EECS2040 Data Structure Hw #4 (Chapter 5 Tree)
due date 5/16/2022, 23:59
by 108011235 陳昭維

Part 1 (2% of final Grade)

1. (4%) What is the maximum number of nodes in a k-ary tree of height h? Prove your answer.

   <answer>

   For height of 1, there is maximum of 1 node.

   For height of 2, there are maximum of $1 + k$ nodes.

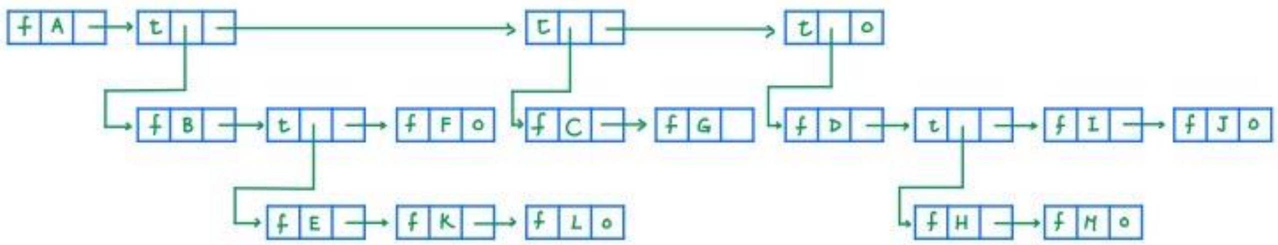
   For height of 3, there are maximum of $1 + k + k^2$ nodes.

   For height of h, there are maximum of $1 + k + k^2 + \cdots + k^{h-1} = \sum_{i=0}^{h-1} k^i = \frac{1(1-k^h)}{1-k} = \frac{k^h-1}{k-1}$

2. (16%) For a simple tree shown below,

   (a) Draw a list representation of this tree using a node structure with three fields: tag, data/down, and next.

   <answer>



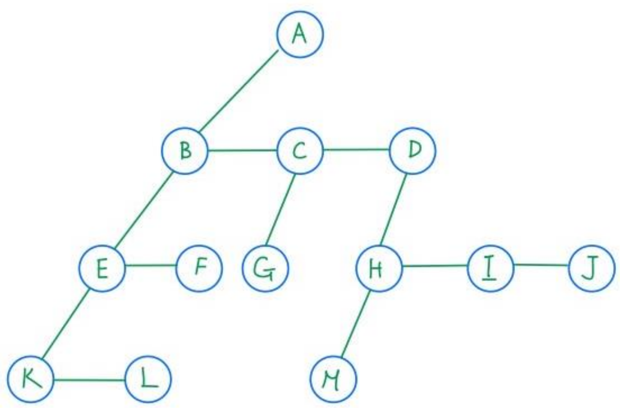
   (b) Write down a generalized list expression form for this tree.

   <answer>

   (A(B(E(K,L),F),C(G),D(H(M),I,J)))

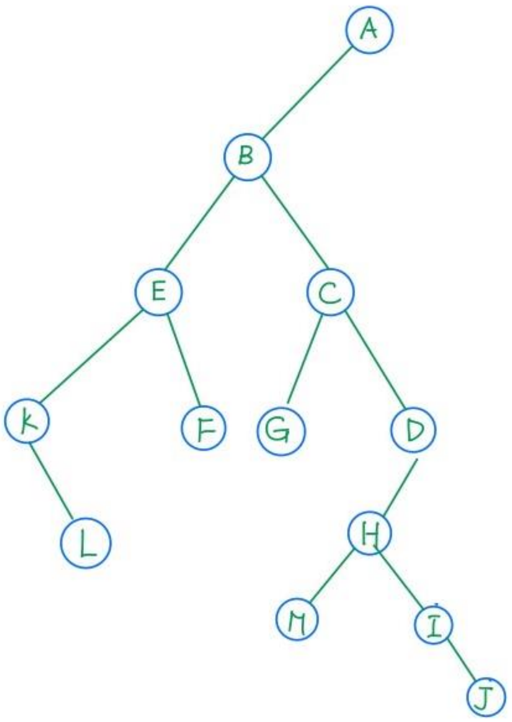
   (c) Convert the tree into a left-child and right-sibling tree representation

   <answer>



(d) Draw a corresponding binary tree for this tree based on (c).

<answer>



(e) What is the depth of node L? What is the height of node B? What is the height of the tree?

<answer>

Consider the original tree in this problem

The depth of node L is 4, the height of node B is 3, the height of the tree is 4

(f) Write out the preorder traversal of this tree.

<answer>

Preorder traversal: ABEKLFCGDHMIJ

(g) Write out the postorder traversal of this tree.

