ITEC201006

Introduction to Computer Science & Engineering

Lecture 7: Abstract Data Types and Subprograms

Jeonghun Park

KYUNGPOOK NATIONAL UNIVERSITY

Abstract Data Types

- Two logical implementations of containers:
- Array-based implementation
  - Objects in the container are kept in an array
- Linked-based implementation
  - Objects in the container are not kept physically together, but each item tells you where to go to get the next one in the structure

Linked-based Implementation

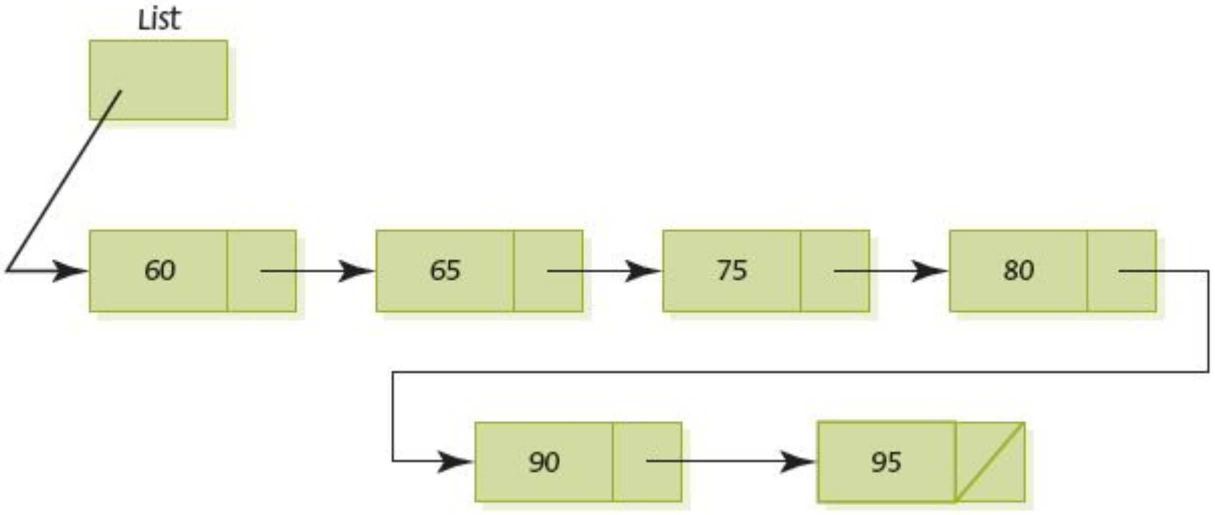


FIGURE 8.4 A sorted linked list

Array-based implementation

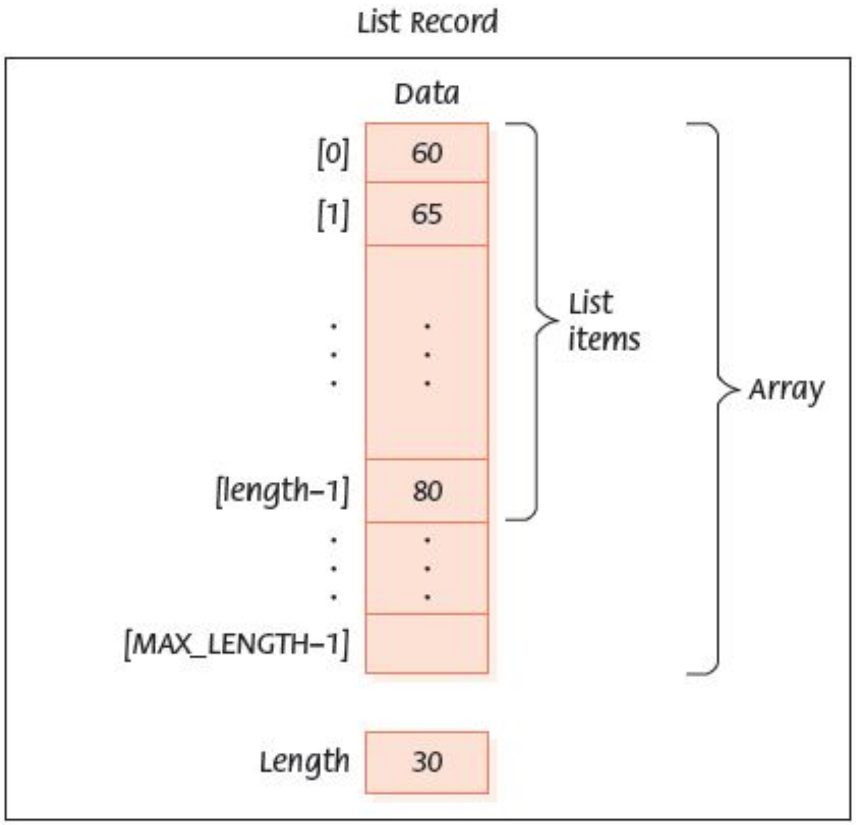


FIGURE 8.3 A sorted list of integers

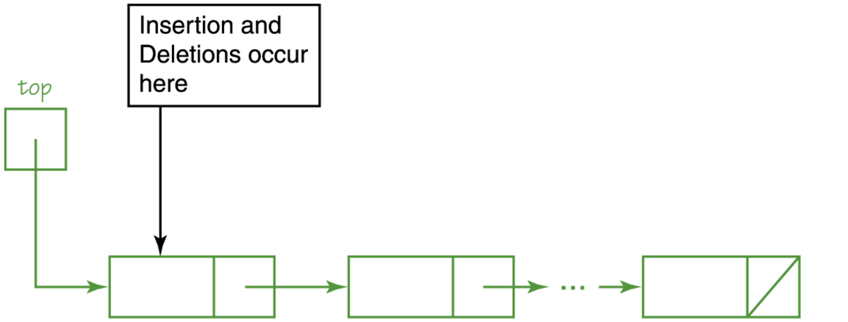
