Chapter 5, Binary search trees:

- Binary search trees; the balanced BST problem (before break)
- AVL trees (last week Monday and Wednesday)
- Traditional red-black trees (last week Friday)
- Left-leaning red-black trees (this past Monday)
- “Wrap-up” BSTs, B-trees (Today)
- Begin dynamic programming (Friday)

Today:

- B-tree origin stories
  - Two-three trees
  - Sorted arrays
    - Linked/array hybrid
- B-tree definition
- B-tree implementation
Formally, a B-tree with maximum degree $M$ over some ordered key type is either

- empty, or
- a node with $d - 1$ keys and $d$ children, designated as lists $\text{keys}$ and $\text{children}$ such that
  - $\lceil M/2 \rceil \leq d \leq M$,
  - $\text{children}[0]$ is a B-tree such that all of the keys in that tree are less than $\text{keys}[0]$,
  - for all $i \in [1, d - 1)$, $\text{children}[i]$ is a B-tree such that all of the keys in that tree are greater than $\text{keys}[i - 1]$ and less than $\text{keys}[i]$,
  - and $\text{children}[d - 1]$ is a B-tree such that all of the keys in that tree are greater than $\text{keys}[d - 2]$. 
keys
eight bytes each position

values
four bytes each position

children
eight bytes each position

deg

offset in struct
0 8 88 96 100 140 144 152 240 248 252

node for keys less than ANT
node for keys between ANT and BEE
node for keys greater than WASP

node for keys less than ANT

A N T \0

B E E \0

W A S P \0
\[(M - 1) \sum_{i=0}^{h-1} M^i = (M - 1) \frac{M^h - 1}{M - 1} = M^h - 1\]

\[n = M^h - 1\]

\[M^h = n + 1\]

\[h = \log_M(n + 1)\]
\[ n = M^h - 1 \]

\[ M^h = n + 1 \]

\[ h = \log_M(n + 1) \]

\[ h = \log_{\frac{M}{2}}(n + 1) = \frac{\log_M(n + 1)}{1 - \log_M 2} \]
Cost of a search:

\[ \lg M \cdot h = \lg M \cdot \frac{\log_M(n+1)}{1 - \log_M 2} \]

\[
= \lg M \cdot \frac{\lg(n+1)}{1 - \frac{\lg 2}{\lg M}} 
\]

\[
= \frac{\lg(n+1)}{1 - \frac{1}{\lg M}} \]

\[
= \frac{\lg M}{\lg M - 1} \lg(n + 1)
\]

Compare: 1.44 \lg n for AVL trees, 2 \lg n for RB trees.
Let $c_0$ be the cost of searching at a node (proportional to $\log M$ and $c_1$ be the cost of reading a node from memory. The the cost of an entire search is

$$(c_0 + c_1) \frac{\log_M(n+1)}{1 - \log_M 2}$$

Now, consolidate the constants by letting $d = \frac{c_0 + c_1}{1 - \log_M 2}$, and we have

$$d \log_M(n + 1)$$
Coming up:

**Do Traditional RB project** *(suggested by Monday, Mar 28)*
*(Recommended: Do Left-leaning RB project for your own practice)*

**Due Wed, Mar 23** *(today)* *(end of day)* *(but hopefully you’ve spread it out)*
Read Sections 5.(4-6) *[some parts carefully, some parts skim, some parts optional—see Schoology]*
Do Exercise 5.14
Take quiz

**Due Fri, Mar 25** *(end of day)*
Read Section 6.(1&2)
Do Exercises 6.(5–7)
Take quiz

**Due Mon, Mar 25** *(end of day)*
Read Section 6.3
Do Exercises 6.(16, 19, 23, 33)
Take quiz