<answer>

Postorder traversal:

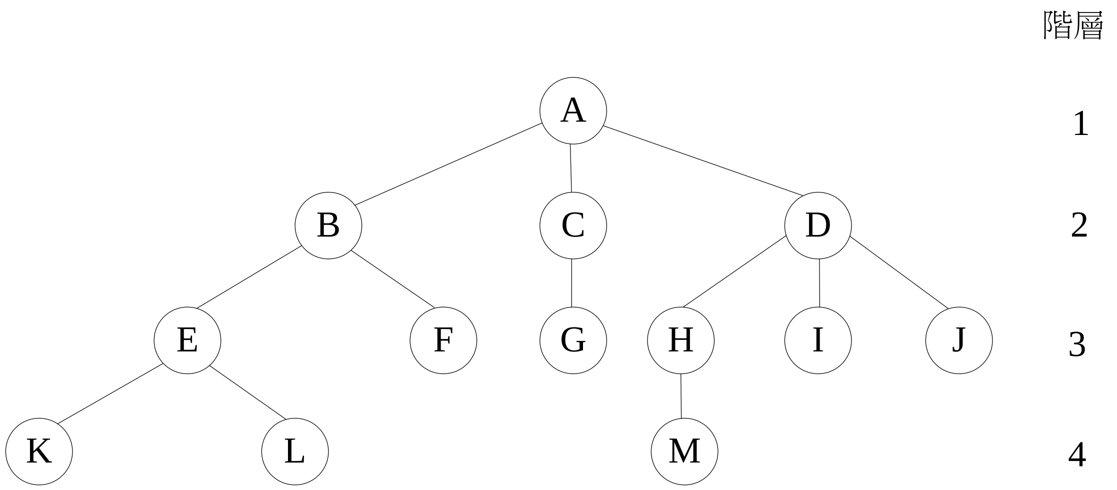
KLEFBGCMHIJDA

(h) Write out the level order traversal of this tree.

<answer>

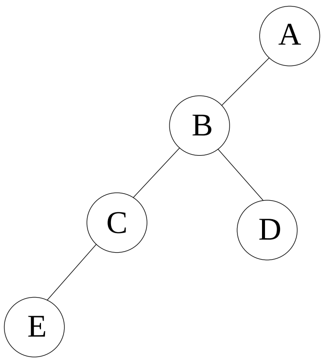
Level order traversal:

ABCDEFGHIJKLM

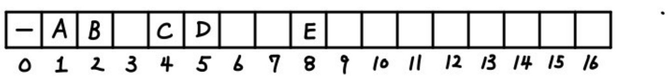


3. (10%) Draw the internal memory representation of the binary tree below using (a) sequential and (b) linked representations.

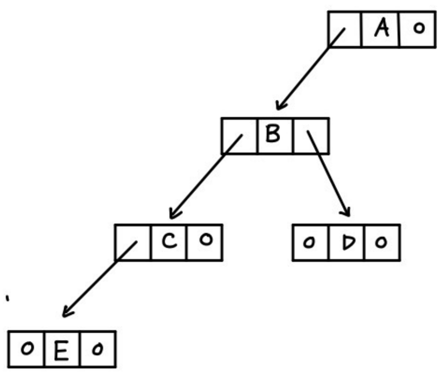
<answer>



(a) sequential



(b) linked



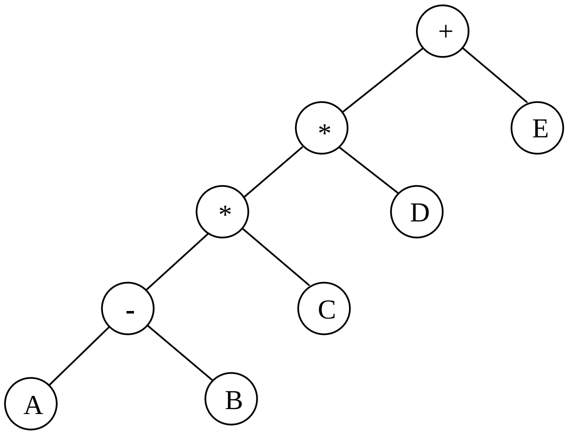
4. (4%) Extend the array representation of a complete binary tree to the case of complete trees whose degree is d, d > 1. Develop formulas for the parent and children of the node stored in position i of the array.

