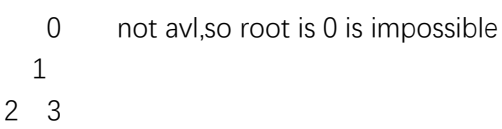
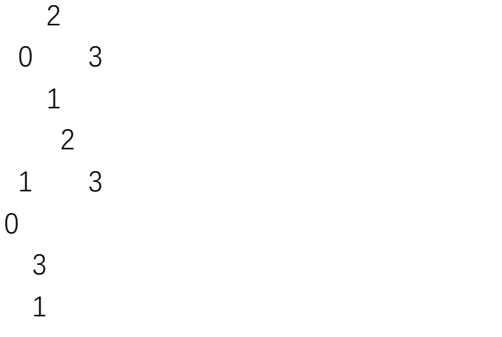
1. (a)

14

Catalan(n)=C(2n,n)/(n+1)。=70/5=14





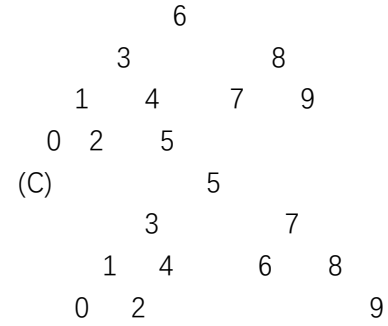
so there are 4 avl

(b)

Minimum and max are both 3

to find minimum we just need to build a complete binary tree

to find maximum we should find if we move the last node of right tree to left tree, will height be changed ? this example is no.



after rotation

```
          5
      3       8
    1   4     7   9
  0   2
```

most rotation is 1

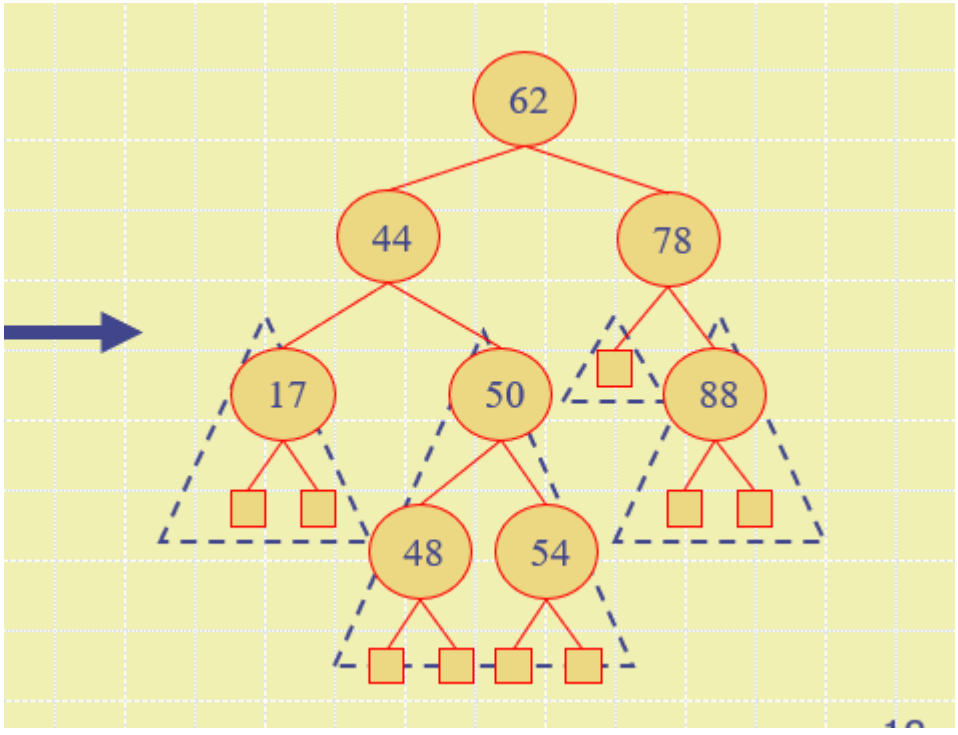
2,

(a)every insert is O logn

n insert is nlogn

(b) cause it is AVL tree, you can just delete in inorder which can make sure you move the element is sorted

for example



use inorder:you will remove in that order, 17,44,48,50,54,62,78,88

which is already sorted

(c) remove 1 is logn, remove all is nlogn

(d)nlogn

(e)O(n)

3.
(a) put(k,v) ,you need to compare the k with smallestkey returned by firstentry(), if k is smaller, you should replace smallestkey by k
remove(k),if the key you removed is the smallestkey,
case 1, the entry you remove is left node, you should replace the smallest key by the parent of your node,
case 2, the entry you remove is root, (in this case ,no left node, just have right node), you should use right subtree to replace the root which is removed, and update the smallest key by the root of right subtree
(b) Yes,
in previous question, though we have a sorted order, but we still need to use logn to search with firstEntry() and sorted order array, we can just loop the array ,index0, 1, 2, 3, ...
and use firstEntry to return the entry object, then we can directly get the place of smallest entry object. Then we don't need to search it.

4.
AVL(root tree)
int h=0

Height(root tree)
if tree.balance()=0&&tree.left=null
return 0 //base case,which means is external node
if tree.balance(n)=1 //left height is higher
return height(tree.left)+1
if tree.balance(n)=-1
return height(tree.right)+1
if tree.balance()=0,&&tree.left!=null
return height(tree.left)+1 //same
end Height(root tree)

h = Height(root tree)

return h
end AVL(root tree)

5. after mod
6,4,5,0,12,11,8,3,7,7,0,3,6,10,11,12,3,9

0	1	2	3	4	5	6	7	8	9	10	11	12
195			16	147	265	32	189	21	77	75	180	207
91			81			162	202		48		37	77

(2)collisions happend 7 times,and the biggest collision is 2

6

2.12,10,0,12,0,6,1,9,7,1,4,12,0,7,2,6,3

there are three 0s ,which means collision is bigger,

Reason: you should choose a prime number to mod in hash funciton, while 15=3*5 is not prime number

7. first every put mod 19(size of Array)

4,0,10,1,15,18,10,8,9

there is collission in put 48

11-48mod11=4

still collission

4+10=14

so we need to put 48 in 14

4,0,10,1,15,18,5,8,9

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
0	20			42				27	9	48				48	72			18

(b) 4

(c) 2

(d) 9/19