# CSCI 136
# Data Structures &
# Advanced Programming

Bill Jannen

Lecture 25

April 17, 2017

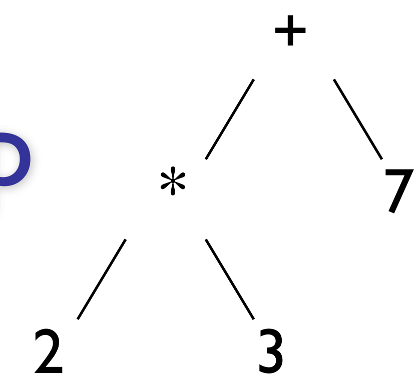
# Administrative Details

- Taxes due tomorrow
- Lab 8 posted – Super Lexicon!
  - Read through it and plan your designs
  - Look for updates posted (starter files and hints)
- Morgan is gone for conference travel
  - Jon and Bill are here to answer questions

# Last Time

- Binary Trees
  - Finished discussing tree traversal methods and iterators
    - In-order, post-order, level-order, priority-order…
    - DFS, BFS

# Tree Traversal Recap




- Pre-order: +*237
  - Each node is visited before any children. Visit node, then each node in left subtree, then each node in right subtree.
- In-order: 2*3+7
  - Each node is visited after all nodes in left subtree are visited and before any nodes in right subtree.
- Post-order: 23*7+
  - Each node is visited after its children are visited. Visit all nodes in left subtree, then all nodes in right subtree, then node itself.
- Level-order: +*723
  - All nodes of level i are visited before nodes of level i+1.

# Tree Search Strategies

- Two main approaches
  - Breadth-first search (BFS)
    - Search across tree before searching down to another level
    - Level-order traversal
  - Depth-first search (DFS)
    - Search down tree (to leaf) before search across tree
    - Pre-order traversal
  - DFS is more efficient if solution is "far away" from root (i.e., many edges between root and solution)
  - Unix `grep` scans file system in BFS

# Today's Outline

- Cool tree application: Huffman Coding
- Alternative tree representation
- Quick Trie Description for Lab

# Representing Strings

- How many bits to represent a character?
  - Often 8 bits (1 byte)
- If so, how many bits to represent the string:

AN ANTARTCTIC PENGUIN

  - 20 characters * 8 bits = 160 bits

# Huffman Codes

- We can compress the representation of some data when the distribution of 1's and 0's is non-uniform
- General idea
  - Use less bits for most common letters

