Syllabus

4. Intro Data Structures

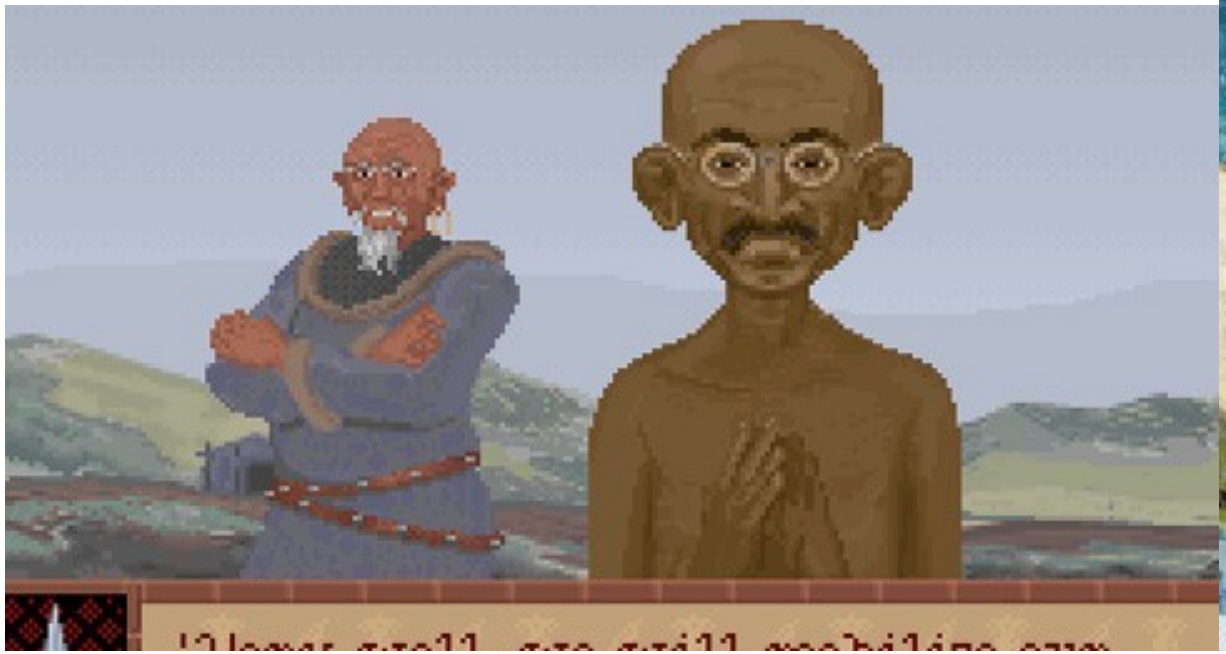
- Hardware/Math Data Types
- Abstract Data Types
- Basic: Boolean, Integer
- Derived: Array, Queue, Stack
- Linked Data Structures
- (Balanced) Search Trees
- AVL Trees

