



Data Structures & Algorithms: Trees

Code Fellows

Data Structures

Abstracted representation of reality

- Build a simulation of reality
- Include just what we need
- Leave out unnecessary details
- Some general solutions are repeatedly useful:
 - Strings, Arrays, key-value pair sets
 - Linked lists, Stacks, Queues
 - Graphs, Trees
 - Custom object constructors



CODE FELLOWS

```
[ (class-14) > tree
.
├── LICENSE
└── README.md
└── dist
    ├── admin.html
    └── data
        ├── hackerIpsum.json
        └── ipsumArticles.json
    ├── index.html
    ├── new.html
    └── package.json
    └── scripts
        ├── aboutController.js
        ├── article.js
        ├── articleController.js
        ├── articleView.js
        ├── githubToken.js
        ├── repo.js
        ├── repoView.js
        └── routes.js
        └── webdb.js
    └── server.js
    └── styles
        ├── base.css
        └── fonts
            ├── icomoon.eot
            ├── icomoon.svg
            ├── icomoon.ttf
            └── icomoon.woff
        ├── icons.css
        ├── layout.css
        └── modules.css
    └── vendor
        └── scripts
            ├── handlebars.js
            ├── highlight.pack.js
            ├── html5sql.js
            ├── jquery-2.1.4.js
            └── marked.js
```

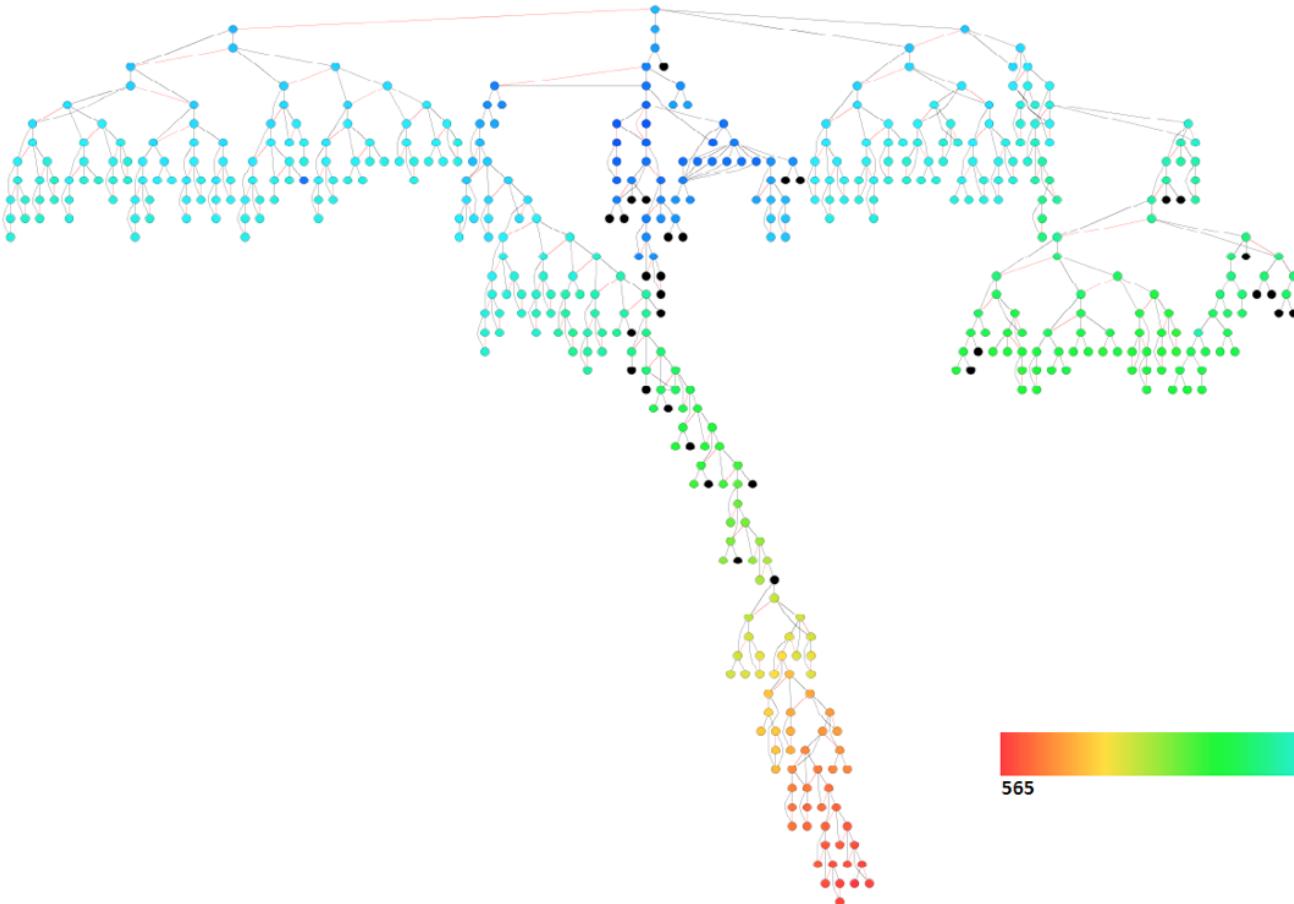
Abstraction





Reality





Abstraction





Reality



```
1 <!DOCTYPE HTML>
2 <html>
3   <head>
4     <title>Chocolate Pizza</title>
5     <link rel="stylesheet" type="text/css" href="style.css" />
6   </head>
7   <body>
8     <div>
9       <header>
10      <div class="all-things">
11        <div class="image-logo">
12          
13        </div>
14        <div id="tagline">
15          <h3><em>Delicious</em></h3>
16          <p class="slogan">THE BEST FOOD BLOG ON THE WEB</p>
17        </div>
18      </div>
19      <div class="line"></div>
20      <div class="all-images">
21        <div class="social-images">
22          
23          
24          
25          
26          
27          
28        </div>
29        <div class="rss-mail">
30          
31          
32        </div>
33      </div>
34    </header>
35    <section>
36      <h2>Chocolate Pizza</h2>
37      <div class="recipe-title">
38        <div class="posted-on">
```

