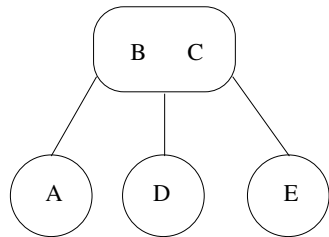
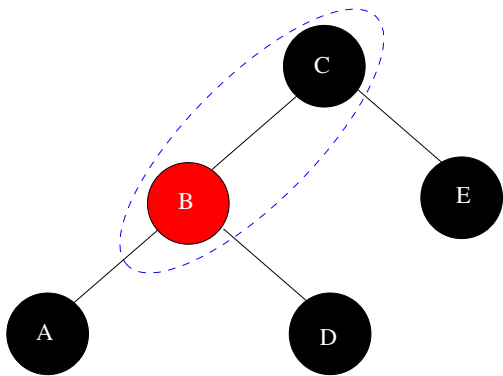


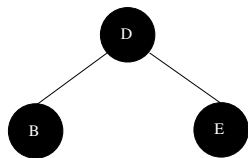
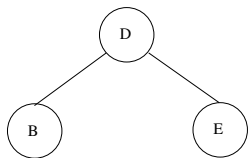
Chapter 5, Binary search trees:

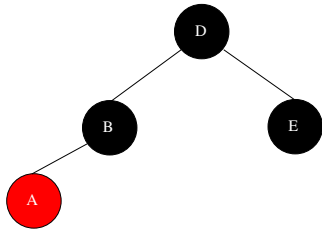
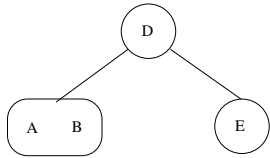
- ▶ Binary search trees; the balanced BST problem (before break)
- ▶ AVL trees (last week Monday and Wednesday)
- ▶ Traditional red-black trees (last week Friday)
- ▶ Left-leaning red-black trees (this past Monday)
- ▶ “Wrap-up” BSTs, B-trees (**Today**)
- ▶ Begin dynamic programming (Friday)

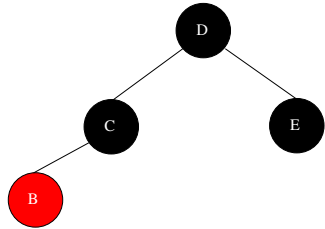
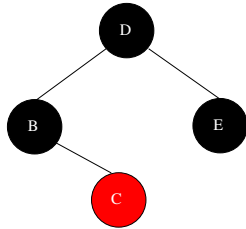
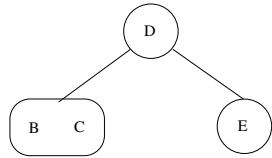
Today:

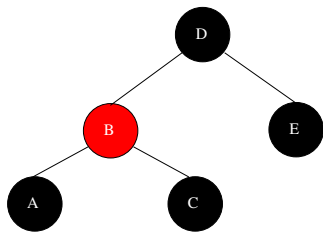
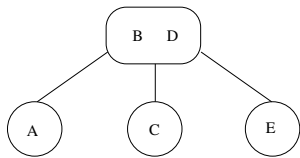
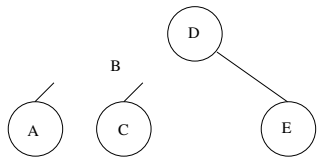
- ▶ B-tree origin stories
 - ▶ Two-three trees
 - ▶ Sorted arrays
 - ▶ Linked/array hybrid
- ▶ B-tree definition
- ▶ B-tree implementation

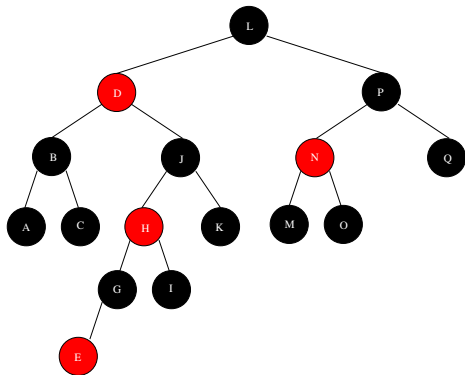
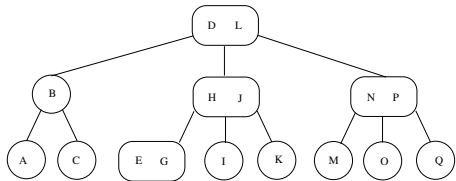