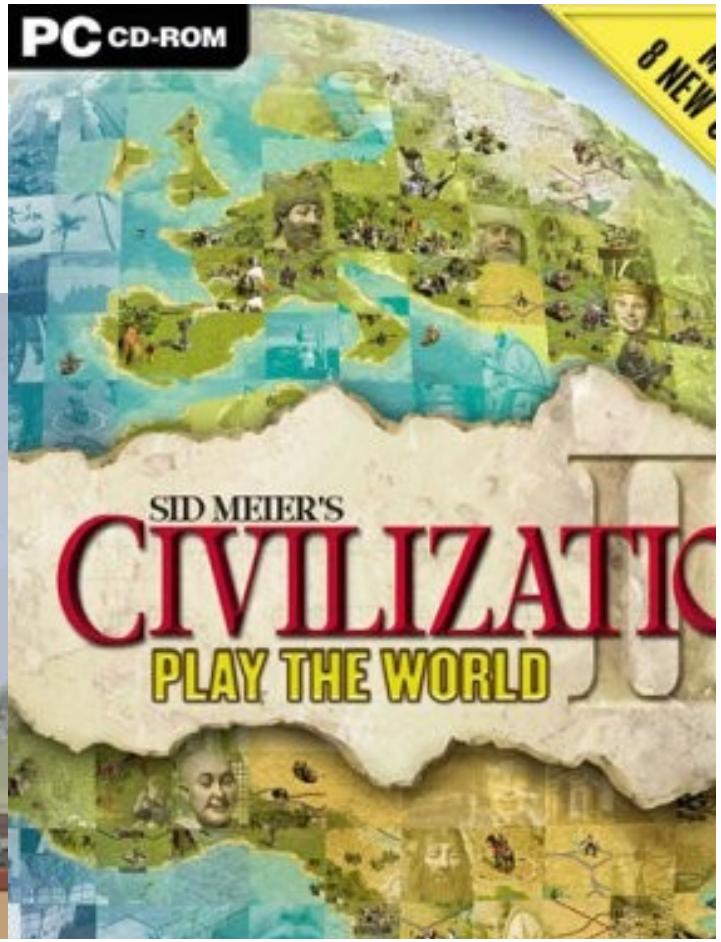
- Specification
- Primitives: semantics and cost
- Design
- Analysis
- (Optimality)

Hardware vs. Math. Data Types

- Each country/leader described by "A.I." character parameters.
- Initially *Gandhi*.**aggression** := 1
- When country adopts democracy, **`aggression -= 2`**.

(signed) byte/word/dword
≠ (ℕ) ℤ





Recap from Logic / *Model Theory*

Boolean $\mathbb{B}$	lattice	$0, 1, \wedge, \vee, \neg$
Integer $\mathbb{Z}$	ring	$0, 1, +, -, \times,$ div $, >$
Real $\mathbb{R}$	field	$0, 1, +, -, \times, \div, >$
Complex $\mathbb{C}$	field	$0, 1, +, -, \times, \div$
polynomials $\mathbb{R}[X]$	ring	$0, 1, +, -, \times,$ div
powerset $\mathcal{P}(S)$	lattice	$\varnothing, S, \cap, \cup, \setminus$
Grassmannian $\mathcal{G}(V)$	lattice	$\{0\}, V, \cap, \oplus, {}^{\perp}$
Presburger Integers $\mathbb{Z}$	group	$0, 1, +, -, >$

Meta-Def: A *structure* is a set, together with some

- constants,
- functions, and
- relations

Axioms are propositions satisfied by said structure.

Abstract Data *Types*

Boolean $\mathbb{B}$ $\quad 0\,,\,1\,,\,\wedge\,,\,\vee\,,\,\neg$

Integer $\mathbb{Z}$ (vs. byte/word etc.) $\quad 0\,,\,1\,,+\,,-\,,\times\,,\ \mathrm{div}\,,\ >$

Real $\mathbb{R}$ (vs. float/double etc.)

Stack of X $\quad \mathrm{new},\ \mathrm{push}\ x,\ \ \mathrm{pop},\ \ \mathrm{isEmpty}$

Queue of X $\quad \mathrm{new},\ \mathrm{enque}\ x,\ \ \mathrm{deque},\ \ \mathrm{isEmpty}$

FIFO

LIFO:
$\mathrm{pop} \circ \mathrm{push} \ = \ \mathrm{id}$

Definition: An *abstract data type* is a "*class of objects*" together with certain operations.

Axioms describe their behavior

regardless of **implementation**

Data *Structures* and Cost

Boolean $\mathbb{B}$ — $O(1)$ — 0 , 1 , $\wedge$, $\vee$, $\neg$

Integer $\mathbb{Z}$ (vs. byte/word etc.) — 0 , 1 , $+$, $-$, $\times$, div , $>$ — $O(n \cdot \log n)$

Real $\mathbb{R}$ (vs. float/double etc.)

Stack of X — $O(n) = O(\log |x|)$ — new, push x, pop, isEmpty

Queue of X — new, enque x, deque, isEmpty

(Dynamic) array of X — $O(1)$? — [], size, (re-size), search — $O(n)$

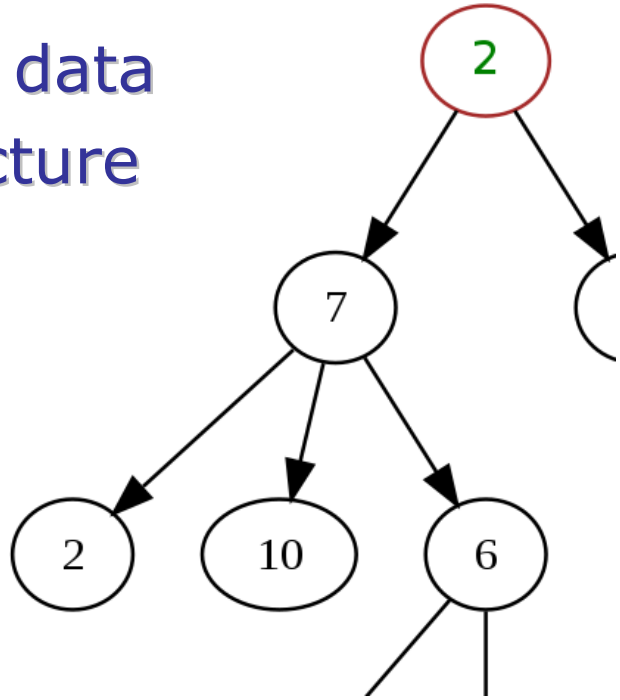
Sorted array of Y — $O(\log n)$ — search y , insert y , delete y — $O(n)$

Priority queue of Y — findmin, insert y — $O(\log n)$

where Y is totally ordered

Linked Data Structures

Tree data structure

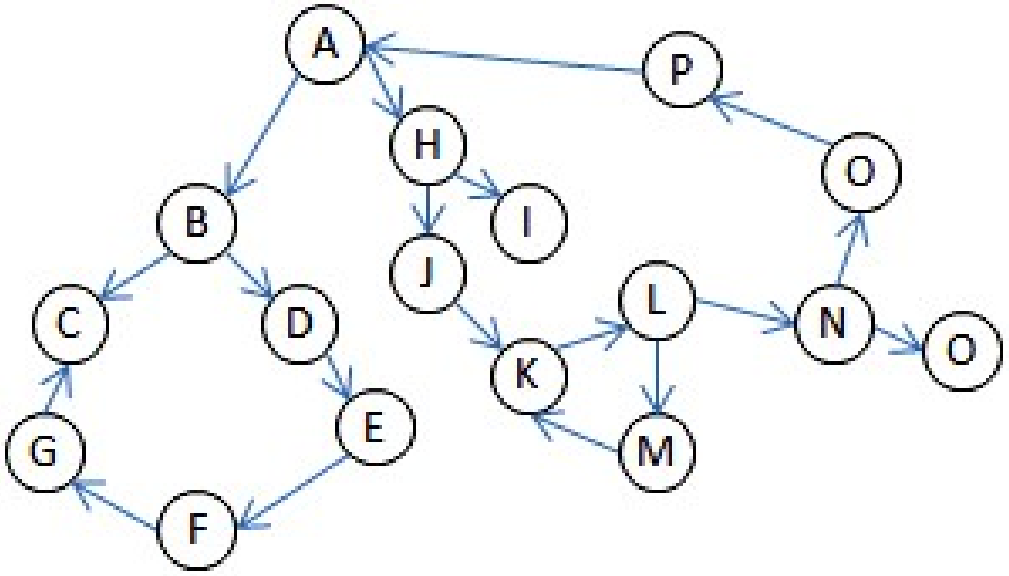


(Doubly) linked list:

- insert, delete in $O(1)$
- **search in $O(n)$**

(Un-) directed graph:

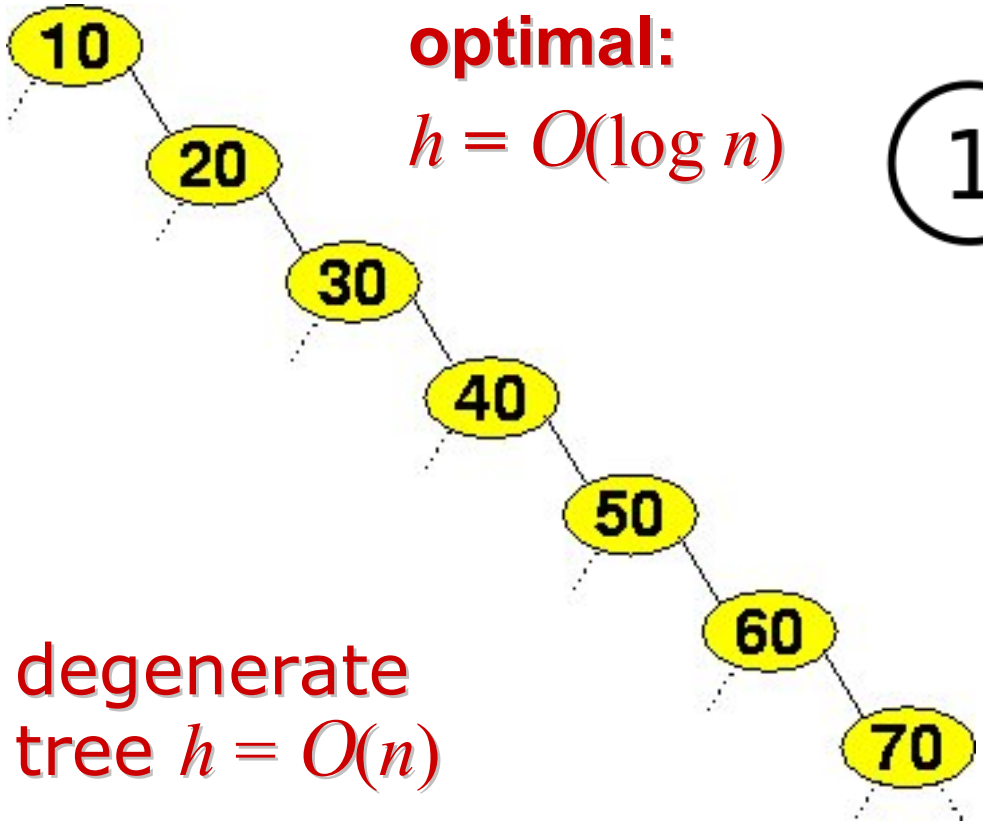
- more in Chapter 5



Binary Trees

Lemma: $n \leq 2^{h+1} - 1$

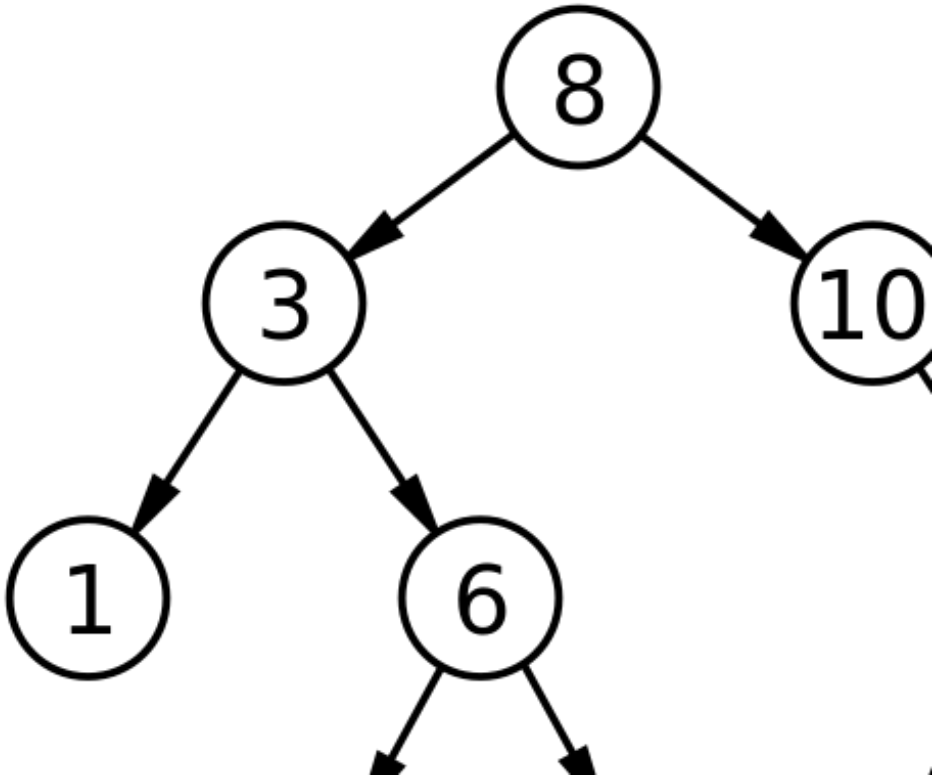
$\Rightarrow \; h \geq O(\log n)$



optimal:

$h = O(\log n)$

degenerate tree $h = O(n)$

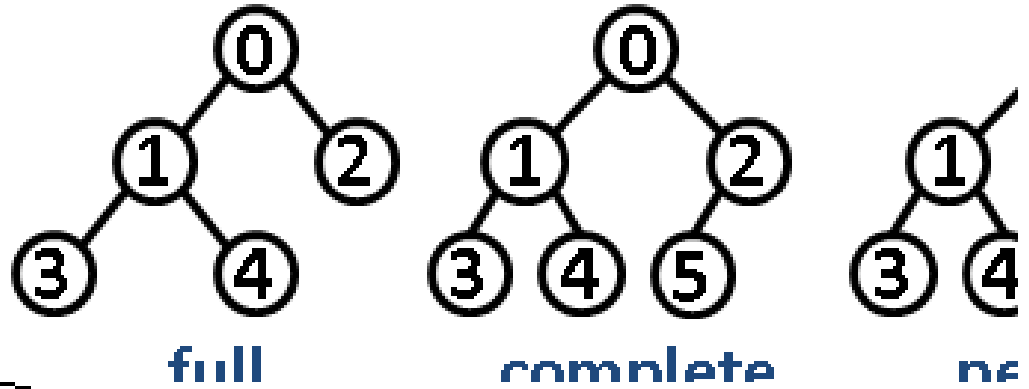


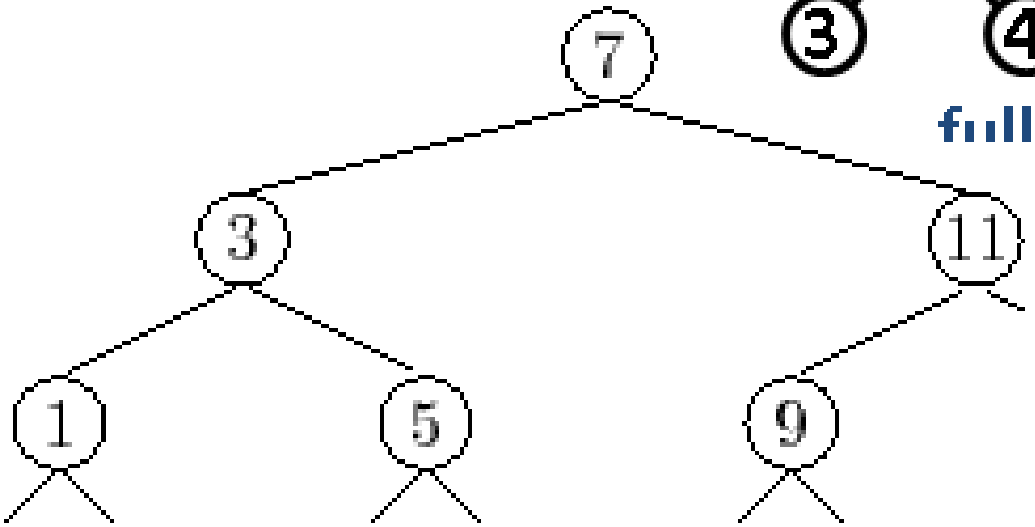
Binary **search** tree:

search, insert, delete

in $O(h)$, h = height

Balanced Binary Trees





every level, *except possibly the last*, is completely filled

Now **insert(15)**

and **delete(0):**

Rebalancing is costly!

Binary **search** tree:

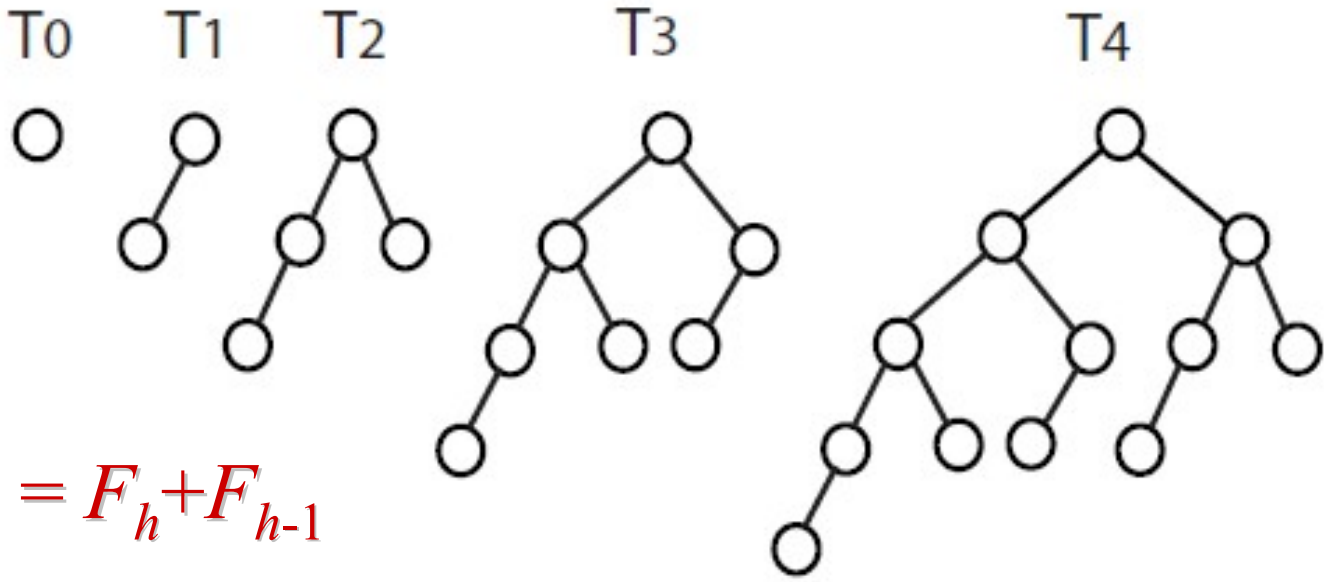
search, insert, delete

in $O(h)$, h = height

Adelson-Velsky-Landis'62

Binary tree s.t. *left and right subtrees have height difference at most 1!*

minimal AVLTrees:



Fibonacci

$F_0=0,\ F_1=1, F_{h+1} = F_h+F_{h-1}$

$n(h)$:=min #nodes of AVLTree of height $h \le \mathrm{O}(\log n)$

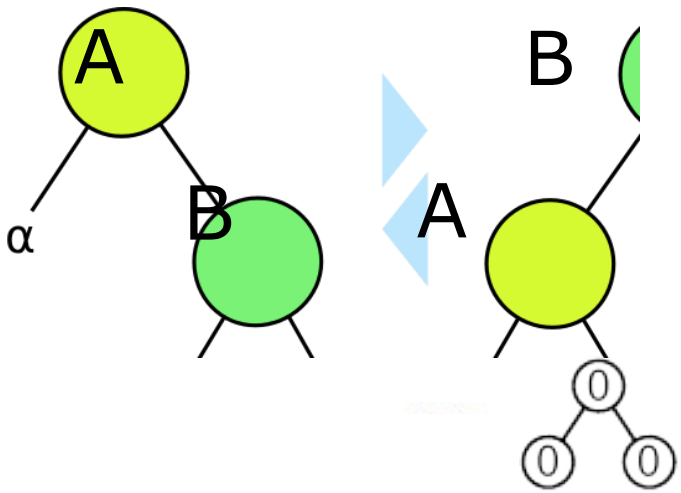
$\#n(0)+1=F_3,\quad \#n(h+1)+1=\#n(h)+1+\#n(h-1)+1 = F_{h+4}$

Recall $F_h = (\varphi^h-(-1/\varphi)^h)/\sqrt{5} \ge \Omega(1.6^h) \Rightarrow h=\mathrm{O}(\log F_h)$

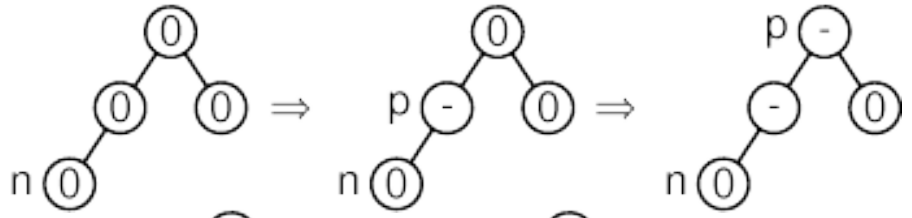
AVL Tree Maintenance

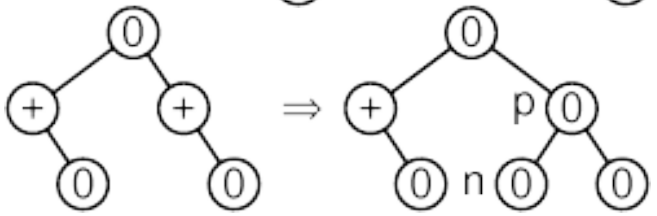
Binary tree s.t. *left and right subtrees have height difference at most 1!*

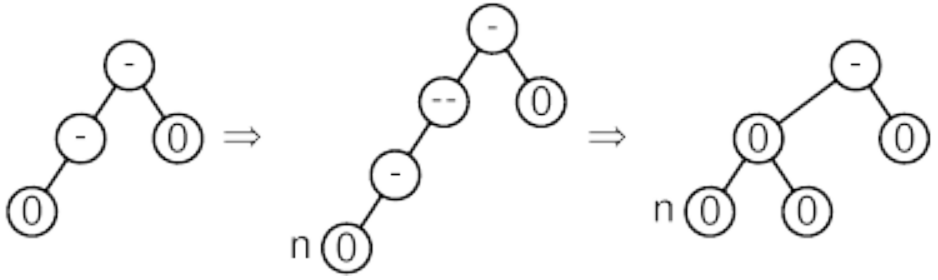
$$h \leq O(\log n)$$



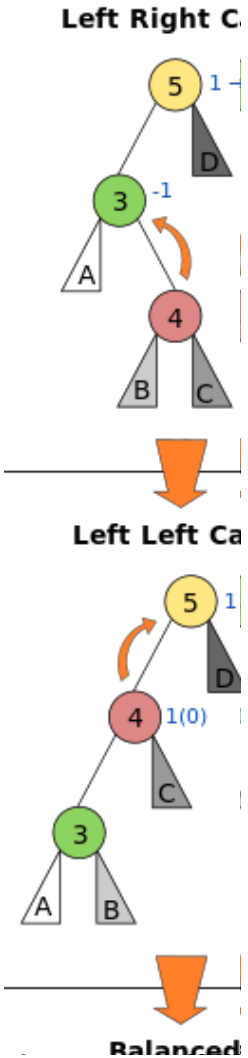
Store & recursively update balance indicators +, 0, –.
After **insert** at a left leaf, propagate up: three cases







search $O(\log n)$
insert $O(\log n)$
delete $O(\log n)$
merge $O(n)$



Recap

4. Intro Data Structures

- Hardware/Math Data Types
- Abstract Data Types
- Basic: Boolean, Integer
- Derived: Array, Queue, Stack
- Linked Data Structures
- (Balanced) Search Trees
- AVL Trees

- Specification
- Primitives: semantics and cost
- Design
- Analysis
- (Optimality)