Lam Dang.
Binomial Heap.

Writeup 1B. For Binomial Heap

Biggest Advantage: Merging of 2 heaps is O(log n) time. Since a Tree with order k can hold 2^k node, and we only merge tree with same order, we will only merge log(n) tree.

Biggest Dis advantage: Get min requires O(log n), Since a heap is a composition of trees, we need to check root of every tree to find min node

Writeup 1C:

a/ Insertion: Worst - log(N), Amortized (1).

b/ Merge: Log(N). A tree of order k can hold 2^k values. We only need to merge tree with same order (that takes O(1) exec time). Therefore we only have to merge log(n) time, resulting in O(log n) time

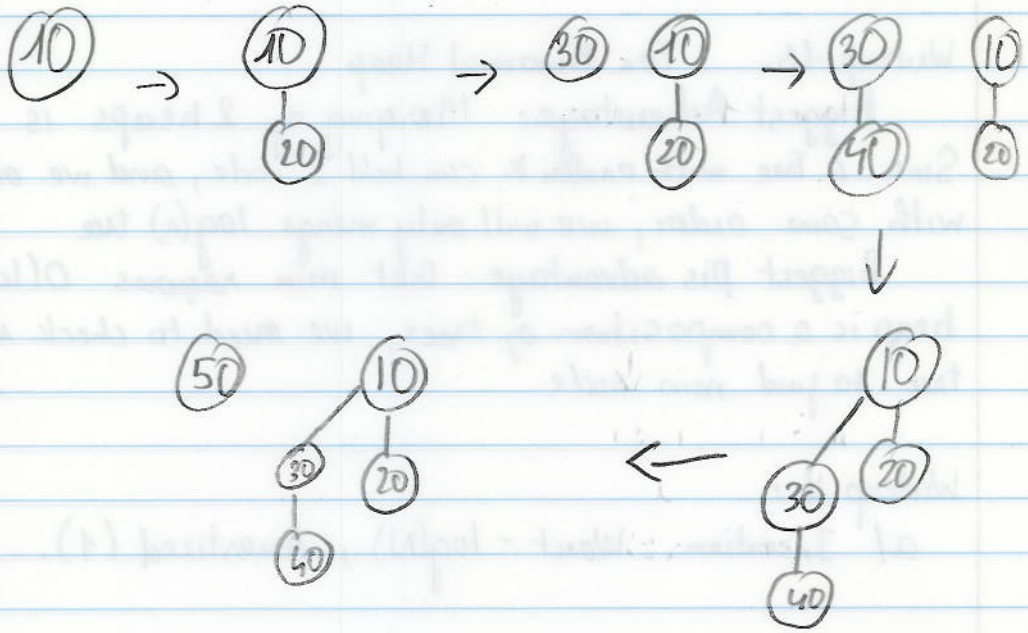
c/ Find_min: log(n).

Since Binomial heap is a composition of tree, you need to check the root of each tree to find the min. Since for n nodes, the number of tree, at worst, is log(n) trees, find_min will take log(n) time.

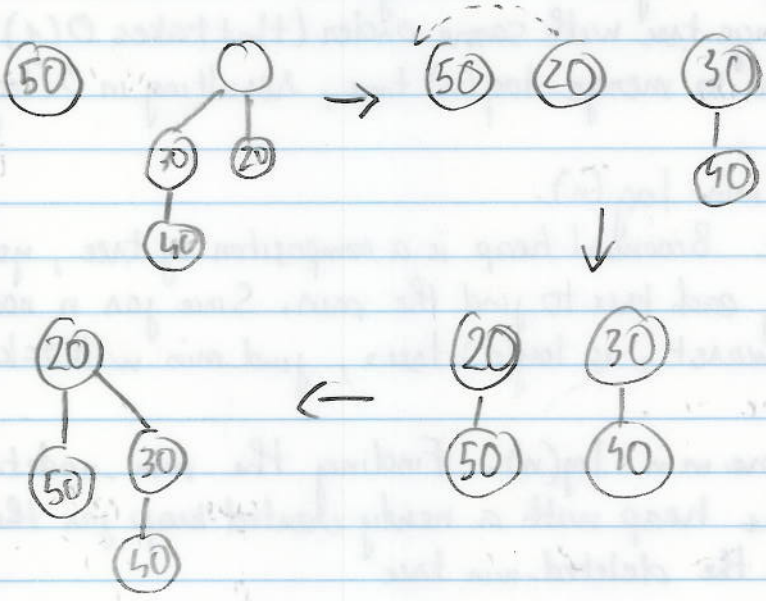
d) Remove min: log(n). Finding the min_node takes log(n) time. Merging the heap with a newly created heap for the ~~other no~~ nodes that are in the deleted_min tree

*Writeup 1A.

