CSC 220 Data Structures

Trees I – Definitions and Abstractions
Parkland College Fall 2016
20161003
Trees!

- non-linear, hierarchical data structures
- lots of associated vocabulary
root leaves/external nodes

node

element's parent

element's child

ancestors, depth 4

descendants, height 4

edge

internal nodes

Path BACDG

D's subtree

siblings

E's parent

E's child

4 ancestors, depth 4

4 descendants, height 4

siblings

leaves/external nodes
Abstract Tree

- `tree.py`
- **main job** – provide interface for concrete trees
- Recursion used extensively
  - Base case: a particular node
  - Otherwise recurse on children or parent, depending on particular algorithm
- `depth()`
- `_height1()` *vs* `_height2()`
Binary Trees

• 0, 1, or 2 children/node only
• *proper/full* – every parent has 2 children
• *improper* – not every parent has 2 children
• Recursive definition:
  • base case: tree is empty
  • otherwise: node $r$ has left and right binary subtrees
    • (note: these are possibly empty subtrees)
Arithmetic

• Infix binary tree

(\(((5 + 9) * 7) / ((13 - 7) + 2)) - ((8 * (2 - 7)) + 6)\)

\[5 \cdot 9 + 7 \cdot 13 - 2 + / 2 \cdot 7 - 8 \cdot 6 + -\]
Binary Tree Properties

- For both proper and improper (h = height)
  - $h + 1 \leq n \leq 2^{h+1} - 1$
  - $1 \leq \text{leaves} \leq 2^h$
  - $h \leq \text{non-leaves} \leq 2^h - 1$
  - $\log(n + 1) - 1 \leq h \leq n - 1$

- For just proper
  - $2^h + 1 \leq n \leq 2^{h+1} - 1$
  - $h + 1 \leq \text{leaves} \leq 2^h$
  - $h \leq \text{non-leaves} \leq 2^h - 1$
  - $\log(n+1) - 1 \leq \text{height} \leq (n-1)/2$
  - non-leaves + 1 = leaves
Binary tree implementations

- `simpletree.py`
  - just to get the idea down
- `binary_tree.py`
  - more complete, uses `Tree.Position` objects
- `linked_binary_tree.py`
  - see complexity chart in notes
- see array-based binary tree example in notes