<answer>

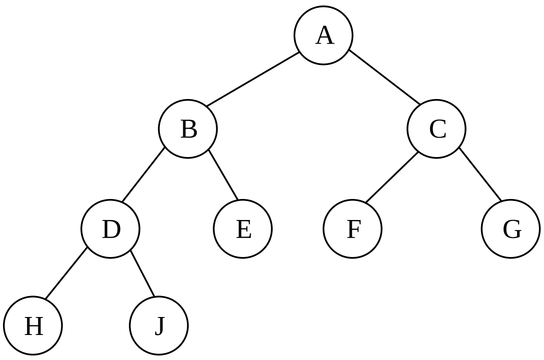
For node i, the parent node is node $\left\lfloor \frac{i-2+d}{d} \right\rfloor$ for $i > 1$, if the node is 1 then it is root node

The jth child node of node i is $i \times d - d + j + 1$, if and only if the sum is less or equal to n, else the child doesn't exists.

5. (16%) Write out the inorder, preorder, postorder, and levelorder traversals for the following binary trees.



(a)



(b)

<answer>

For tree (a):

Inorder: A – B * C * D + E

Preorder: +**-ABCDE

Postorder: AB-C*D*E+

Levelorder: +*E*D-CAB

For tree(b):

Inorder: HDJBEAFCG

Preorder: ABDHJECFG

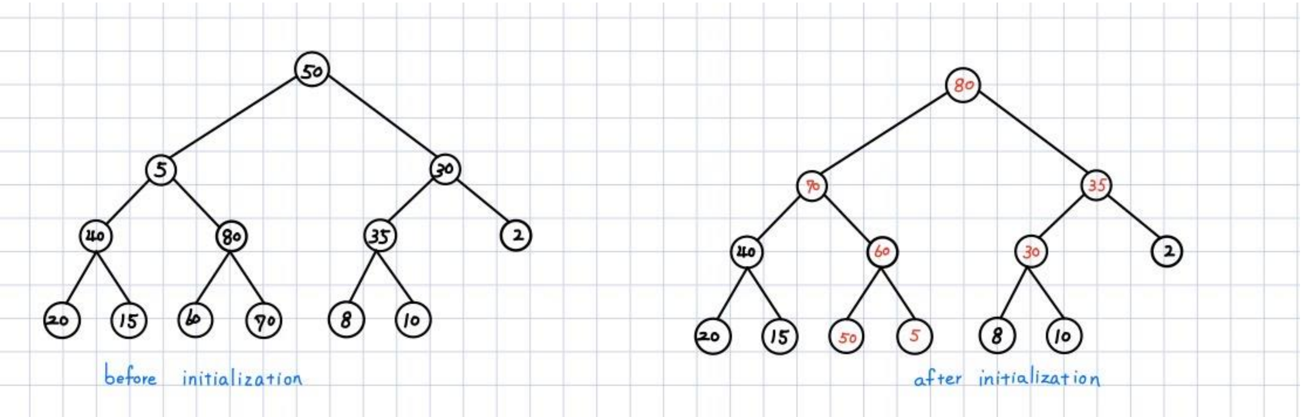
Postorder: HJDEBFGCA

LevelOrder: ABCDEFGHJ

6. (16%) Given a sequence of 13 integer number: 50, 5, 30, 40, 80, 35, 2, 20, 15, 60, 70, 8, 10.

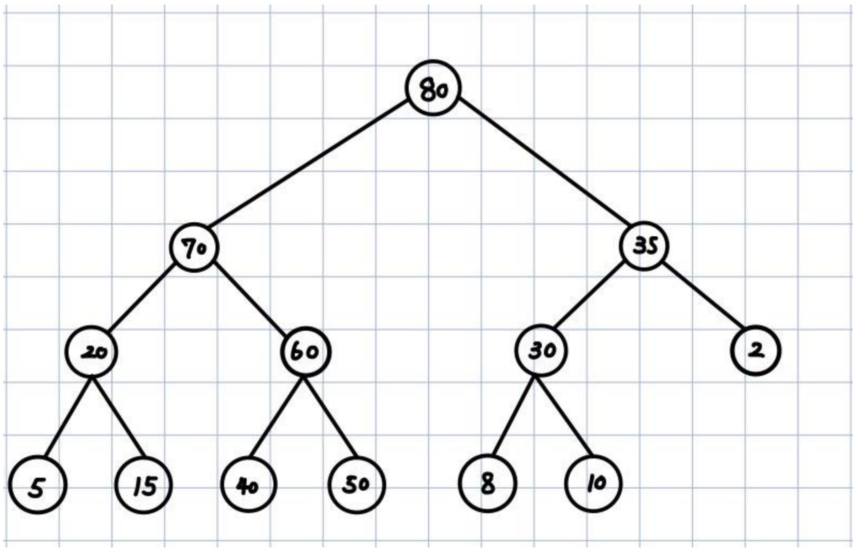
(a) Assume a **max heap** tree is **initialize** with these 13 numbers placed into nodes of the tree according to node numbering of complete binary tree by using the **bottom up heap construction initialization** process. Please draw the final Max heap tree after initialization process.

