CSCI 136:
Data Structures
and
Advanced Programming

Lecture 21-2

Trees, part 1

Instructors: Dan & Bill

Williams

Outline

Tree terminology

Tree implementation

Tree ADT

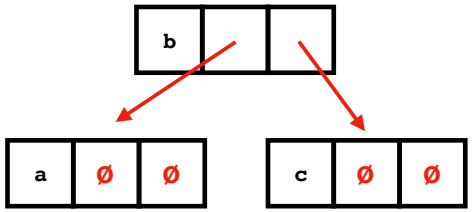
A **tree** is a recursive data structure that stores information hierarchically. A tree is either:

- **empty** (i.e., Ø), or
- a **node** containing a **value** and references to one or more **trees**.

The empty tree:

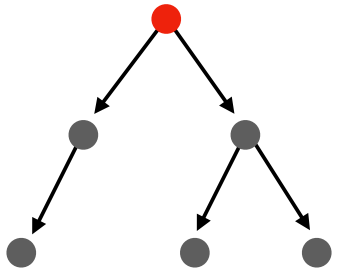
Ø

A non-empty tree:



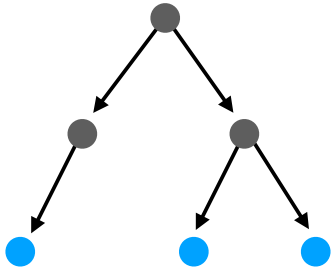
Terminology

The topmost node is called the **root**.



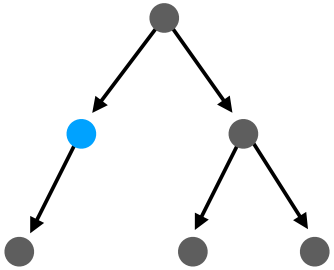
Terminology

The nodes at the bottom of a tree are called **leaves**.



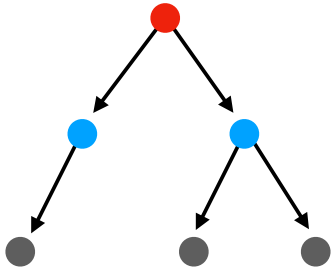
Terminology

Any node that is not a leaf is an **interior node**.



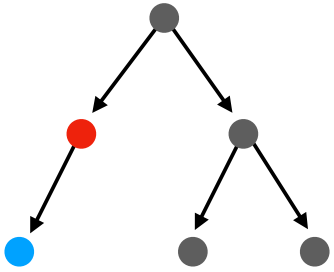
Terminology

A **node** may have **children**.



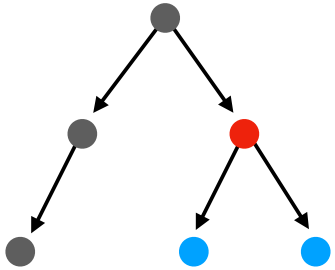
Terminology

A **node** may have **children**.



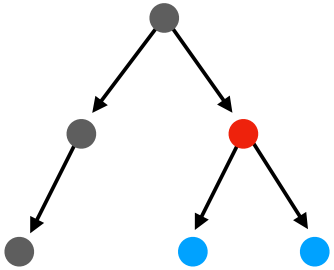
Terminology

A **node** may have **children**.



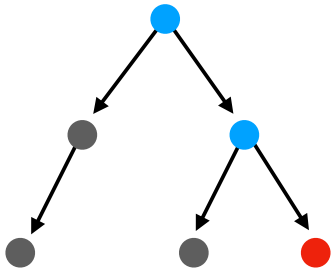
Terminology

A **node** that has **children** is called the **parent** of those children.



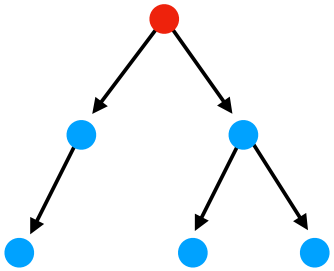
Terminology

For a **given node**, all of the nodes above it are called **ancestors**.



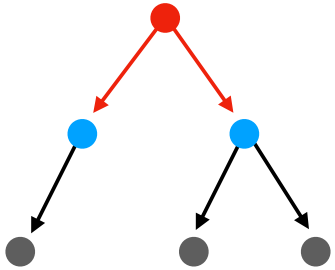
Terminology

For a **given node**, all of the nodes below it are called **descendants**.



Terminology

The **degree** of a tree is the maximum number of **children** had by any node.