Stacks

- An abstract data type in which accesses are made at only one end
  - LIFO, which stands for Last In First Out
  - The insert is called ***Push*** and the delete is called ***Pop***

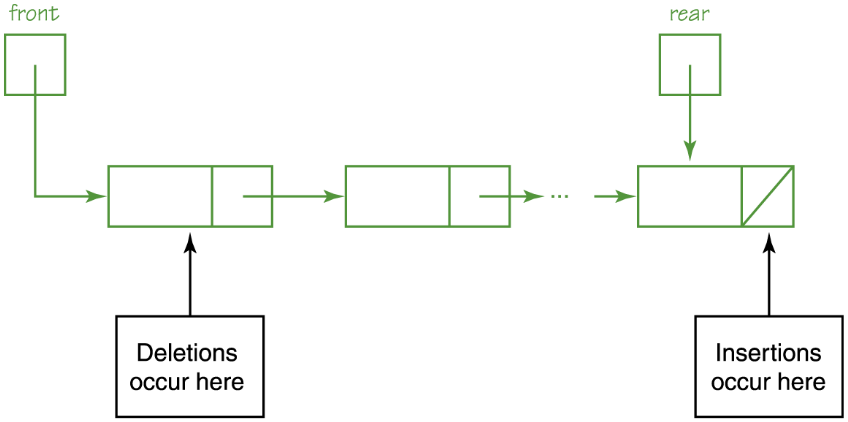
Queues

- An abstract data type in which items are entered at one end and removed from the other end
    - FIFO, for First In First Out

Comparison



(a) A linked stack



(b) A linked queue

List

- Think of a list as a container of items
- Here are the logical operations that can be applied to lists
  - *Add item* Put an item into the list
  - *Remove item* Remove an item from the list
  - *Get next item* Get (look) at the next item
  - more items Are there more items?