<answer>



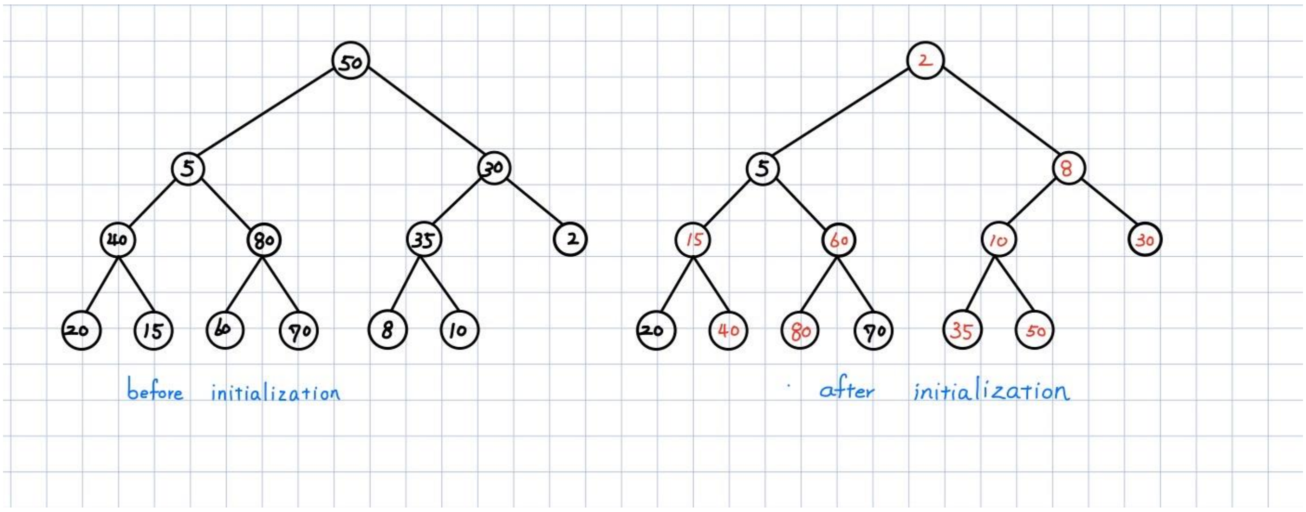
(b) Construct a max heap by **inserting** the given 13 numbers one by one according to the sequence order into an initially empty max heap tree, instead of bottom up heap construction.

<answer>



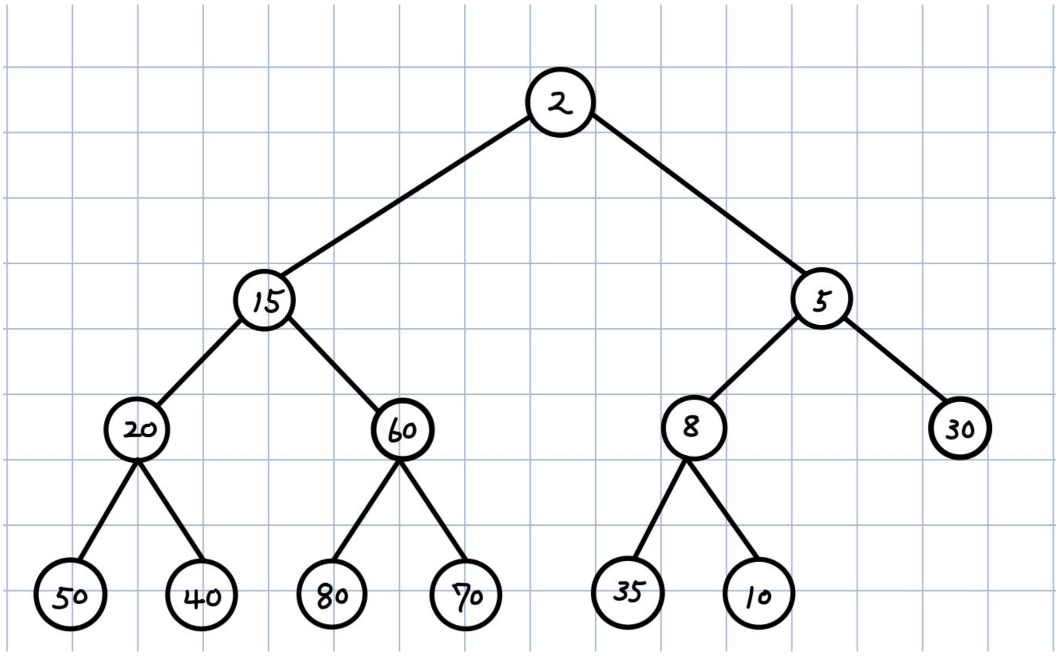
(c) Repeat (a) for min Heap.

