course_content

top



Data Structures and Algorithms

## Sorting Review & Intro Trees

# Week 8

Time:
5 min

# Quiz Time

bit.ly/DSA1920Quiz7

Time:
5 min

# Sorting Review - Stability

- Stable Sorts - two identical elements maintain their existing order
- We want stable sorts when there are multiple properties to sort on
- Any algorithm that **swaps elements that are not near each other** can **break stability** e.g. Selection Sort, Shell Sort, Heap Sort, Quick Sort
- Any sorting algorithm can be made to be stable with O(n) space by including the current position in the comparison (when comparing two equal elements)

Time:
5 min

# Sorting Review

| Algorithm | Time-Best | Time-Avg | Time-Worst | Aux Space | In-place | Stable |
|---|---|---|---|---|---|---|
| Bubble | O(n) | $O(n^2)$ | $O(n^2)$ | O(1) | Yes | Yes |
| Selection | $O(n^2)$ | $O(n^2)$ | $O(n^2)$ | O(1) | Yes | No |
| Insertion | O(n) | $O(n^2)$ | $O(n^2)$ | O(1) | Yes | Yes |
| Shell | O(nlogn) | O(nlogn) | $O(n^2)$ | O(1) | Yes | No |
| Merge | O(nlogn) | O(nlogn) | O(nlogn) | O(n) | No | Yes |
| Quick | O(n) | O(nlogn) | $O(n^2)$ | O(log n) | Yes/No | No |
| Heap | O(nlogn) | O(nlogn) | O(nlogn) | O(1) | Yes | No |

Time:
10 min

# MidTerm Review

- What is the worst-case runtime for searching a sorted array?
- What is the worst-case runtime for deleting a node from a doubly-linked list?
- What is the worst-case runtime for accessing a hash table? (no collisions)
- Definition of ‘x and y’ and ‘x or y’ in Python.
- Mutable types in Python
- Number of Operations → Time Complexity

Time:
10 min

# Trees

- A tree is a **non-linear** data structure
- It has a **root, branches** and **leaves**
- The **root** (only one) is at the **top,** branches come down with leaves at the bottom
- Each item in a tree is called a **node**
- Examples: Animal Kingdom Hierarchy, Family Tree, File System, XML/HTML data, Heaps, compilers (for managing syntax),

Time:
10 min

# Trees

- What do the following mean in the context of trees?
  - Node, Edge, Root, Path, Children, Parent
  - Sibling, Subtree, Leaf Node, Level, Height
- You can think of a tree in 2 ways:
  - A root + one-way edges coming out of each node to new nodes such that there are no ‘cycles’ ---- the intuitive understanding
  - A tree is either empty or a root connecting to zero or more subtrees (which are also trees) ----- the recursive definition

Time:
10 min

# Trees

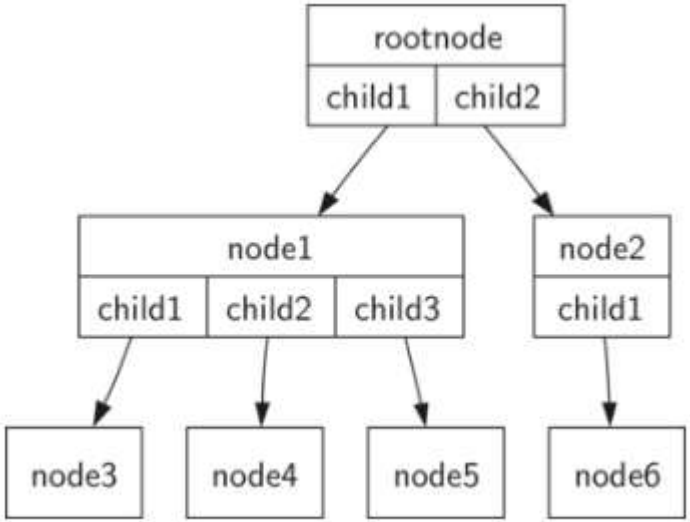


Figure 3: A Tree Consisting of a Set of Nodes and Edges

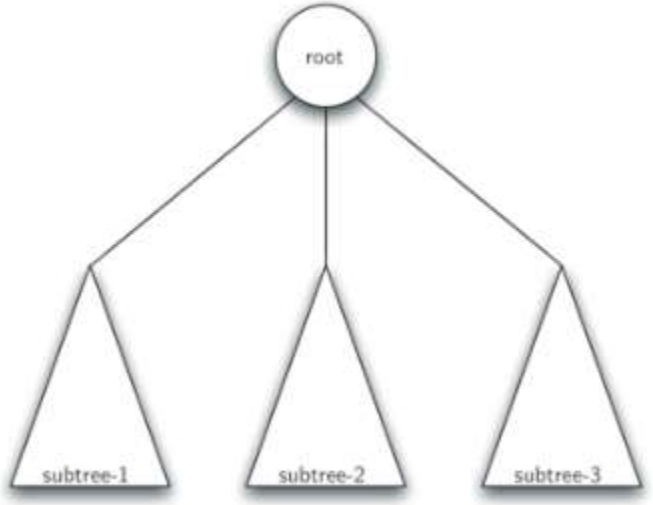


Figure 4: A recursive Definition of a tree

Time:
5 min

# Trees - Implementation

- 2 main ways of implementing Trees
  - List of Lists
    - E.g. [3,[1],[2,[1],[3]]]
  - Node and list of children
    - Think of Linked Lists but instead of just one next node, you have a list of children nodes
- Binary Trees have at most 2 children - left Node and right Node (can be Empty)

Time:
5 min

# Trees - Traversals

- There are many different traversal methods:
  - Depth-First Search (method)
    - Pre-order traversal
    - In-order traversal
    - Post-order traversal
  - Breadth-First Search (method)
- What do they mean? How do they work?

Time:
1 min

# Questions?