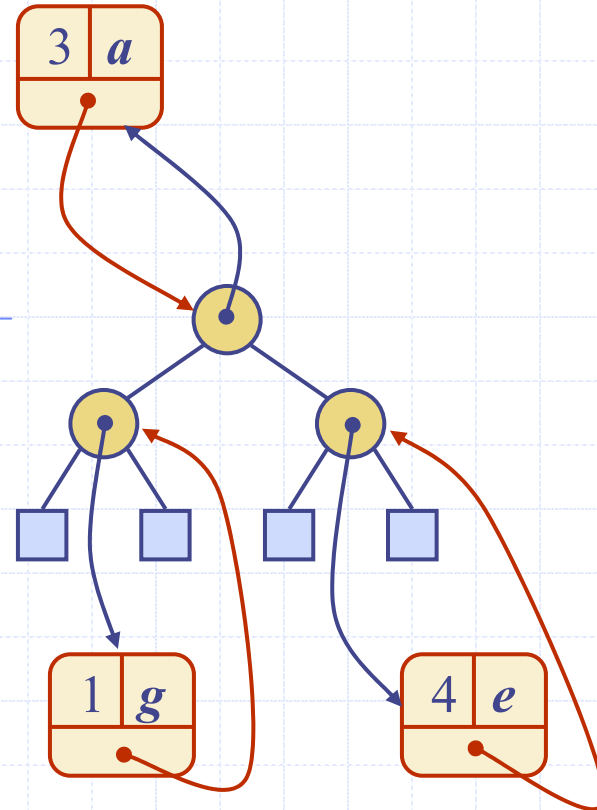


# Locators



# Outline and Reading

- ◆ Locators (§2.4.4)
- ◆ Locator-based methods (§2.4.4)
- ◆ Implementation
- ◆ Positions vs. Locators

# Locators

- ◆ A locator identifies and tracks a (key, element) item within a data structure
- ◆ A locator sticks with a specific item, even if that element changes its position in the data structure
- ◆ Intuitive notion:
  - claim check
  - reservation number
- ◆ Methods of the locator ADT:
  - **key()**: returns the key of the item associated with the locator
  - **element()**: returns the element of the item associated with the locator
- ◆ Application example:
  - Orders to purchase and sell a given stock are stored in two priority queues (sell orders and buy orders)
    - ◆ the key of an order is the price
    - ◆ the element is the number of shares
  - When an order is placed, a locator to it is returned
  - Given a locator, an order can be canceled or modified

# Locator-based Methods

## ◆ Locator-based priority queue methods:

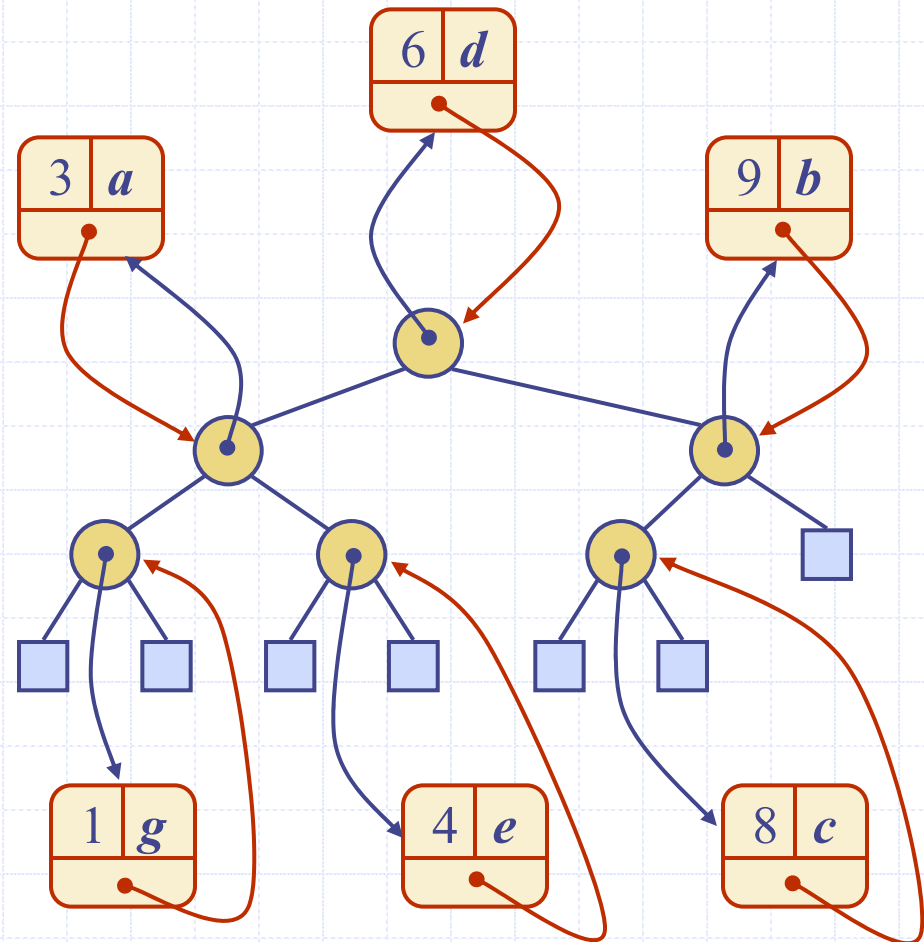
- **insert**(k, o): inserts the item (k, o) and returns a locator for it
- **min**(): returns the locator of an item with smallest key
- **remove**(l): remove the item with locator l
- **replaceKey**(l, k): replaces the key of the item with locator l
- **replaceElement**(l, o): replaces with o the element of the item with locator l

## ◆ Locator-based dictionary methods:

- **locators**(): returns an iterator over the locators of the items in the priority queue
- **insert**(k, o): inserts the item (k, o) and returns its locator
- **find**(k): if the dictionary contains an item with key k, returns its locator, else return the special locator NO\_SUCH\_KEY
- **remove**(l): removes the item with locator l and returns its element
- **locators**(), **replaceKey**(l, k), **replaceElement**(l, o)

# Implementation

- ◆ The locator is an object storing
  - key
  - element
  - position (or rank) of the item in the underlying structure
- ◆ In turn, the position (or array cell) stores the locator
- ◆ Example:
  - binary search tree with locators



# Positions vs. Locators

## ◆ Position

- represents a “place” in a data structure
- related to other positions in the data structure (e.g., previous/next or parent/child)
- implemented as a node or an array cell

## ◆ Position-based ADTs (e.g., sequence and tree) are fundamental data storage schemes

## ◆ Locator

- identifies and tracks a (key, element) item
- unrelated to other locators in the data structure
- implemented as an object storing the item and its position in the underlying structure

## ◆ Key-based ADTs (e.g., priority queue and dictionary) can be augmented with locator-based methods