Abstraction



Chocolate Pizza

POSTED ON 15 DEC 2013 / DESSERTS

[PRINT](#)



For the fig-swirl: Melt butter over medium heat in a saucepan. Add brown sugar and stir to dissolve. Halve all of the figs and toss in the saucepan with water and lemon juice. Cook over medium heat, stirring frequently, until you have a chunky-jammy mixture. Add salt with one or two stirs, set aside and let cool completely.

Reality

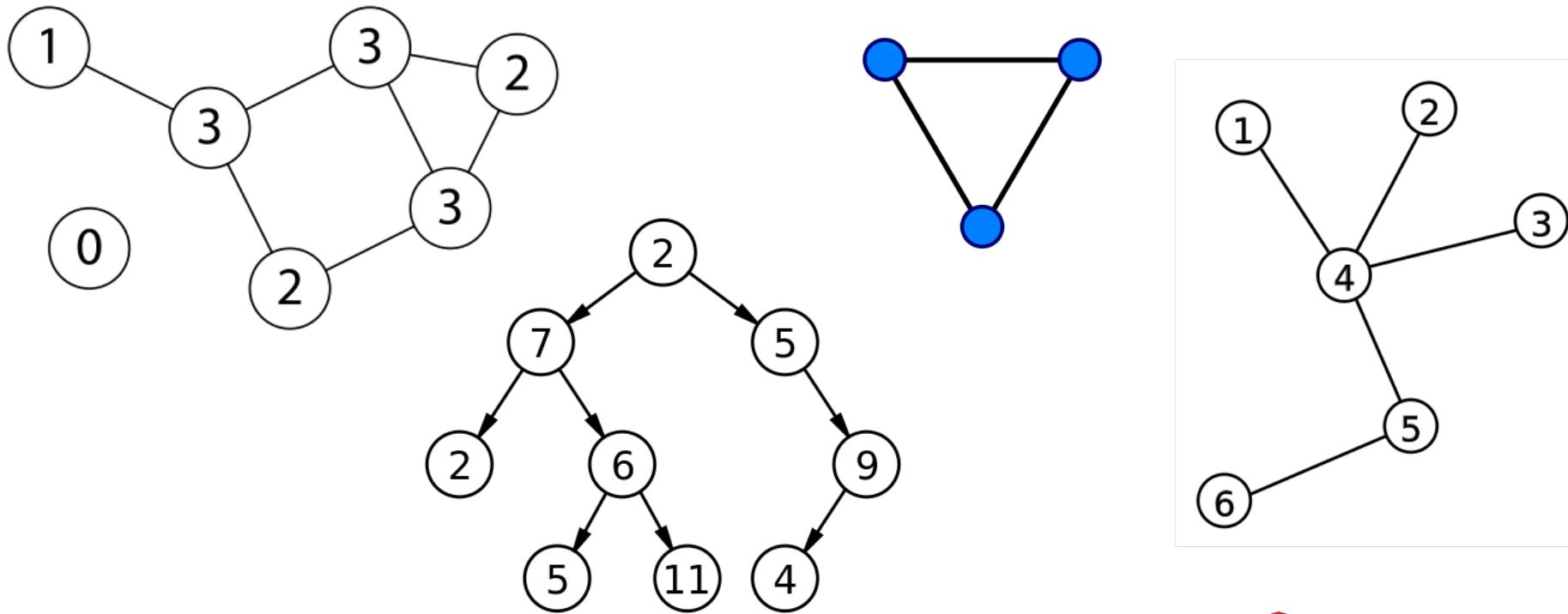


What is a Tree data structure?

- Graph:
 - a set of “nodes” and connections between nodes.
- Tree:
 - A directed, acyclic, connected graph.



What is a Tree data structure?



What is a Tree data structure?