<answer>



(d) Repeat (b) for min Heap.

<answer>



7. (4%) For initializing a max heap from n key values stored in an array, instead of inserting key value one by one, assume that

   The height of heap = h,

   Number of subtrees with root at level j is <= 2^{j-1},

   Time for each subtree is O(h-j). (# of bubbling down),

   Time for level j subtrees is <= 2^{j-1}(h-j) = t(j),

   Then total time is t(1) + t(2) + … + t(h-1) = O(n).

   Please prove the above argument.

   <answer>

Assume the heap is a full binary tree such that it is the maximum case.

And assume the root node is defined at level 1, so that the height count will be +1 compare to some definition.

From the bottom there are 2^{h-j} nodes where (h – j) is the level, each might sift down (h -j -1) level, thus we can represent the summation of t(1) + t(2) + … + t(h-1) as a big T(n)

Where

$$\mathrm{T(n)} = \sum_{j=1}^{h} (j-1)2^{h-j} = \sum_{j=1}^{h} (j-1)\frac{2^h}{2^j} < \sum_{j=1}^{h} j\frac{2^h}{2^j} = 2^h \sum_{j=1}^{h} \frac{j}{2^j} \leq 2^h \sum_{j=1}^{\infty} \frac{j}{2^j} \leq 2^h$$

Thus

$\mathrm{T(n)} \leq n+1$ is bounded by n, $\rightarrow \mathrm{T(n)} = O(n)$

8. (20%) Binary Search Tree

(a) How many different binary search trees can store the keys {1,2,3}?

<answer>

For the root being 1, there are 2 different trees can be constructed

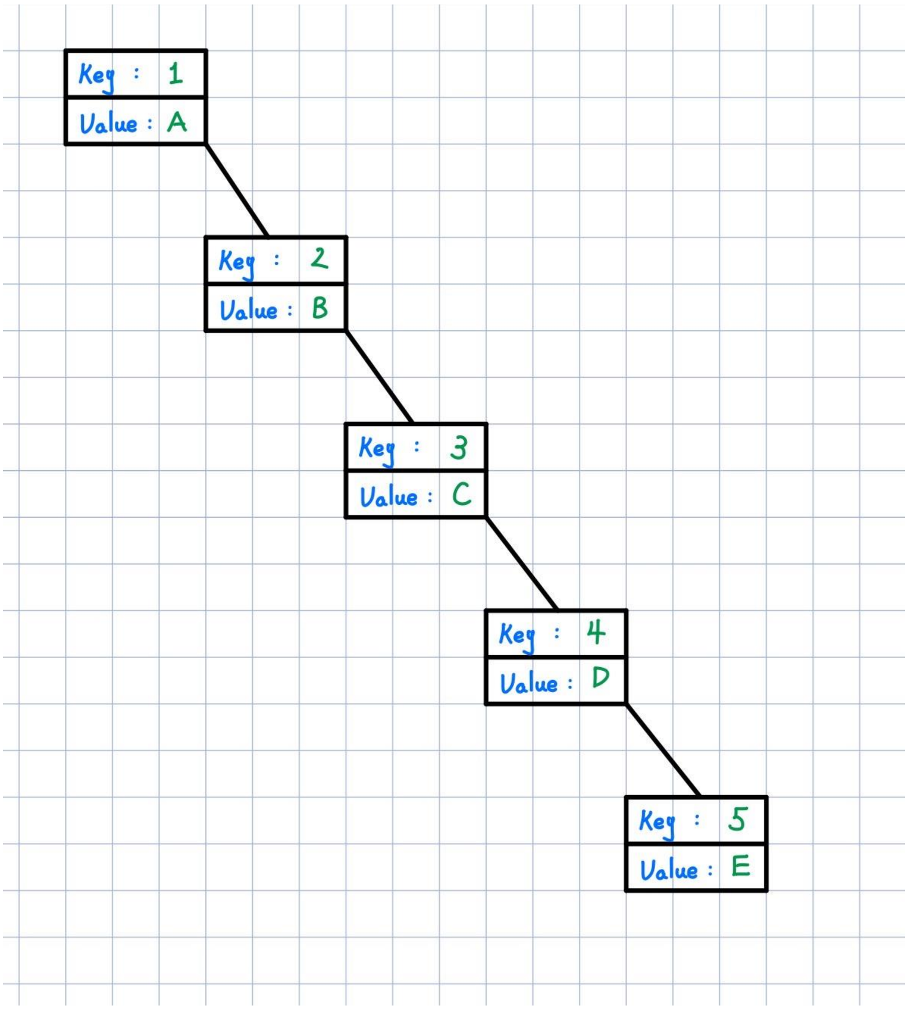
For the root being 2. There are 1 tree can be constructed