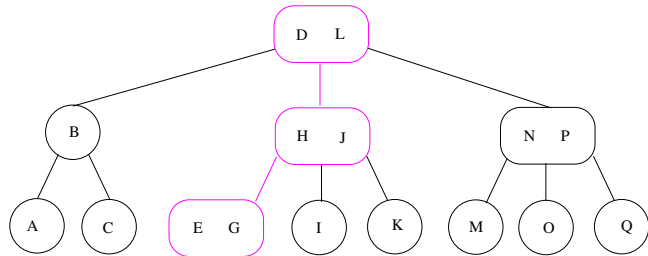


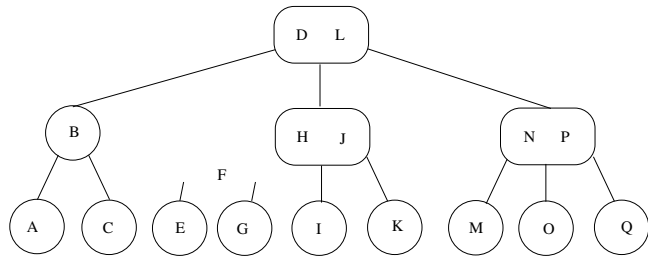


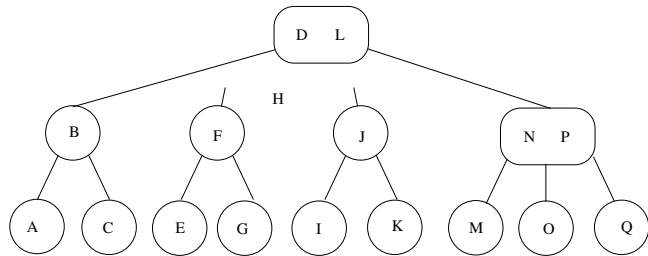


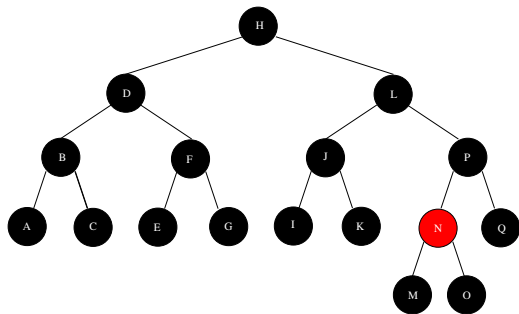
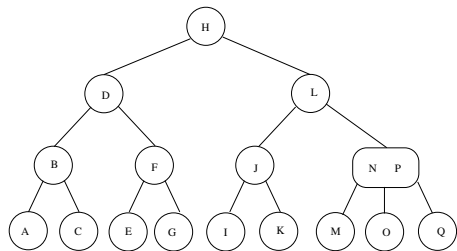