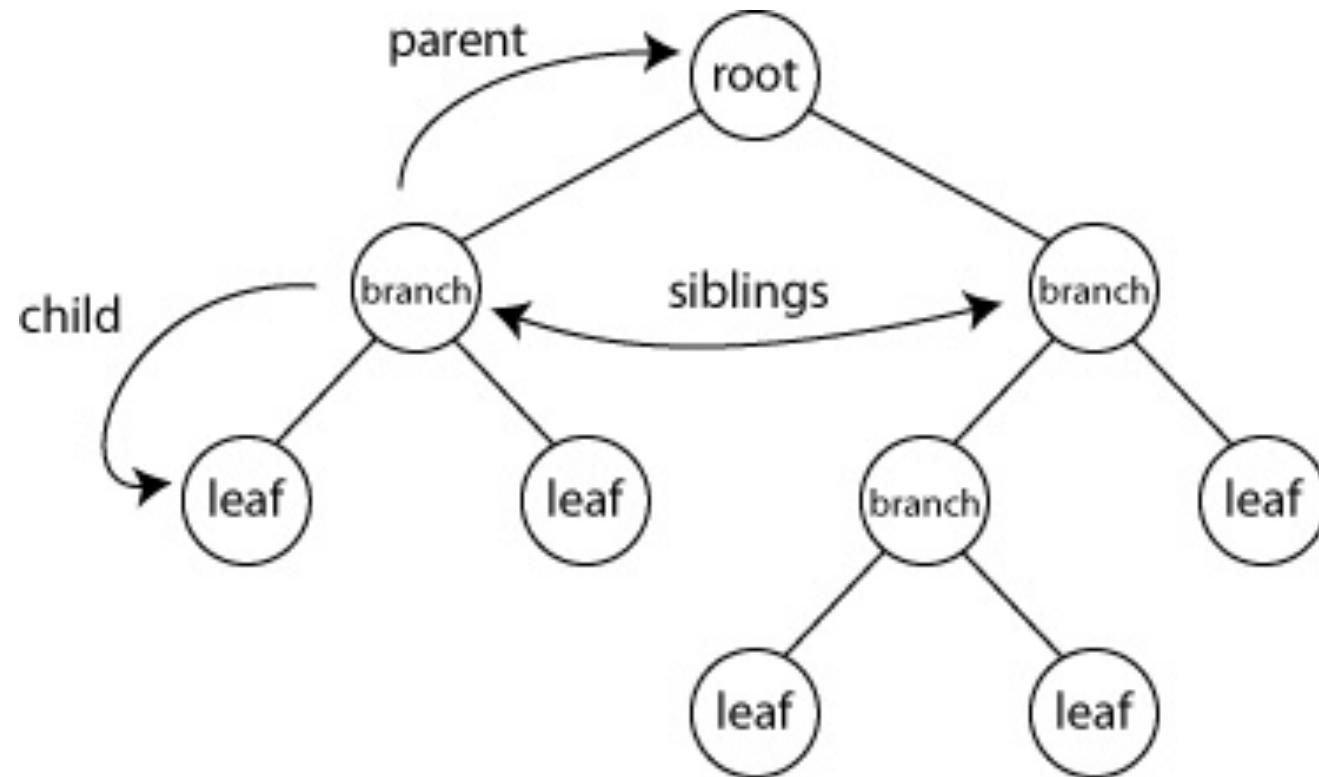
- Can be defined recursively
- Tree:
 - A value, and (possibly empty) list of trees
 - The sub-trees are called “children”



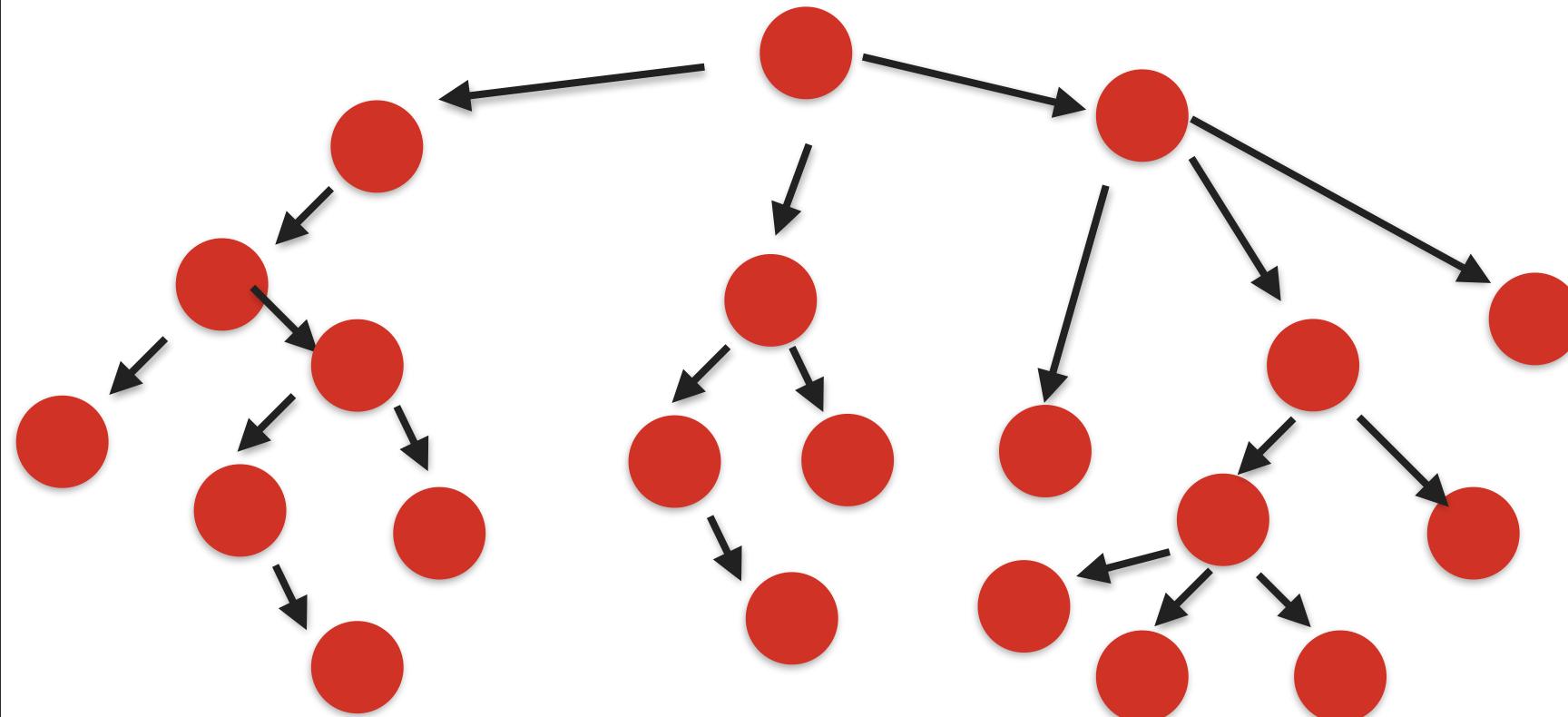
What is a Tree data structure?