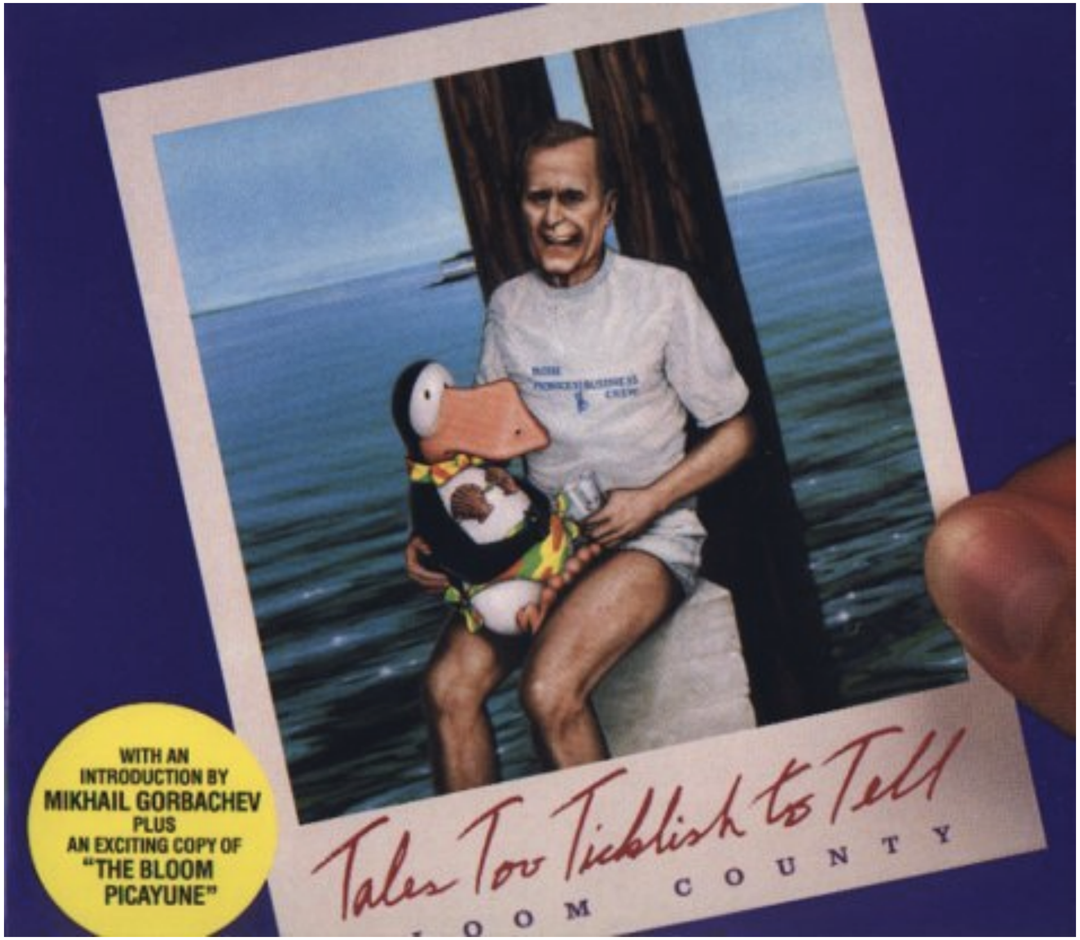
    AN ANTARCTIC PENGUIN

  - Compute letter frequencies

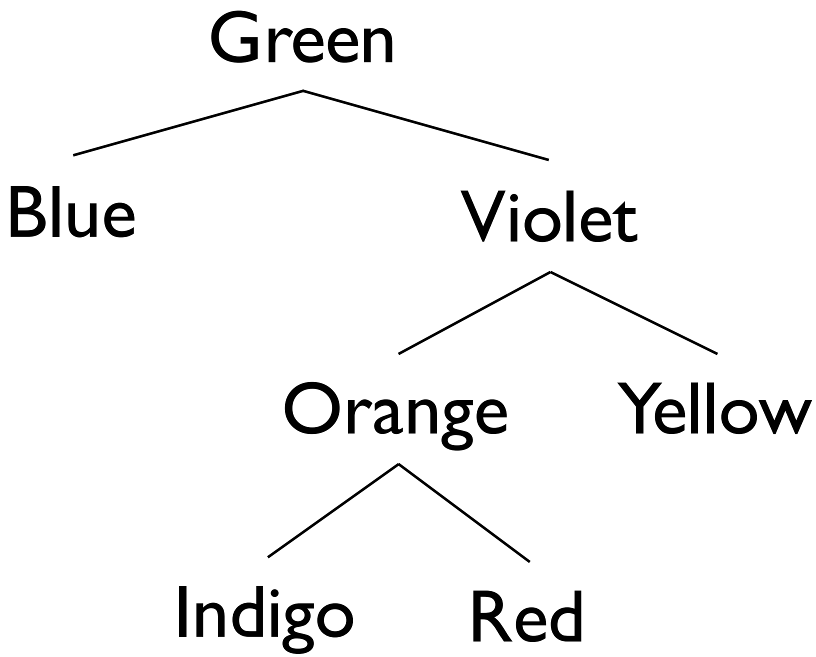
    ```
    A: 3        N: 4
    T: 2        R: 1
    C: 2        I: 2
    P: 1        E: 1
    G: 1        U: 1
    _: 2
    ```

  - Build tree by recursively creating trees of smallest weighted components
  - Result: 67 bits

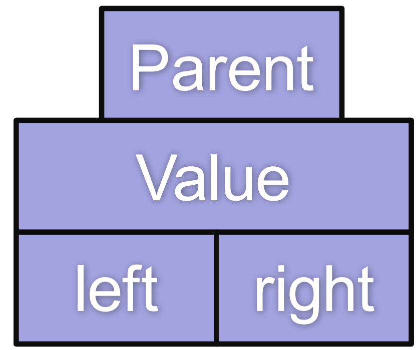
# Huffman Example 2



# Alternative Tree Representations




- Consider Ch 12 Tree class
- Total # "slots" = 4n

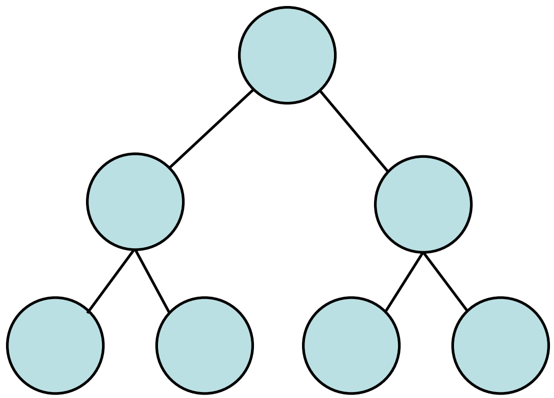



- Compare that to a vector, SLL, array, …
- But trees capture successor and predecessor relationships that other data structures don't…