Tree

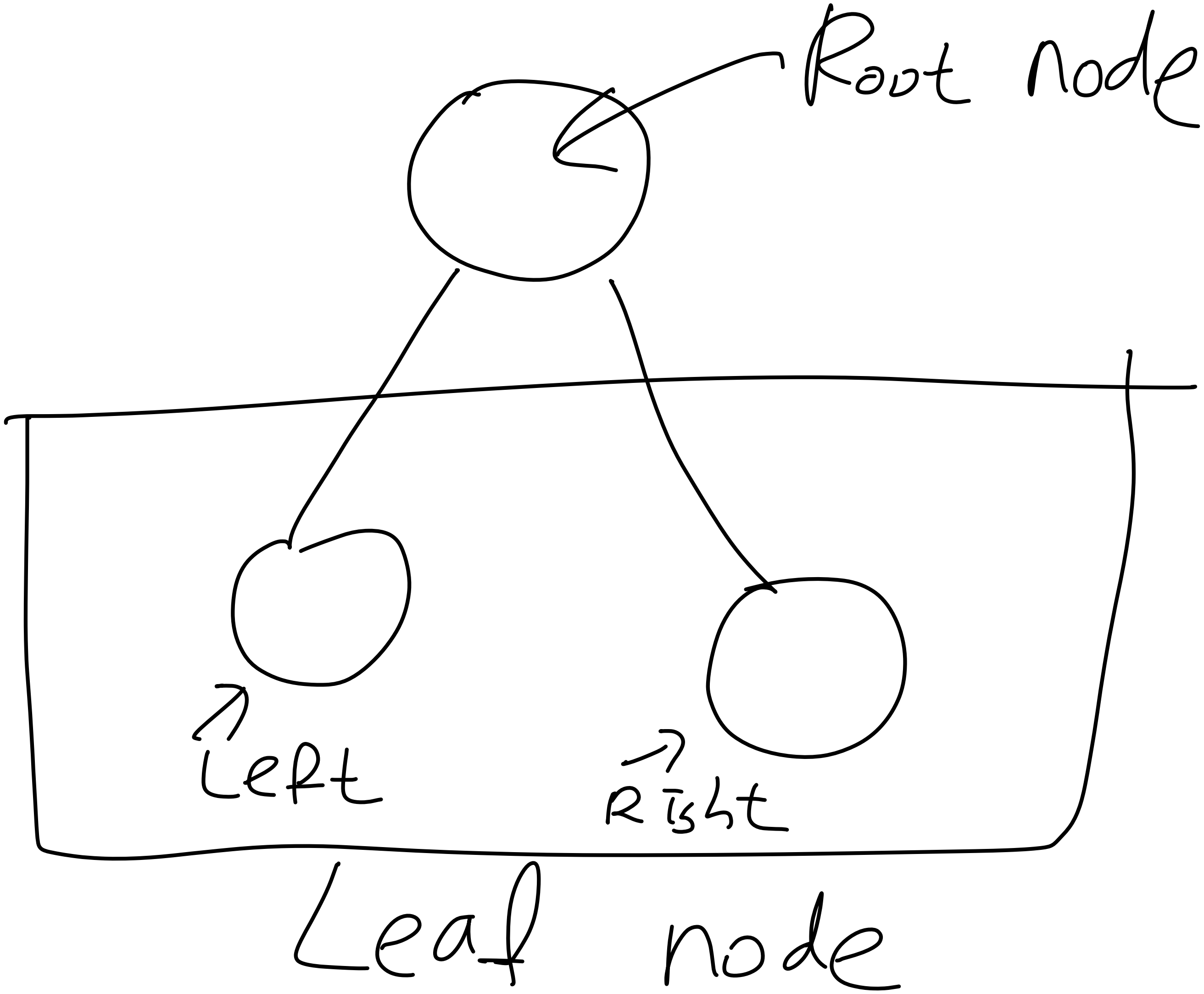
- Structure such as lists, stacks, and queues are linear in nature; only one relationship is being modeled
- More complex relationships require more complex structures
- Can you name three more complex relationships?