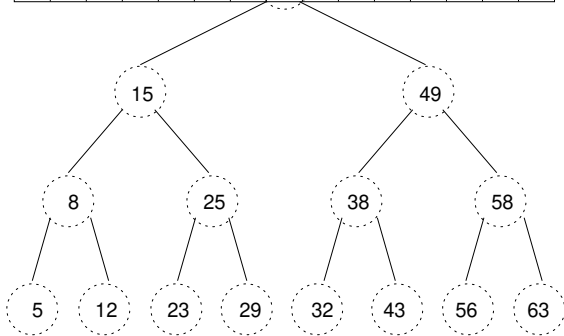


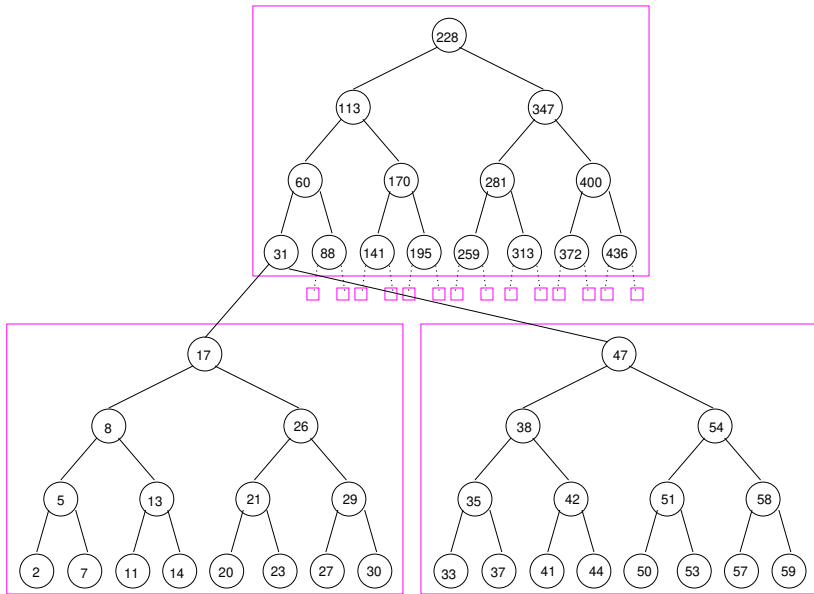


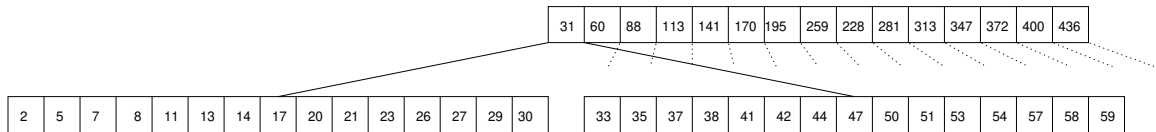


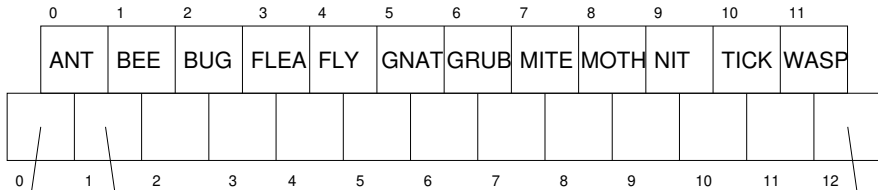


5	8	12	15	23	25	29	31	32	38	43	49	56	58	63
---	---	----	----	----	----	----	----	----	----	----	----	----	----	----









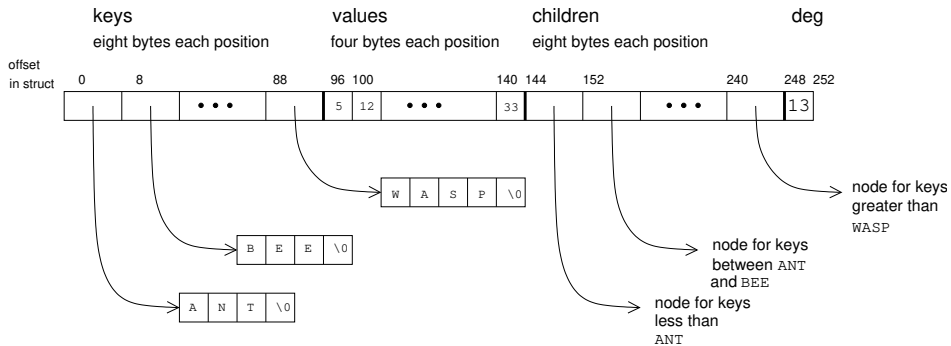
Subtree with keys less than ANT

Subtree with keys between ANT and BEE

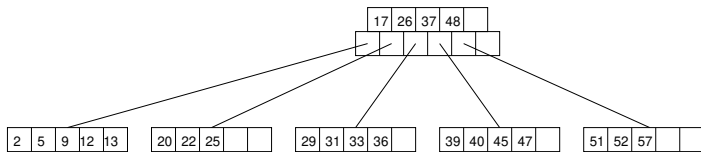
Subtree with keys greater than WASP

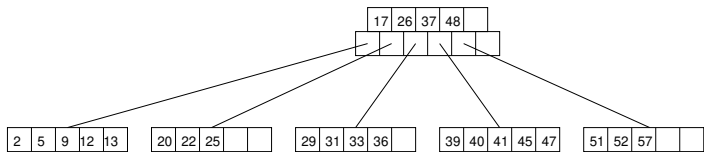
Formally, a B-tree with maximum degree M over some ordered key type is either