- Root: The top node in a tree
- Child: A node that's a direct sub-tree of another node
- Siblings: A group of nodes with the same parent
- Leaf: A node with no children
- k-ary: A tree with at most k children in any one node

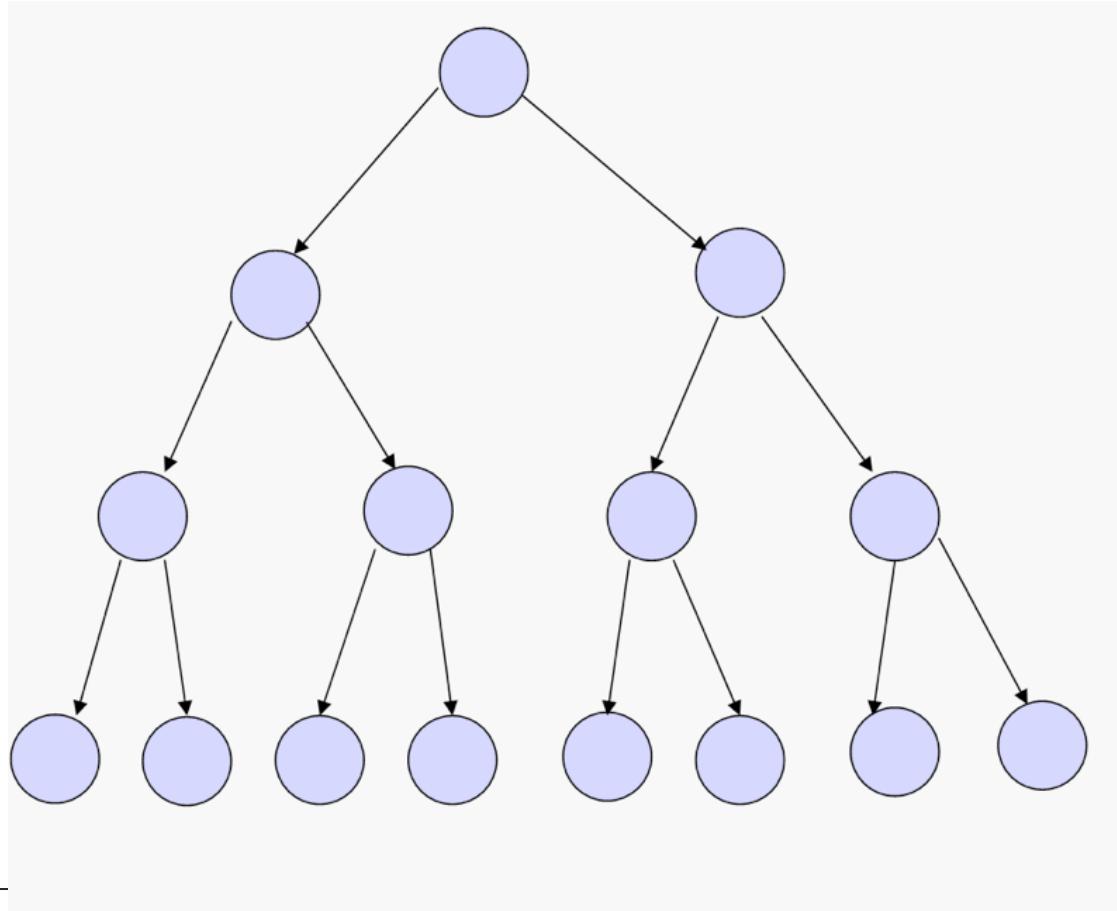




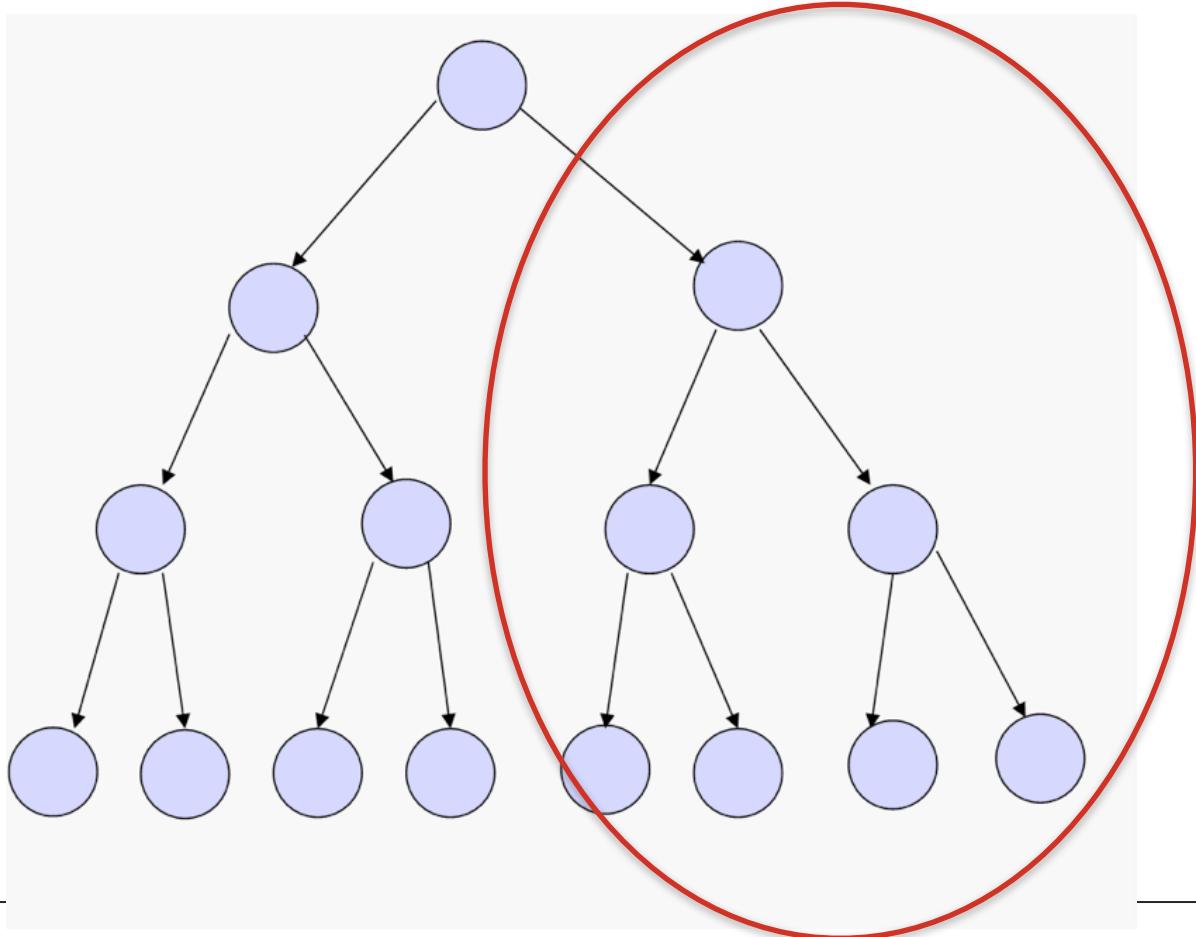
Tell your neighbor: Describe this tree



What do you see here?

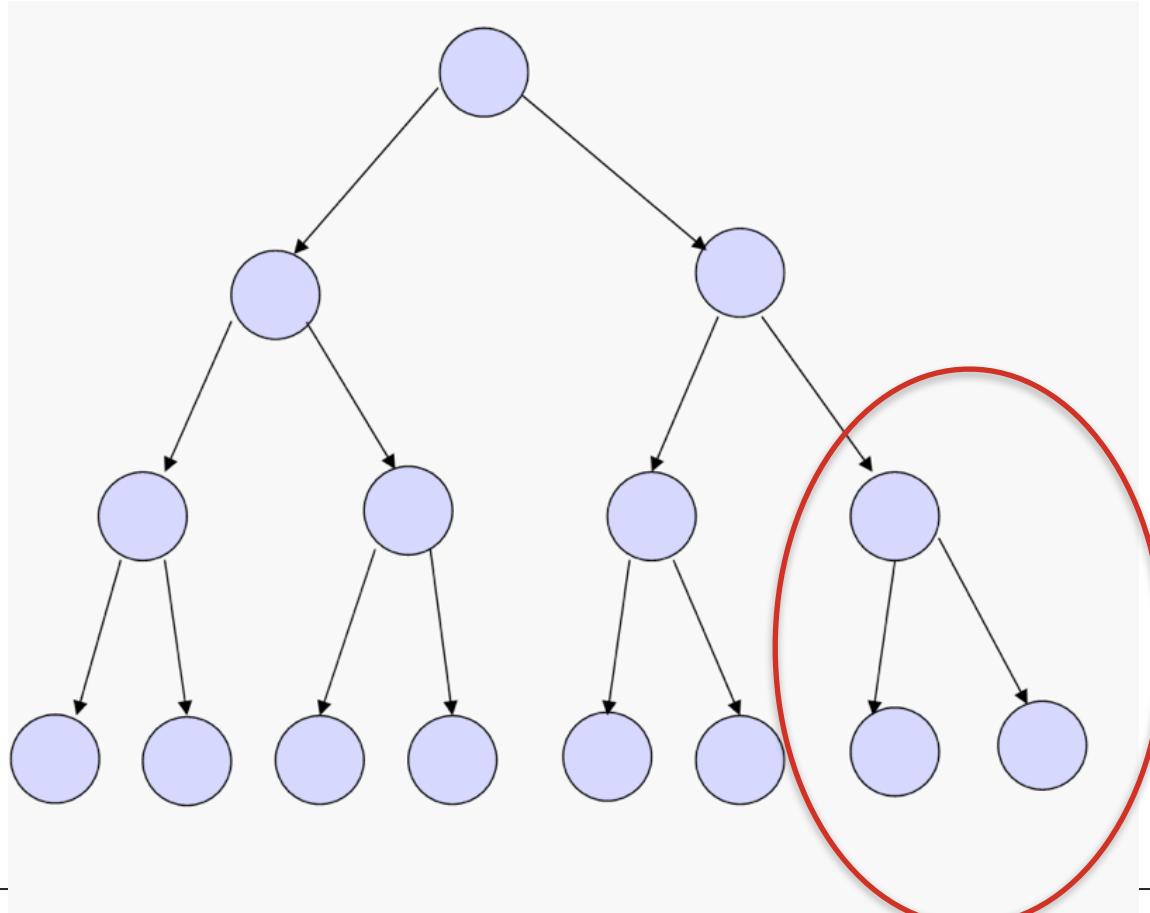


and this... ?