Binary Tree

- A linked container with a unique starting node called the **root**, in which each node is capable of having **two child nodes**, and in which a unique path (series of nodes) exists from the root to every other node

Binary Tree

- What is ..
  - Root node
  - Leaf node
  - Right child
  - Left child



Binary Tree Search

- Binary tree search (BTS)
  - A binary tree (*shape property*) that has the (*semantic*) property that characterizes the values in a node of a tree
  - We already know some examples

Binary Tree Search

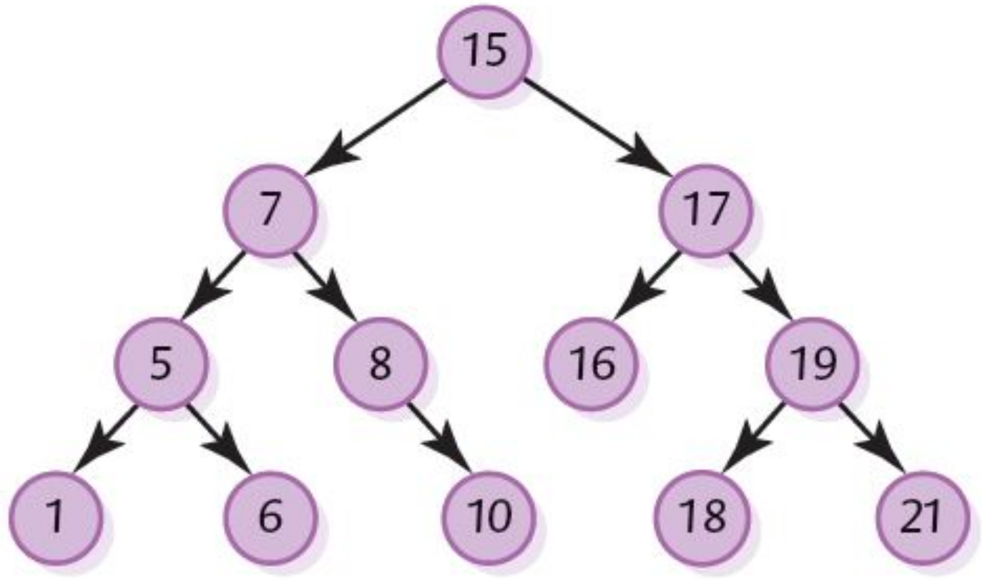
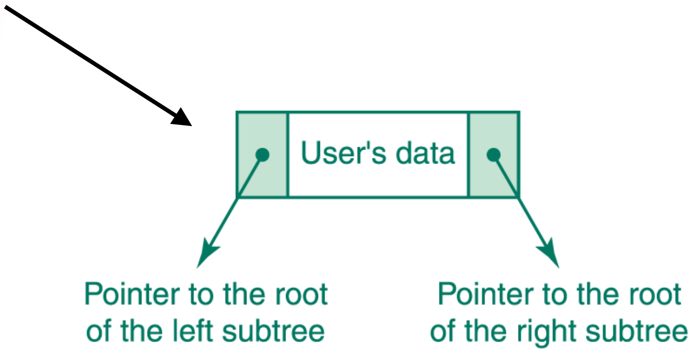


FIGURE 8.7 A binary search tree


User's data
Pointer to the root of the left subtree
Pointer to the root of the right subtree

Binary Tree Search

```
Boolean IsThere(current, item)
    If (current is null)
        return false
    Else
        Set result to item.compareTo(info(current))
        If (result is equal to 0)
            return true
        Else
            If (result < 0)
                IsThere(item, left(current))
            Else
                IsThere(item, right(current))
```

