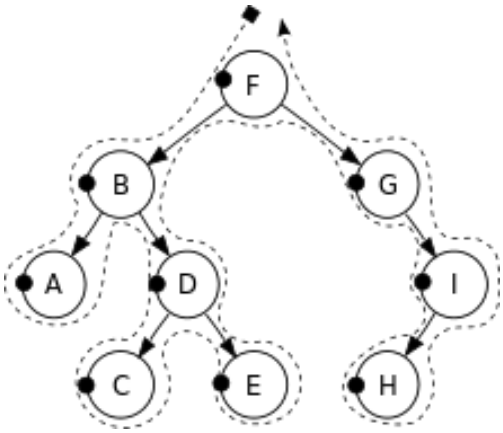
Tutorial 12

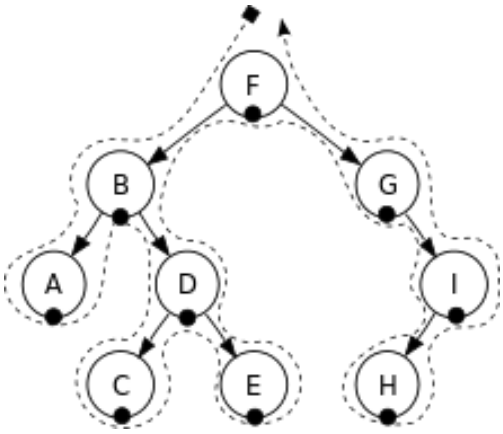
Today we'll be working with trees:

- Orderings: PREORDER, INORDER, POSTORDER.
- Tree of `int` to array.
- `char*` tree and the burden of free
- Review of Data Storage (POOL hint)



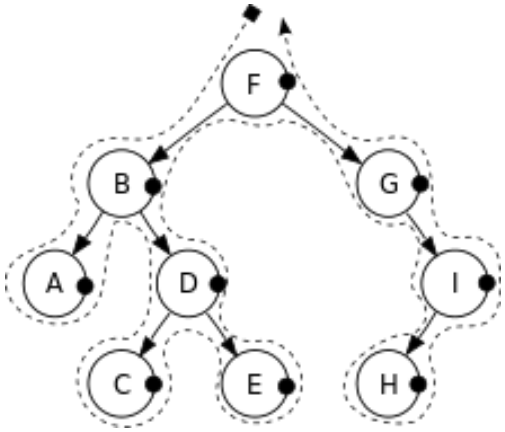
PREORDER

Exectue code on self before going to children



INORDER

Exectue code on left, then self, then right



POSTORDER

Exectue code on children before self

BST to Array (treesort)

The ***treesort*** algorithm sorts items by building a BST and then traversing it INORDER.

For this problem, the items in the BST will be `int`s, and we will generate a **new** array containing all of the items in order.

We will attempt this in two ways:

- call `bst_select(k, t)` n times [slow - $O(nh)$ or $O(n \log n)$ if balanced]
- traverse the tree in order [fast - $O(n)$]

ADT Pro/Con

Functions	Linked List	Dynamic Array	BST
Insert	O(i)(auto-shifts)	O(n)	O(h)
Remove	O(i)	O(n)	O(h)
Lookup	O(i)	O(i) (with address)	O(h)
Destroy	O(n)	O(1)	O(n)