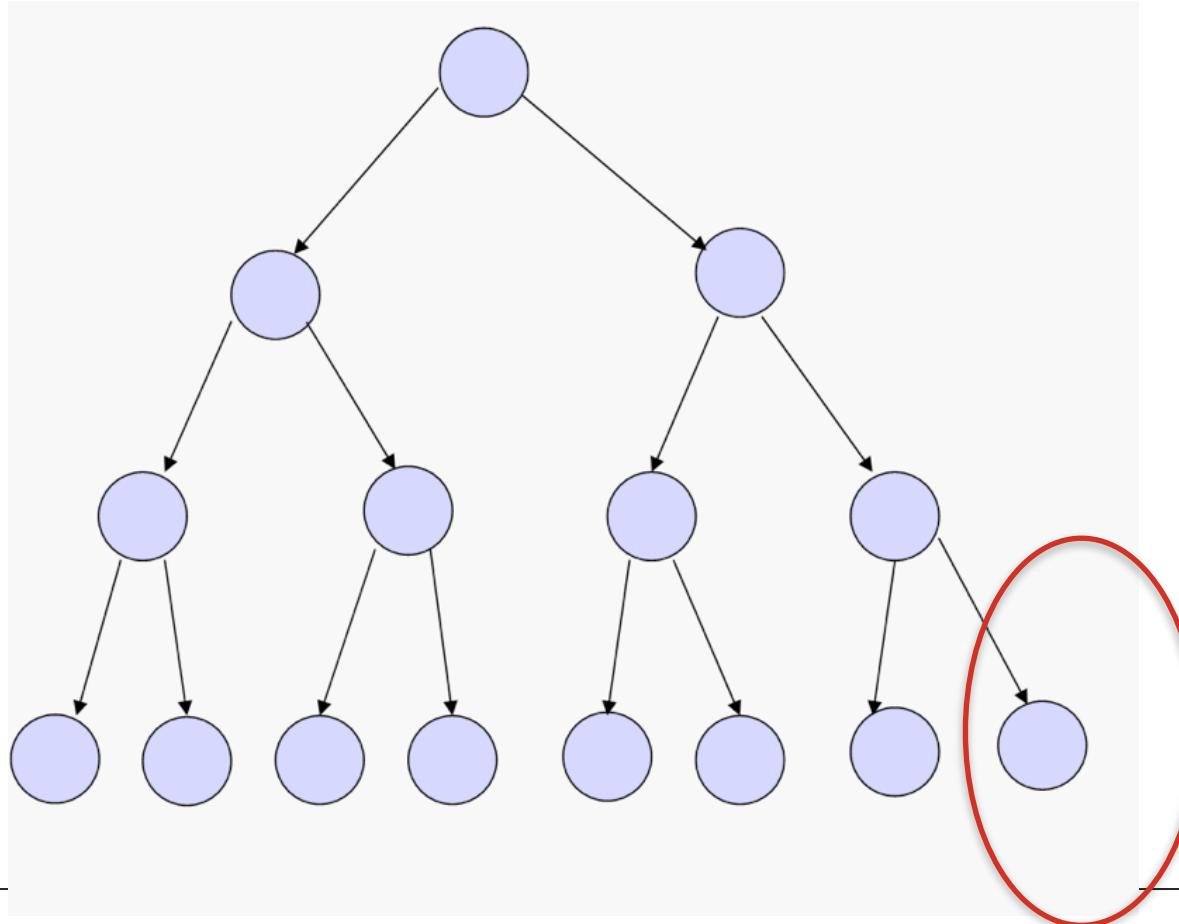
CODE FELLOWS

and then... ?

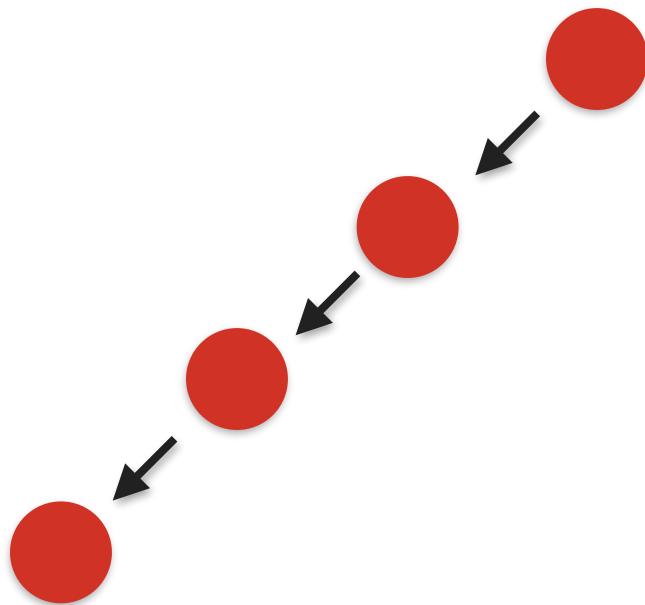


CODE FELLOWS

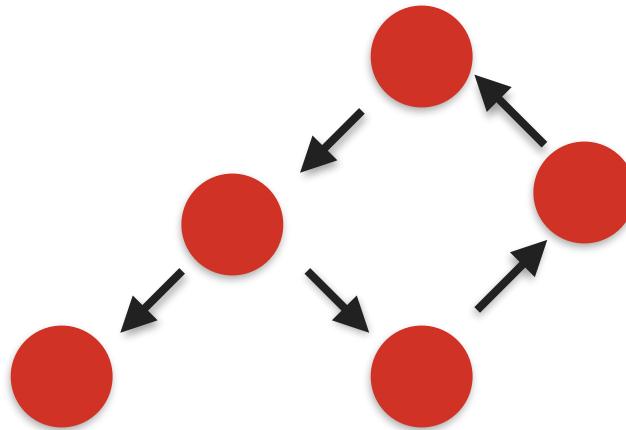
and then... ?