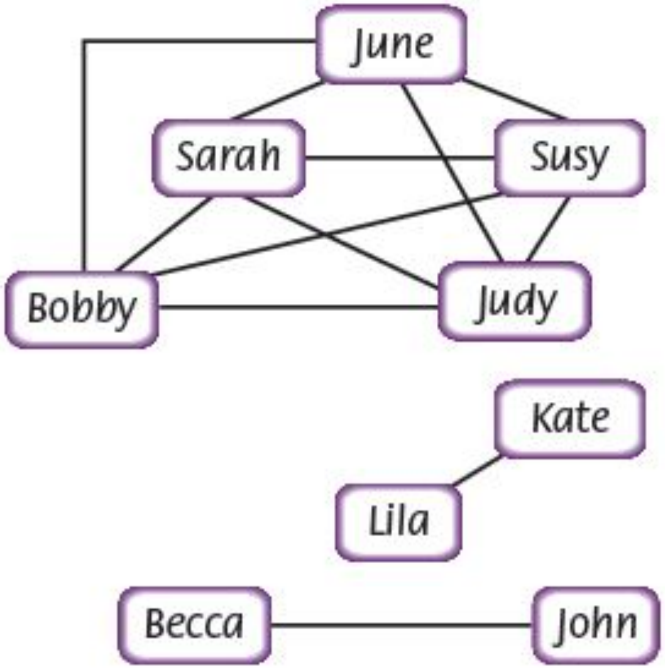
Binary Tree Search

```
IsThere(tree, item)
IF (tree is null)
    RETURN FALSE
ELSE
    IF (item equals info(tree))
        RETURN TRUE
    ELSE
        IF (item < info(tree))
             IsThere(left(tree), item)
        ELSE
             IsThere(right(tree), item)
```

Graph

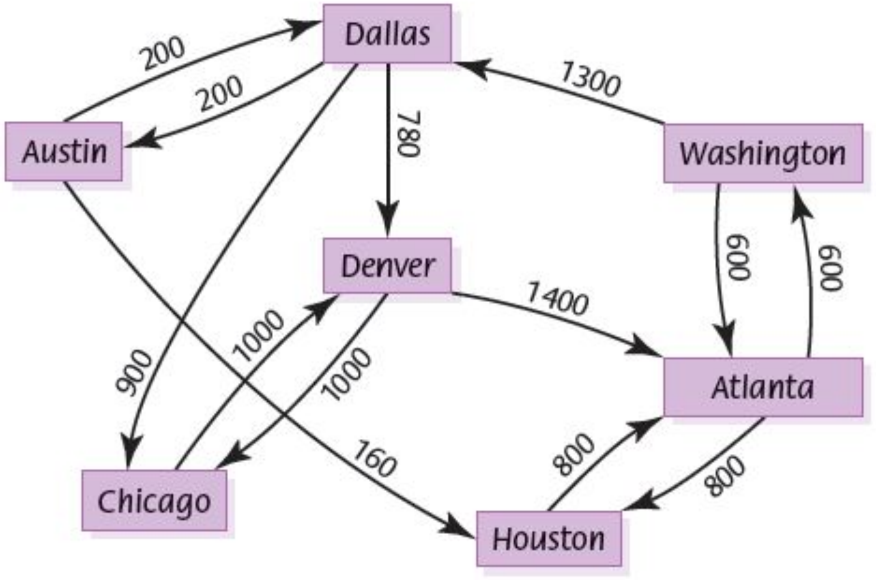
- Graph
  - A data structure that consists of a set of nodes (called vertices) and a set of edges that relate the nodes to each other
- Undirected graph
  - A graph in which the edges have no direction
- Directed graph (Digraph)
  - A graph in which each edge is directed from one vertex to another (or the same) vertex

Graph Example



(a) Vertices: People
Edges: Siblings

Graph Example



(b) Vertices: Cities
Edges: Direct flights