Insert: 10, 20, 40, 30, 50

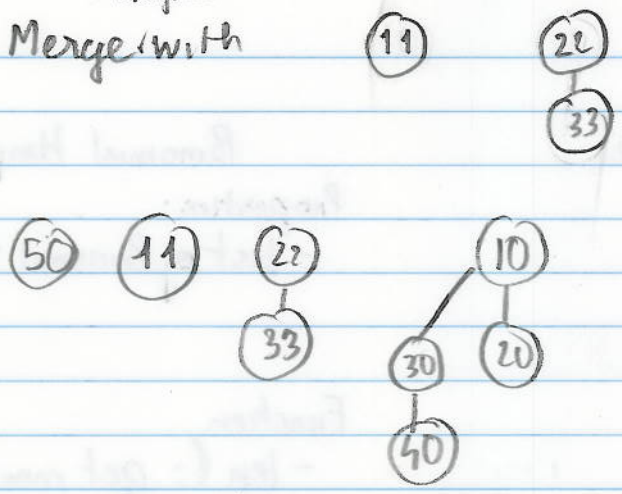


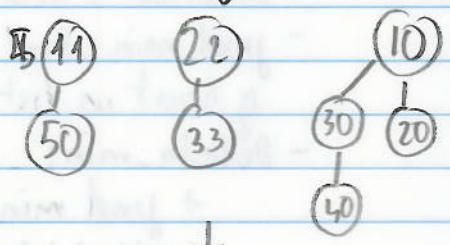
Delete min

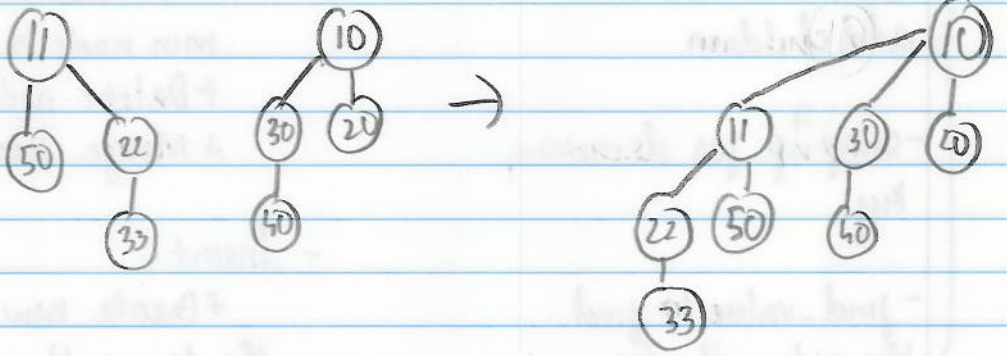


original

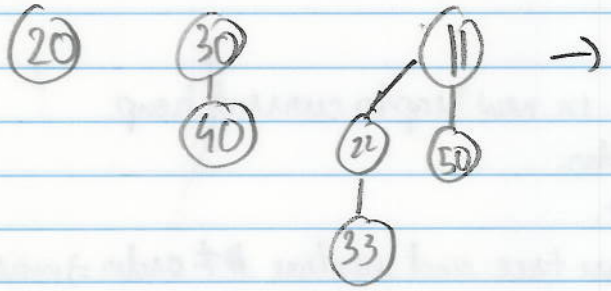
Merge with



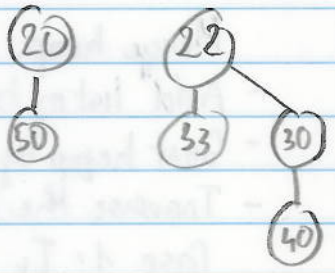




Remove min



Remove min



Design.

Binomial Tree

properties:
- key
- value
- order
- list of children
- parent

function:
- get-set for key, value, order
- add_children
- Increase order when add children
- swap up for decreasing key
- find_value to find the node with value

Binomial Heap

Properties:
- List of Binomial trees

Function
- len(): get number of trees
- add tree: add to tree list
- find min: Find minimum of root in list of tree
- Delete_min:
  + find min. If None, done
  + add children of min node to a heap
  + Delete node
  + Merge heaps.
- Insert:
  + Create new heap, add the tree with only one node that include (key, value)
  + Merge heap.

Merge heap.
- Add list of tree in new heap to current heap.
- Sort heap by order.
- Traverse the list.

Case 1: If cur tree and next tree ≠ order => move on

If same order Case 2: If next tree and next next same order => move on

Else case 3: If root key of cur < next => next is cur child

Case 4: ——————> next => cur is next child