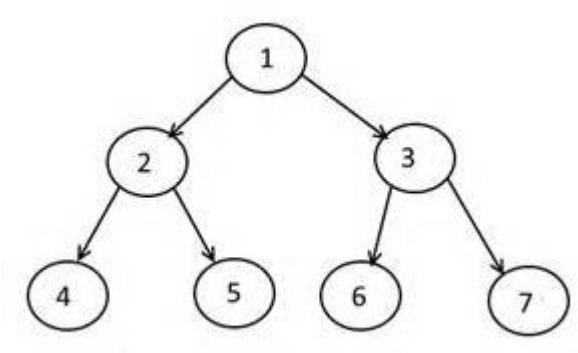
trees2

trees

- First, lets implement a very simple binary tree.
  - Not binary search tree.
- We can use **nodes** for it.
- Similar to linked list.
- Use left and right pointers.



remember traversals

- in order
  - left - root - right
- post order
  - left - right - root
- pre order
  - root - left - right
- breadth first search
  - before we go to level x, we need to finish level x-1
  - commonly used in games

Implementation

- Can be done recursively or iteratively
- Iterative
  - By using a stack (or stacks)
  - And by using a queue for BFS

pre order traversal

- Imagine that we have a stack of books.
- In preorder, we first look at the cover of the book (root), then we read the left page (left subtree) and right page (right subtree).
- First, we put the cover on our book pile.
  - Push the root onto the stack.
- Now, we start going through our pile while the stack is not empty.
  - We read the top book. Take the top book from the pile (pop) and look at it (current node)
  - If there is a right page, add it to pile for later. (push to stack) If there is a left page, add it to pile on top of everything else.(push to stack)

in order traversal

- Imagine we are in a forest.
- First we explore the left part of the forest, then the area we are standing in (root), and finally the right part.
- We start at our current location (root) and always move left as far as we can, adding each spot we visit to our notebook
  - pushing nodes into a stack
- Once we cannot go any further, we start checking our notes. (while the stack is not empty)
  - We go back to the last spot we noted (pop from the stack) and process the node.
  - If there is a path to right, we follow it and keep noting down every left turn we can take.

post order traversal (two stacks)

- First we explore the entire left and right parts of the forest and only then we check where we are standing.
- We will have two piles of leaves. First pile is to remember where to go, second to remember what we saw.
- We start from where we are standing.
  - Push root onto first stack.
- While we have places to visit in the first pile:
  - We take the top place from the first pile (pop from first stack), put it in our second pile (push to second stack) and plan to visit its left and right areas later (push left and right children onto first stack, right child first)
- Once we've planned all visits, we go through the second pile (while the second is not empty) and remember what we saw at each spot (pop from the second stack)

breadth first

- Exploring the forest one step at a time and seeing everything immediately before moving further.
- We start from the root. We put it into our exploration list.
  - Add to queue.
- While we have places to explore, we look around.
  - pop from the queue and print
- Note nearby places
  - Add left and right children to queue