What about this?



This?

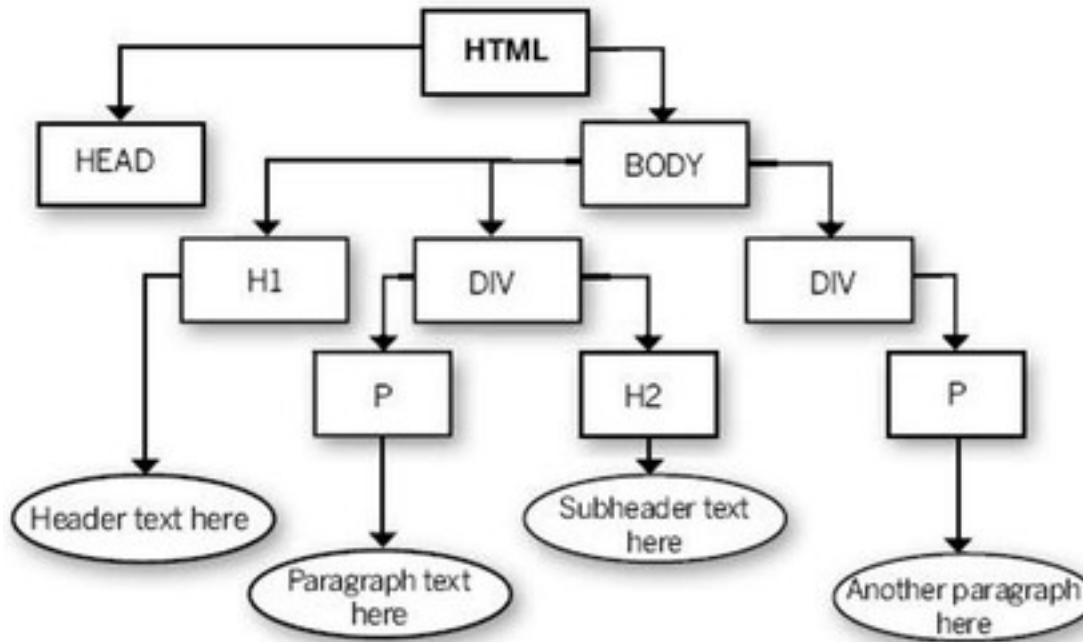


Trees are handy!

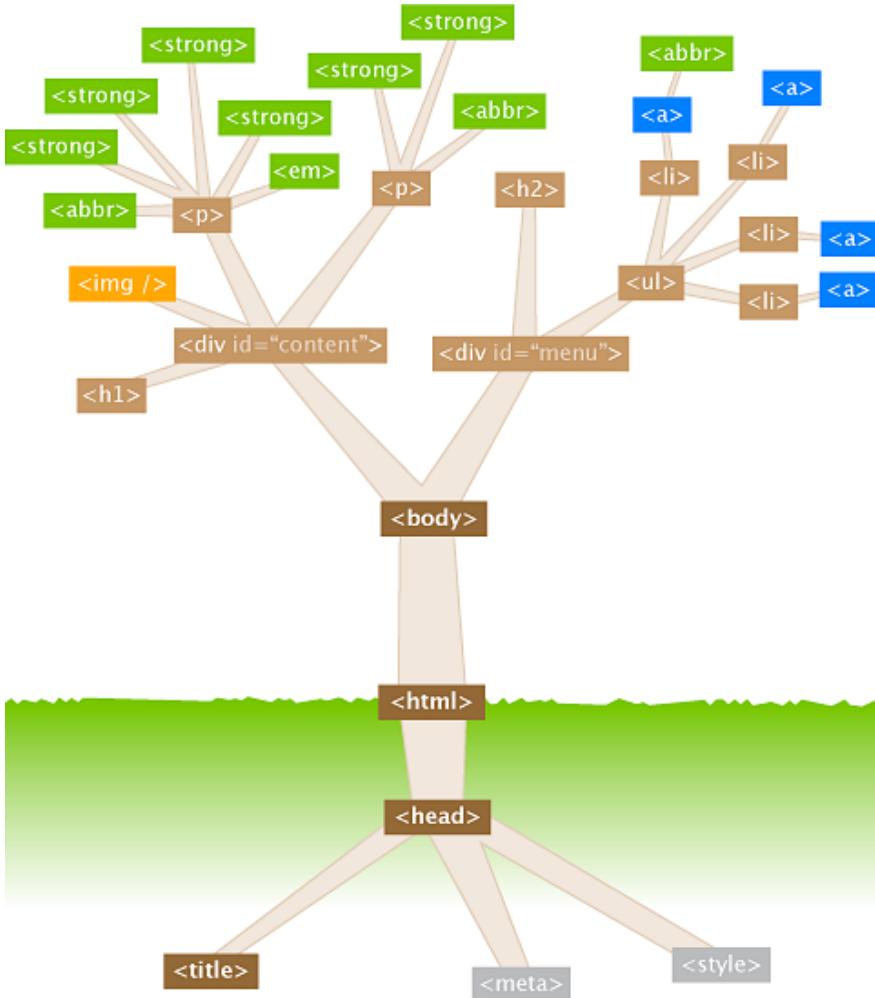
- The DOM
- Computer file system
- Sorting and searching