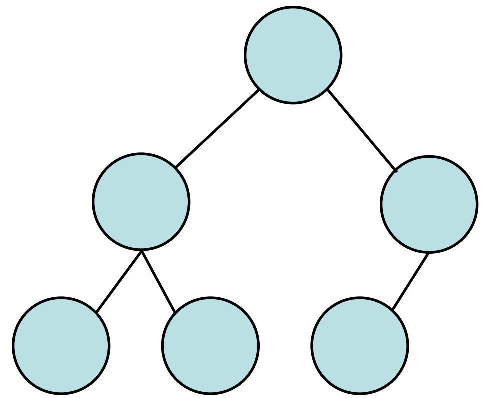
# Using Arrays to Store Trees

- Implicitly encode tree structure using indexes:
  - Consider a **full** tree
  - Index nodes as in level-order traversal

# Full vs. Complete Binary Trees



**Full binary tree:** Every non-leaf node has 2 children

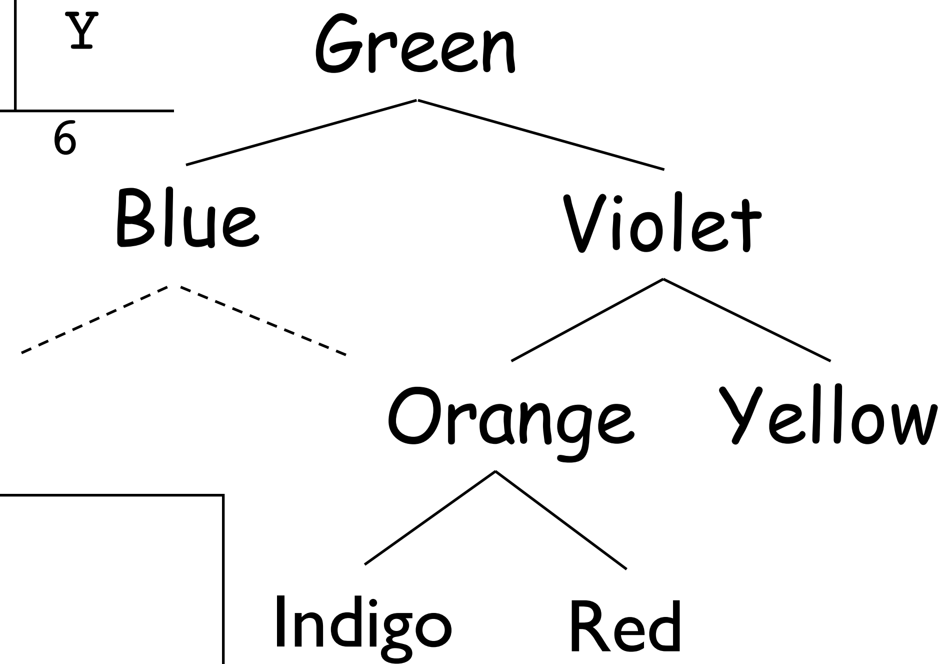


**Complete binary tree:** with the exception of the last level, all levels are completely filled, and all nodes are as far left as possible.

# Using Arrays to Store Trees

- Implicitly encode tree structure using indexes:
  - Consider a **full** tree
  - Index nodes as in level-order traversal
- Where are children of node i?
  - Children of node i are at 2i+1 and 2i+2
- Where is parent of node j?
  - Parent of node j is at (j-1)/2

| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| G | B | V | _ | _ | O | Y |




```
public class ArrayTree {
    protected Object[] data;

    protected int left(int node) {
        return 2*node+1;
    }

    protected int parent(int node) {
        return (node-1)/2;
    }

    …
}
```

# ArrayTree Tradeoffs

- Why are ArrayTrees good?
  - Save space for links (no "slots" needed)
    - Relationships between values are implicitly stored (index + math)
  - Works well for full or complete trees
    - Complete: All levels except last are full and all gaps are at right
    - "A *complete* binary tree of height h is a full binary tree with 0 or more of the rightmost leaves of level h removed"
- Why bad?
  - Could waste a lot of space (sparse trees)
  - Height of n requires $2^{n+1}$-1 array slots even if only O(n) elements