Day 23 – Trees and Binary Search Trees

Chapter 07, 7.1 – Trees, Review Problem(s): ***4, 6***

Chapter 07, 7.2 – Binary Search Trees, Review Problem(s): ***9, 12, 13***

Chapter 07, 7.3 – The Binary Search Interface, Review Problem(s): ***18***

We start with the definition of a Tree, and then the definition of a Binary Search Tree (BST). Students who have completed MA 116 – Discrete Structures, may notice small, subtle, differences between the Tree and BST definitions in our textbook and from MA 116. We will use the definitions from our text.

Let's review several definitions which you should be familiar with:

- Tree
- Root
- Parent node
- Subtree
- Children
- Ancestor
- Descendant
- Leaf
- Interior node
- Siblings
- Level
- Height

A tree can be traversed in several ways:

- Breadth-first
- Depth-first

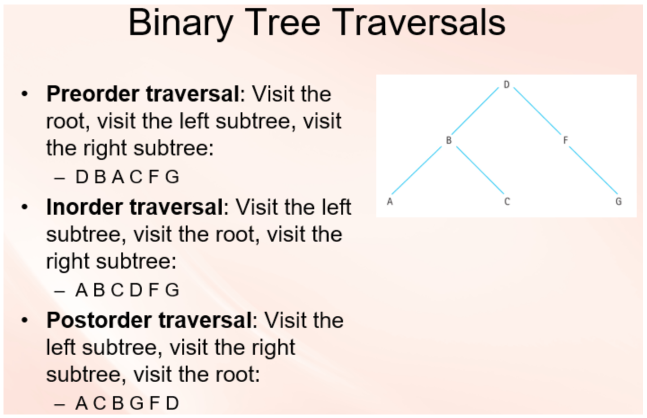
Note the algorithms for each of these use Queues and Stacks respectively.

Chapter 07 PPT, pg 01-15.

BSTs are a subset of Trees, nodes can have 0, 1, or 2 children, and the nodes have a key value that is `Comparable`. All key values of the nodes in the left subtree are less than or equal to the value of the parent node, and the key values of the nodes in the right subtree are greater than the key value of the parent.

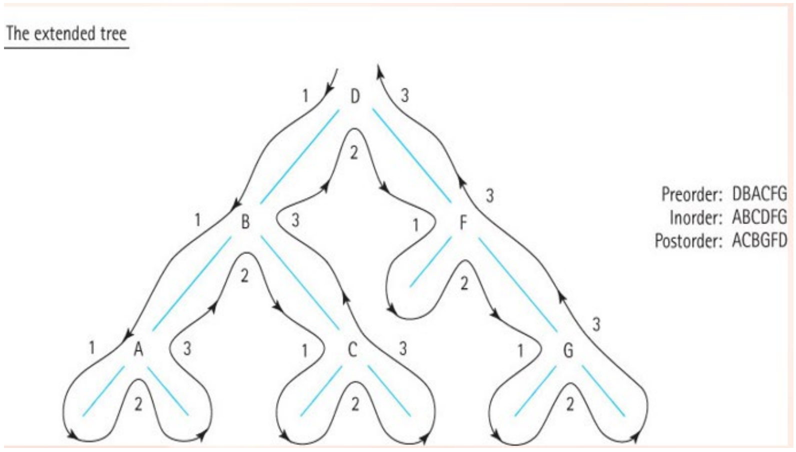
In addition to the traversals above, BSTs have additional traversals:

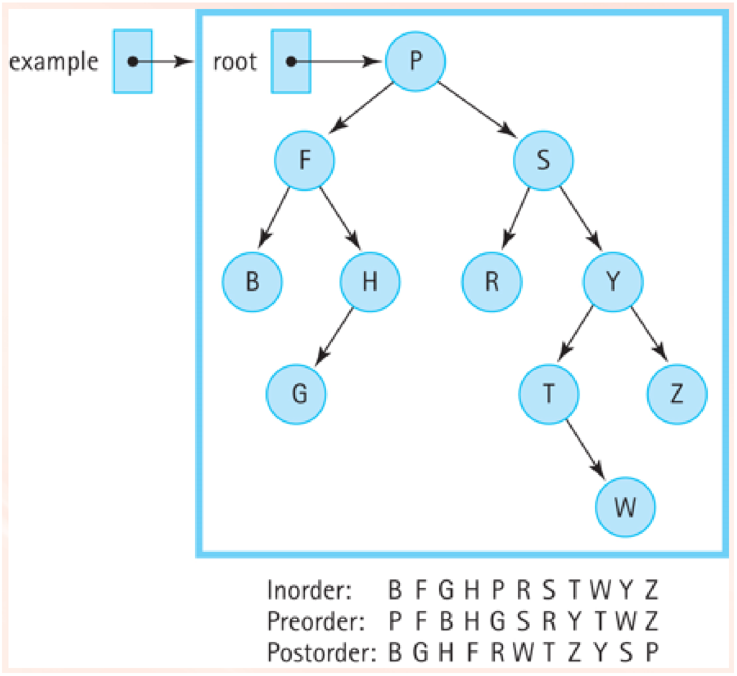
- Preorder
- Inorder
- Postorder



Binary Tree Traversals

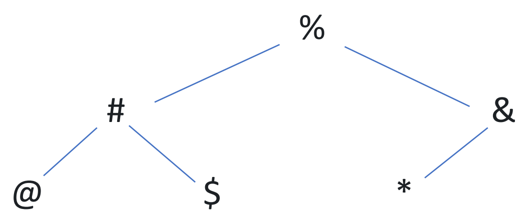
- **Preorder traversal**: Visit the root, visit the left subtree, visit the right subtree:
  - D B A C F G
- **Inorder traversal**: Visit the left subtree, visit the root, visit the right subtree:
  - A B C D F G
- **Postorder traversal**: Visit the left subtree, visit the right subtree, visit the root:
  - A C B G F D




example
root
P
F
S
B
H
R
Y
G
T
Z
W
Inorder: B F G H P R S T W Y Z
Preorder: P F B H G S R Y T W Z
Postorder: B G H F R W T Z Y S P

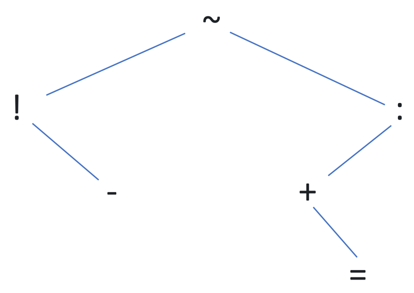
Now, consider the following two traversals of the ***same*** tree:

Inorder: @, #, $, %, *, &
Preorder: %, #, @, $, &, *



Let’s try another.

Inorder:	!, -, ~, +, =, :
Postorder:	-, !, =, +, :, ~



Chapter 07 PPT, pg 16-21.

The Binary Search Tree Interface is similar to the sorted lists from the previous chapter. A class that implements the `BSTInterface` must provide a separate iterator method, because `BSTInterface` extends `Iterable`. This method should return an `Iterator` that provides iteration in the “natural” order of the tree elements. For most applications this would be an inorder traversal, and we make that assumption in our implementation.

Chapter 07 PPT, pg 22-26.