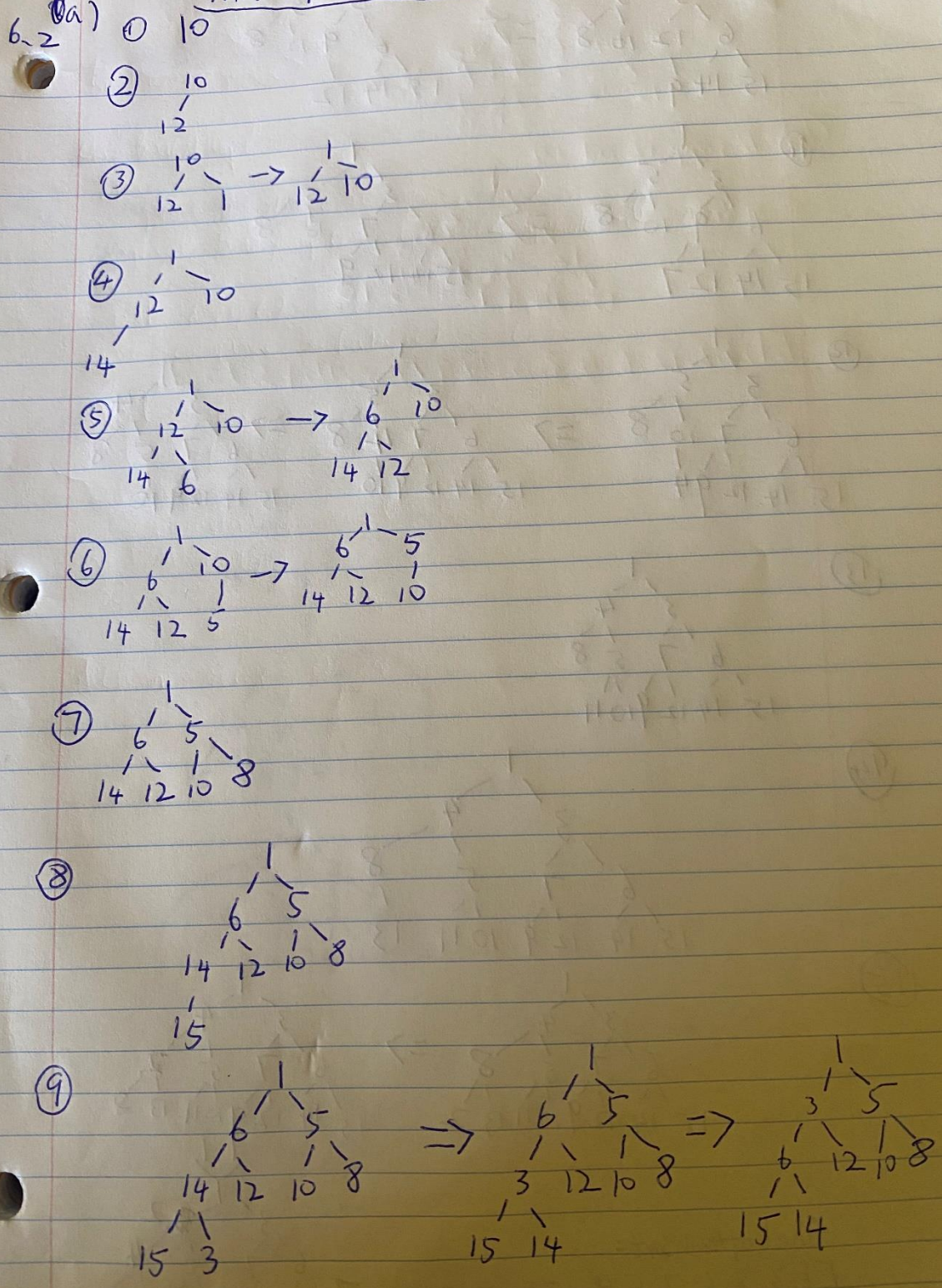
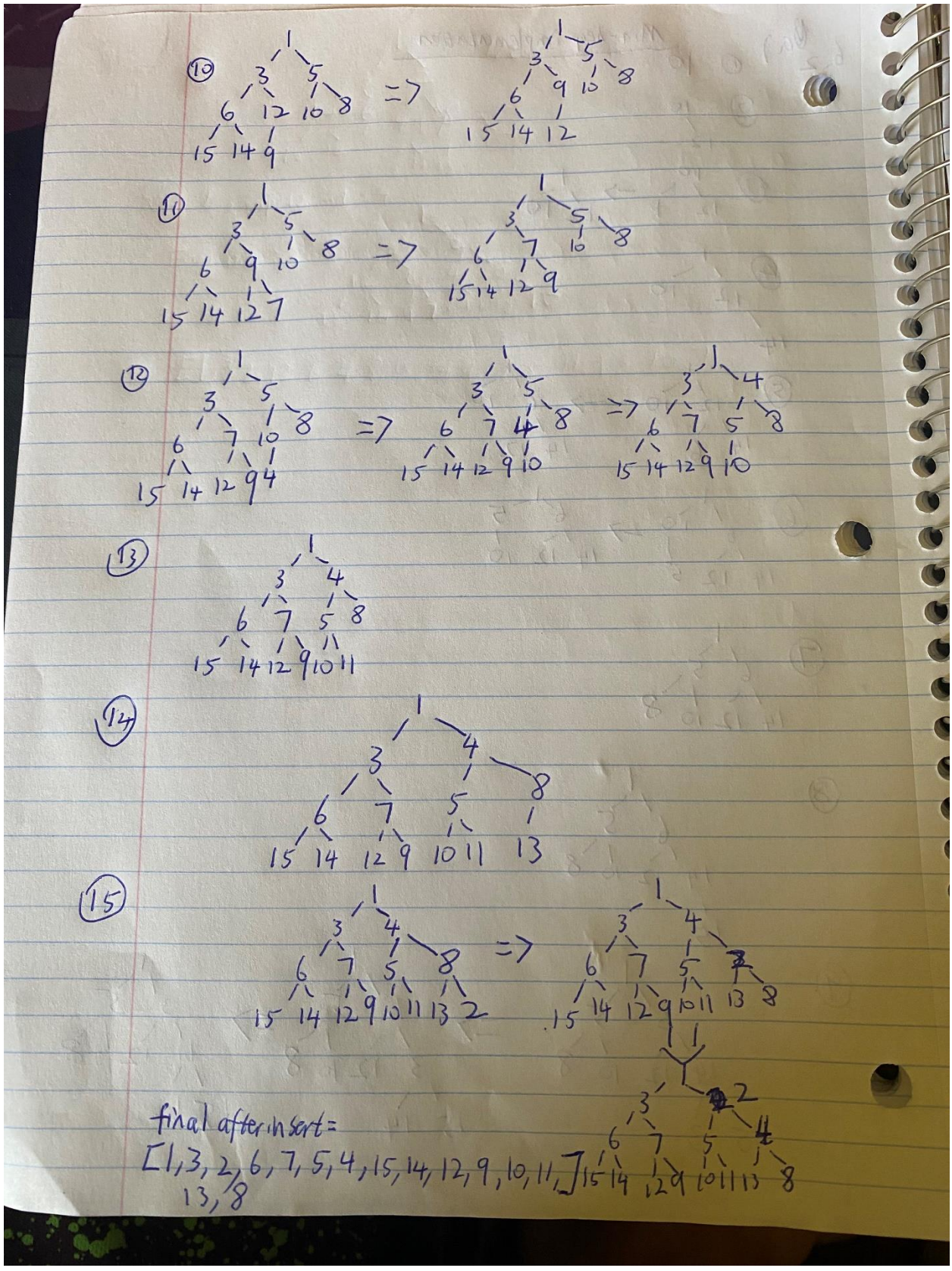
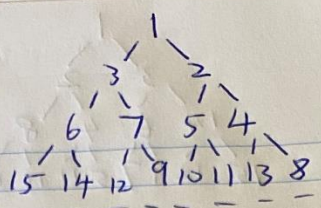
6_2 (a) Min-heap implementation