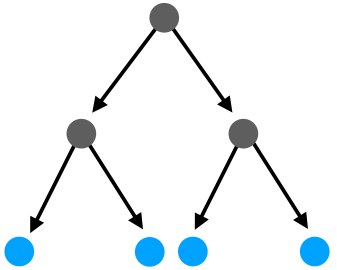


Degree of this tree: 2

Degree 2 trees are common: we call them **binary trees**.

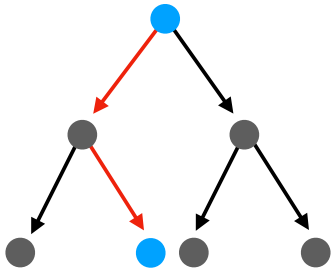
Terminology

A tree that is missing no leaves is **full**.



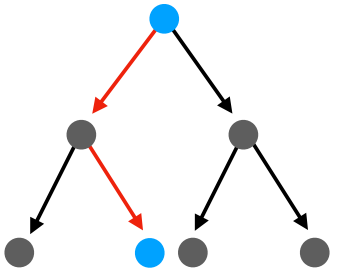
Terminology

A **path** is a sequence of edges between **two nodes**.



Terminology

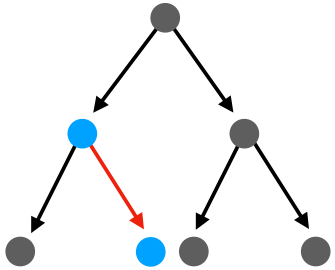
The **length** of a path is the **number of edges** in the path.



Length = **2**

Terminology

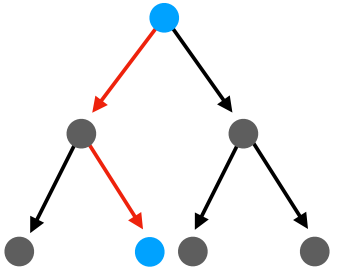
The **height** of **node n** is the length of the longest path between **n** and **any leaf**.



Height of **n** = **1**

Terminology

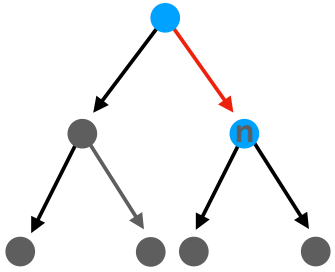
The **height** of **a tree** is the length of the longest path between **the root** and **any leaf**.



Height of **tree** = **2**

Terminology

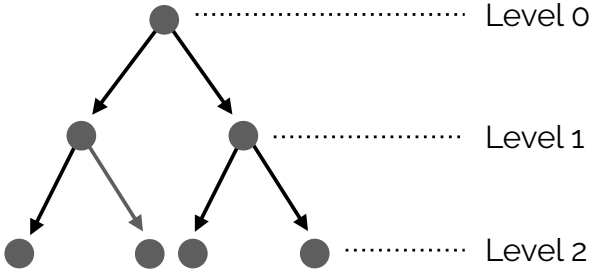
The **depth** of **node n** is the length of the longest path between **the root** and **n**.



Depth of **n** = **1**

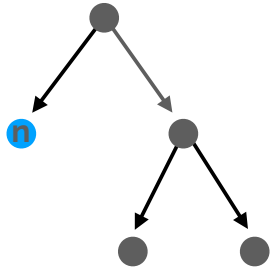
Terminology

The **level** of **any node** is its depth.



Terminology

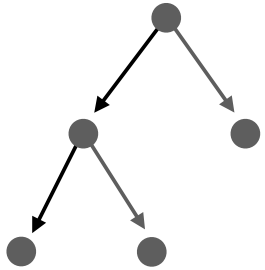
The depth of **n** + the height of **n** ≤ the height of the tree.



(depth of **n**: **1**) + (height of **n**: **0**) ≤ (height of **tree**: **2**)

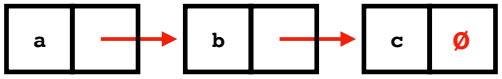
Terminology

A **complete** tree of **height h** is a full tree with zero or more rightmost leaves of **level h** removed.



Terminology

Is a **list** a **tree**?



Yes, a list is a tree whose nodes have **degree** 1.

We call such trees **degenerate**.

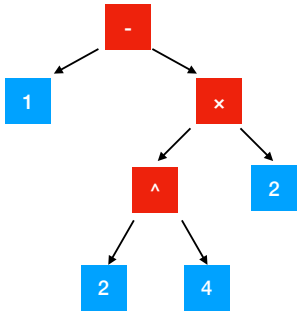
Binary Tree

Binary trees are one of the commonest building blocks of modern data structures.

Let's implement one together.

Uses

$1 - 2^4 \times 2$



Recap & Next Class

This lecture:

- Terminology
- Implementation

Next lecture:

- Traversals
- Tree height
- Binary Search Trees