 + \frac{1}{2} E[H\left(\frac{3n}{4}\right)]$

$E[H(n)] \leq 2 + E[H\left(\frac{3n}{4}\right)] = O(\log n)$

$2 \log_{4/3} n \approx 2 \times 2.4 \log_2 n < 5 \log n$