For the root being 3. There are 2 trees can be constructed

Thus, there are 5 BST that can store the keys {1,2,3}

(b) If we insert the entries (1,A), (2,B), (3,C), (4,D), and (5,E), where the number denotes the key value of the node, in this order, into an initially empty binary search tree, what will it look like? Please draw this BST.

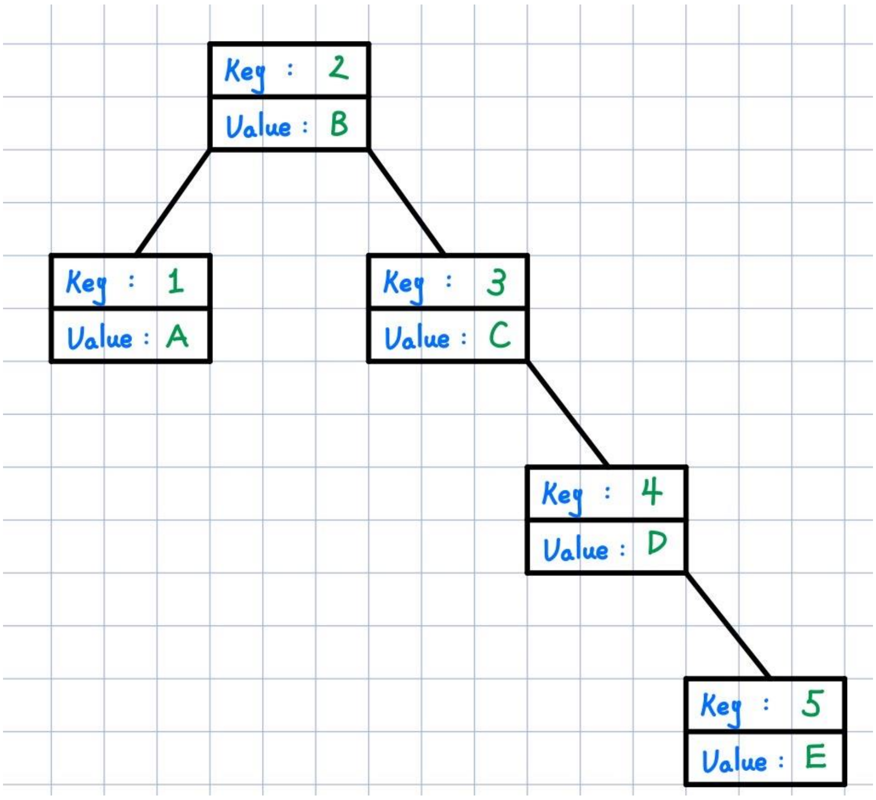
<answer>



(c) John claims that the order in which a fixed set of entries is inserted into a binary search tree does not matter—the same tree results every time. Give a small example that proves he is wrong.