DOM TREE

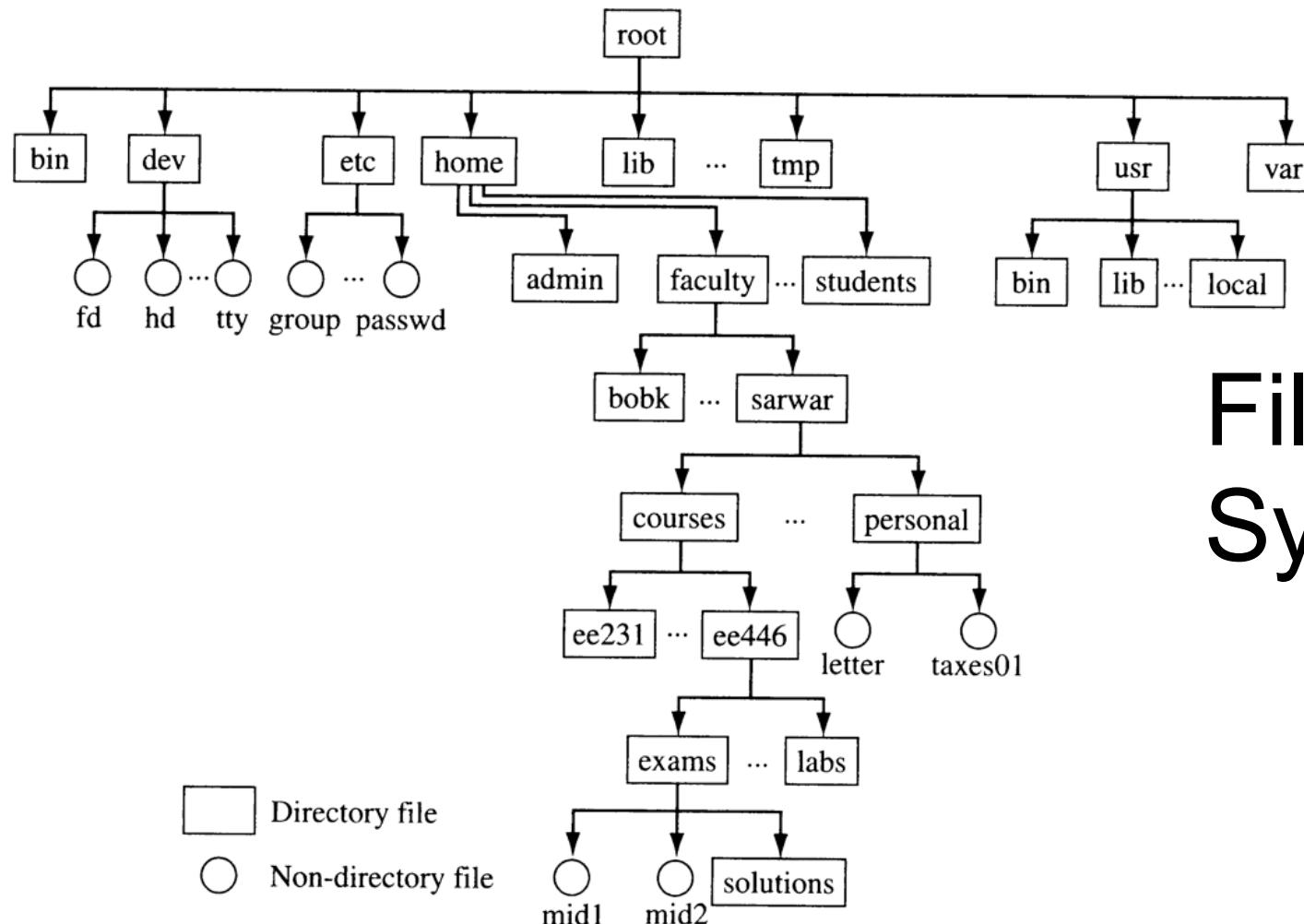


Level order: html, head, body, h1, div, div, header text, p, h2, p, para text, subheader text, another para



What is the root element
of this tree?





File System



Directory file



Non-directory file



CODE FELLOWS

Tree node as an object literal

object that holds a value and references to child nodes

```
{
```

```
  value: null,
```

```
  children: [ ]
```

```
}
```



CODE FELLOWS

Tree node as an object literal

```
{  
  value: { any: valStoredHere },  
  children: [ tree1138, tree42, tree9001 ]  
}
```



CODE FELLOWS

Tree Operations

- Traversal: iterating through every node
- Insert and Delete: add/remove nodes at a position
- Search for a value, min, max
- Calculate properties: size, depth, height



```
1 var D = {  
2   data: 'D',  
3   children: []  
4 };  
5  
6 var B = {  
7   data: 'B',  
8   children: [D]  
9 };  
10  
11 var C = {  
12   data: 'C',  
13   children: []  
14 };  
15  
16 var A = {  
17   data: 'A',  
18   children: [B, C]  
19 };
```

← Consider this set of nodes

**1. Which of the nodes is the root?
How do you know?**

2. Draw a tree diagram for these
nodes similar to the tree below:

