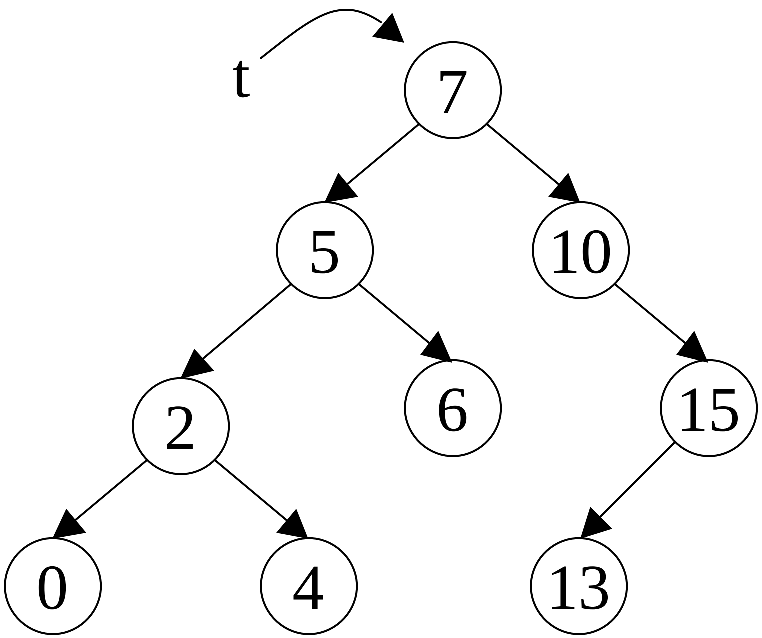
Order Statistics on Binary Trees

- Goal: find the k^{th} element (in order) of a binary tree

  where 1 <= k <= N

- Why?
  – Find the median: k = n/2, or k = $\lfloor n/2 \rfloor$ and $\lceil n/2 \rceil$
  – Find quartiles: k = n/4, n/2, 3n/4



t.size() → 9

t.elementAt(1) → 0 // min

t.elementAt(3) → 4 // 25%-ile

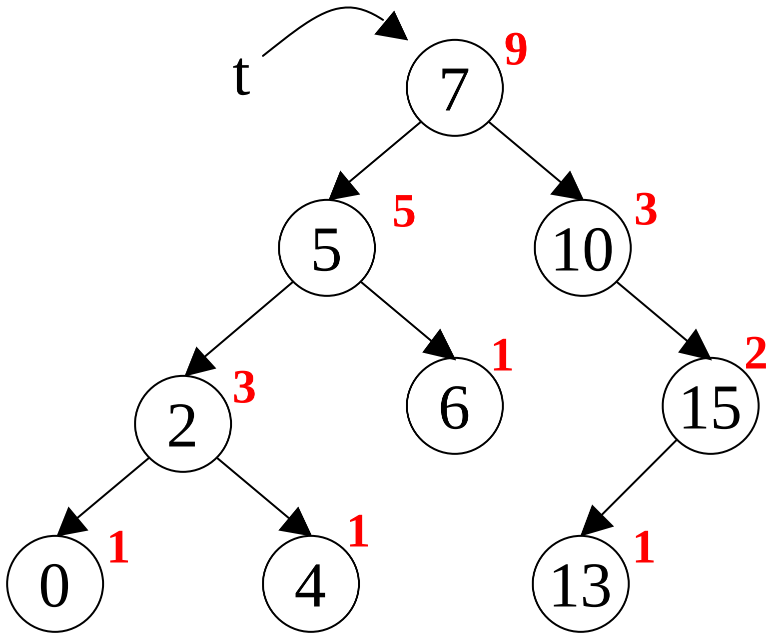
t.elementAt(5) → 6 // median

t.elementAt(7) → 10 // 75%-ile

t.elementAt(9) → 15 // max

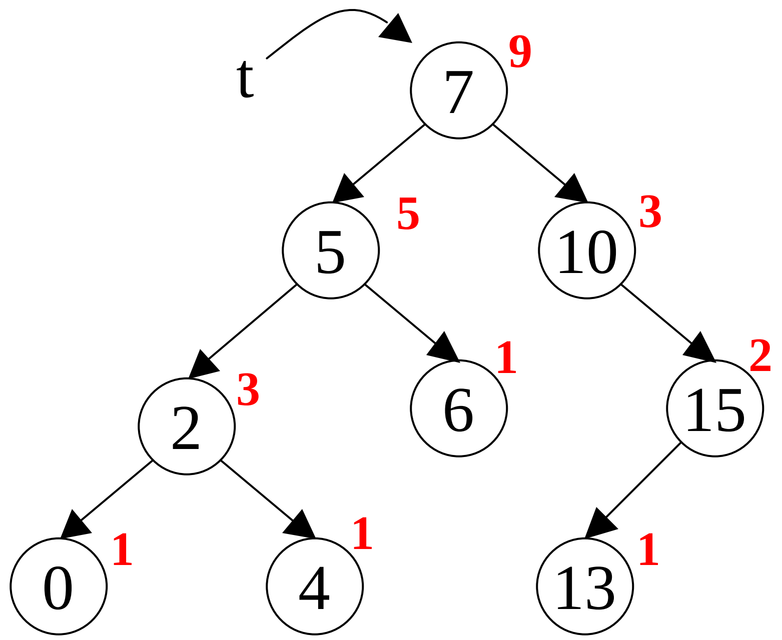
Maintaining Order Information

- Keep count tree sizes at each node
  - Update whenever tree is modified (easy)



Using Order Information

- Searching for k^{th} element:
  - If k == left.size + 1, return data
  - If k < left.size + 1, search for k^{th} element in left subtree
  - If K > left.size + 1, search for $(k\text{-}left.size\text{-}1)^{th}$ element in right sub-tree



Structural Induction

- We saw induction on natural numbers:
  Proof that property P holds for all natural numbers *n*
  - base case: P(0)
  - inductive hypothesis: Suppose P(*n*) holds
  - inductive case: P(*n* + 1)
- Natural numbers have an inductive definition:
  - base case: **0 is a natural number**
  - inductive case: **if *n* is a natural number, so is *n* + 1**
  - (and nothing else is a natural number, except those things created using these two rules)
- Coincidence?

Structural Induction

- We can generalize this to other *structures* that have an inductive definition
- E.g., a *full binary tree* can be defined as follows:
  - base case: **make_tree(data) is a f-b-tree** (consisting of one leaf node having given data)
  - inductive case: **if left and right are f-b-trees, so is make_tree(left, right, data)** (consisting of one internal node having given data and given left and right subtrees)
  - (and nothing else is a f-b-tree except those things created using these two rules)
  - (in particular, can't have empty f-b-tree)
- How to do induction on these things?

Tree Induction

- Prove the following property for every f-b-tree
  P(t) : # leaf nodes in t = # internal nodes in t + 1
- Base case: P(make_tree(data))

- Inductive Hypothesis: Suppose P(*left*) and P(*right*) hold

- Inductive Case: P(make_tree(left, right, data))

Structural Induction

- Now we can prove (inductively) properties about all sorts of (inductively defined) things:
  - Natural Numbers, Rational Numbers, Real Numbers
  - Trees, BSTs, Linked Lists, Doubly-linked lists, etc.
  - Expressions (e.g. E : integer | E + E)
  - Java programs, methods, classes, etc.
- Ain't induction grand?