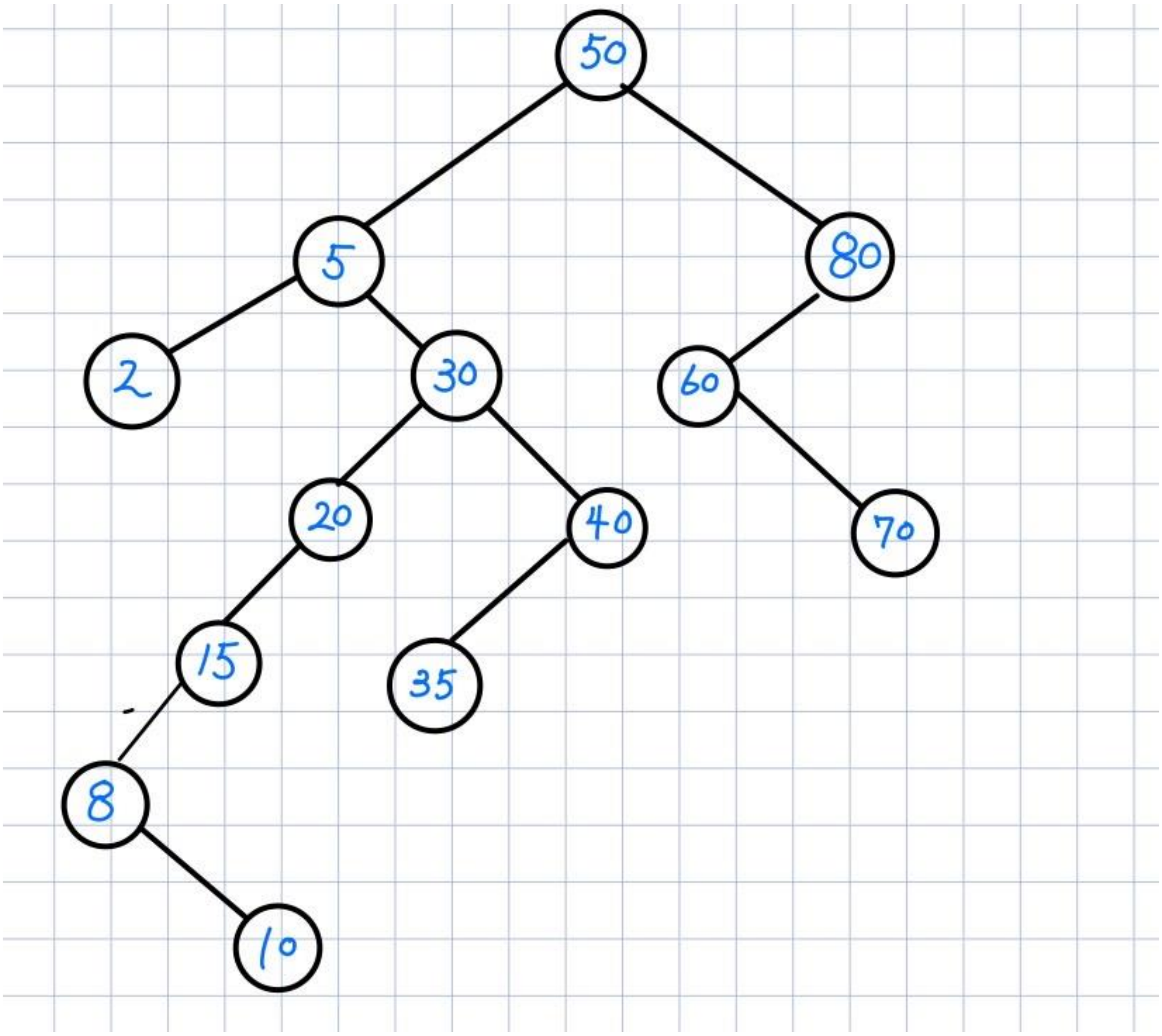
<answer>

Just consider the previous case, while switching the first 2 entries (2,B), (1,A), (3,C), (4,D), and (5,E), then the tree will look like the below



(d) Given a sequence of 13 integer number: 50, 5, 30, 40, 80, 35, 2, 20, 15, 60, 70, 8, 10, use the BST Insert function (manually) to insert the 13 number sequentially to construct a binary search tree. Draw the final 13-node BST.