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



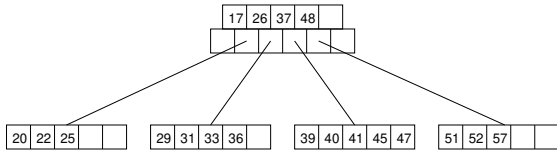
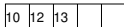


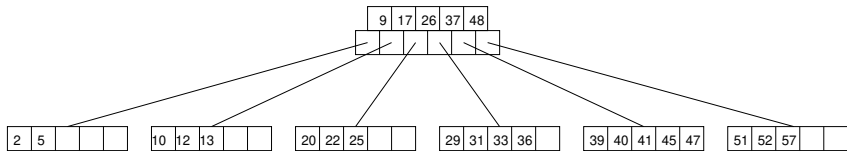


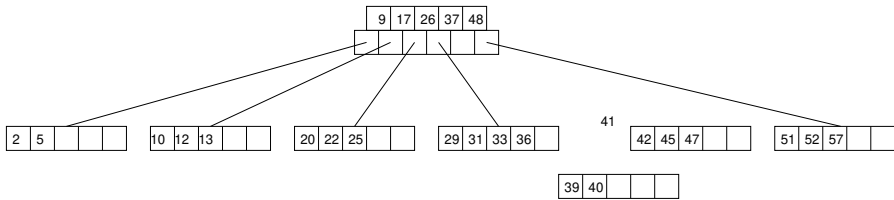




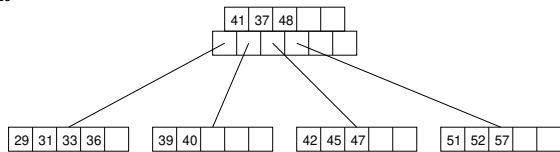
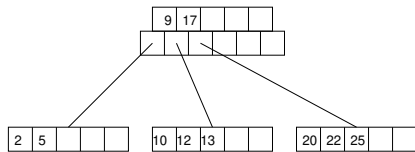
9

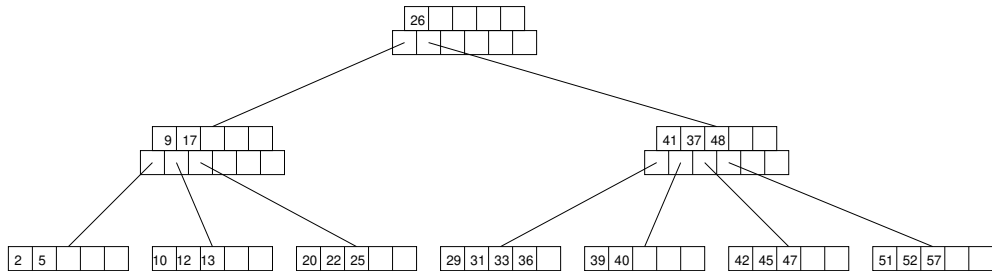






26





$$\underbrace{(M-1)}_{\substack{\text{keys per} \\ \text{node}}} \underbrace{\sum_{i=0}^{h-1} M^i}_{\substack{\text{sum of} \\ \text{nodes} \\ \text{at each} \\ \text{level}}} = (M-1) \frac{M^h - 1}{M - 1} = M^h - 1$$

$$n = M^h - 1$$

$$M^h = n + 1$$

$$h = \log_M(n + 1)$$

$$n = M^h - 1$$

$$M^h = n + 1$$

$$h = \log_M(n + 1)$$

$$h = \log_{\frac{M}{2}}(n + 1) = \frac{\log_M(n + 1)}{1 - \log_M 2}$$

Cost of a search:

$$\begin{aligned}\lg M \cdot h &= \lg M \cdot \frac{\log_M(n+1)}{1-\log_M 2} \\ &= \lg M \frac{\frac{\lg(n+1)}{\lg M}}{1-\frac{\lg 2}{\lg M}} \\ &= \frac{\lg(n+1)}{1-\frac{1}{\lg M}} \\ &= \frac{\lg M}{\lg M - 1} \lg(n+1)\end{aligned}$$

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 $\lg M$) 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