<answer>

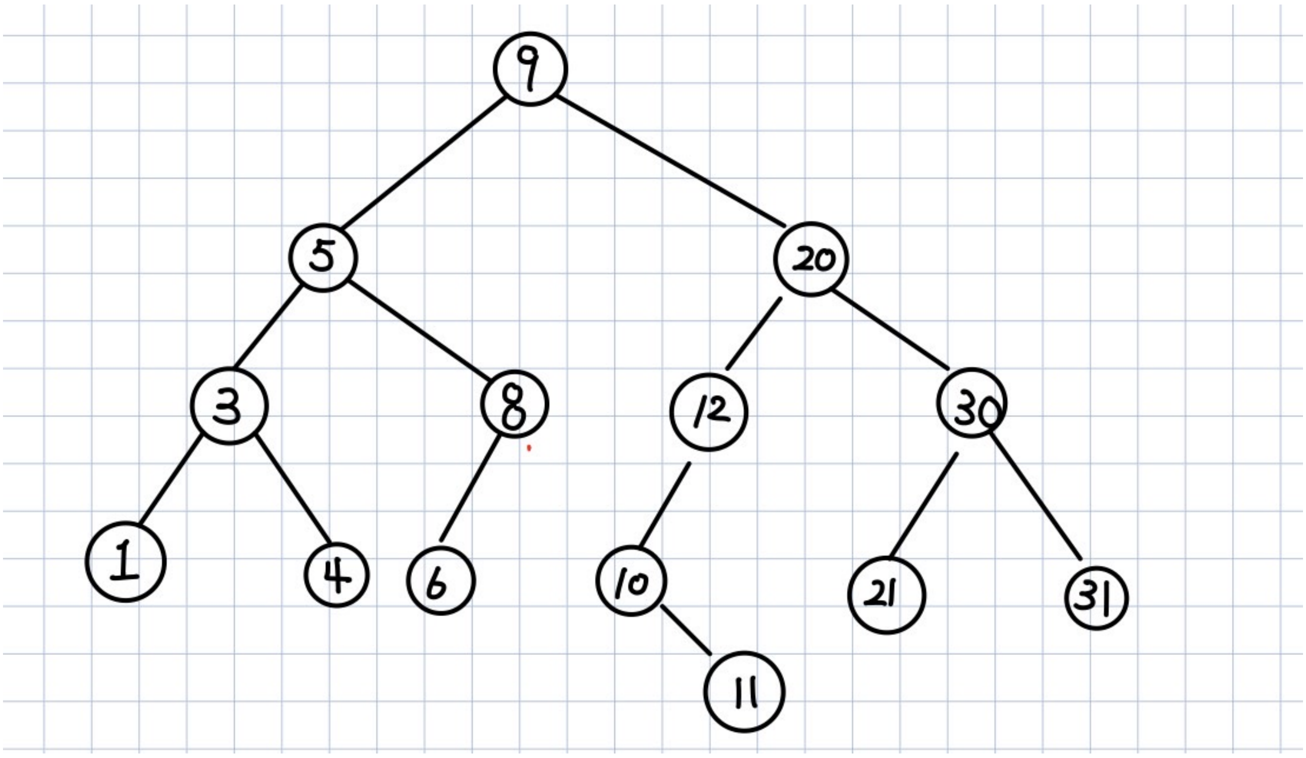


(e) A binary search tree produces the following preorder traversal, where “null” indicates an empty subtree (i.e. the left/right child is the null pointer).

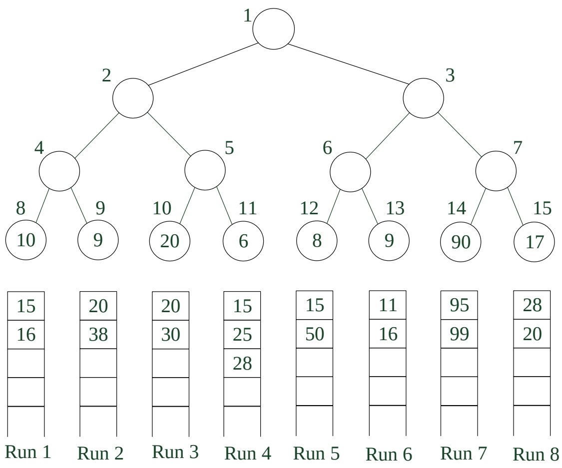
9,5,3,1,null,null,4,null,null,8,6,null,null,null,20,12,10,null,11,null,null,null,30,21,null,null,31,null,null

Draw the tree that produced this preorder traversal.

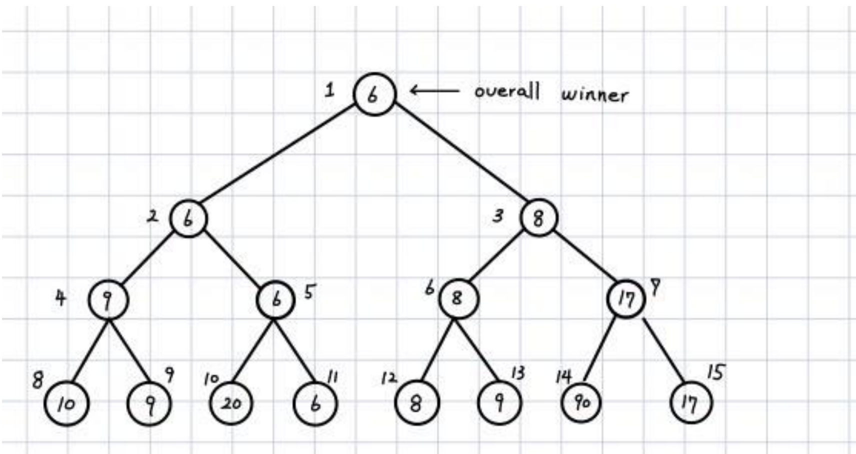
<answer>



9. (10%) An 8-run with total of 25 numbers are to be merged using Winner tree and Loser tree, respectively. The numbers of the 8 runs are shown below. The first numbers from each of the 8 runs have been placed in the leaf nodes of the tree as shown. Then these eight numbers enter the tournament to get the overall winner.



(a) Draw the winner tree and indicate the overall winner of this tournament.



(b) Draw the loser tree and indicate (draw) the overall winner of this tournament.