10
11
12
13
14
15
final after insert =
[1, 3, 2, 6, 7, 5, 4, 15, 14, 12, 9, 10, 11, 13, 8]

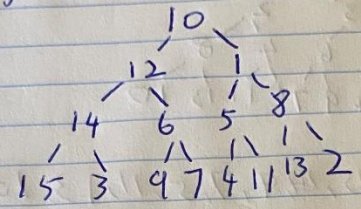
6.2a

Final tree min-heap:

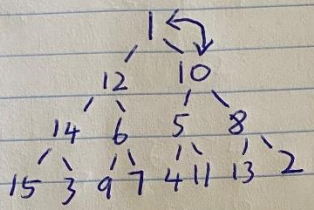


6.2b) linear time algorithm:

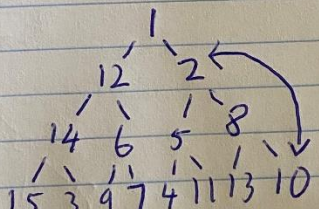
①



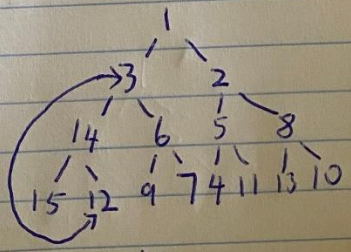
②



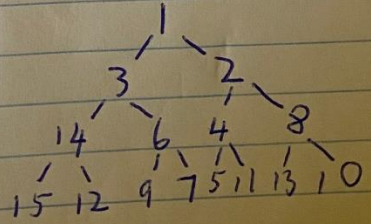
③



④

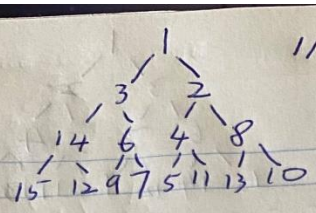


⑤



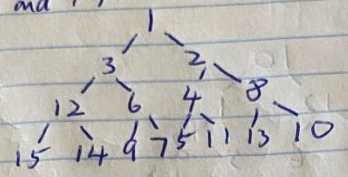
// swap 4 and 5

// copy from previous page



(6) // swap 12 and 14

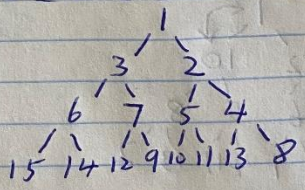
Final tree:



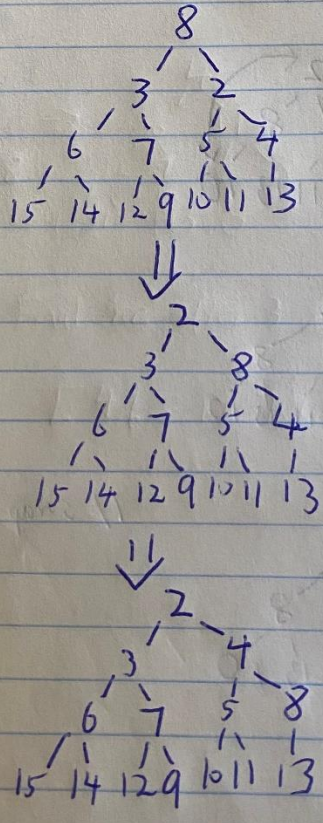
[1, 3, 2, 12, 6, 4, 8, 15, 14, 9, 7, 5, 11, 13, 10]

6.3 Part 1

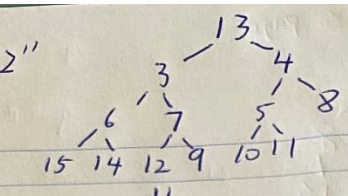
a) original tree:

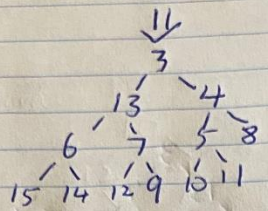


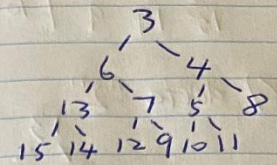
b) delete "1"



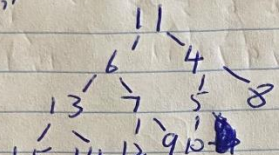
c) delete "2"



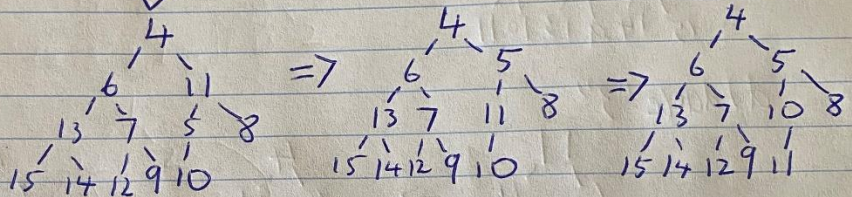




d) delete "3"



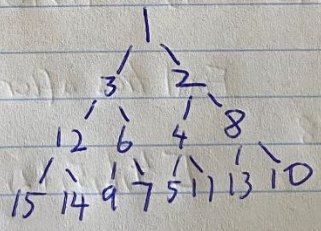
Final tree



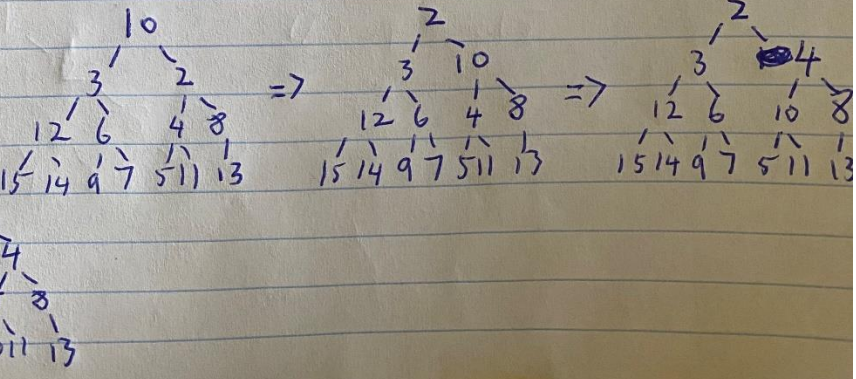
Min-heap after 3 delete min = [4, 6, 5, 13, 7, 10, 8, 15, 14, 12, 9, 11]

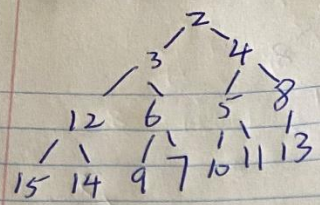
6.3 Part 2

a) original tree:

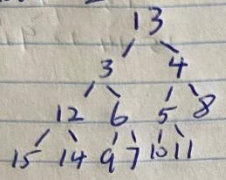


b) delete "1"

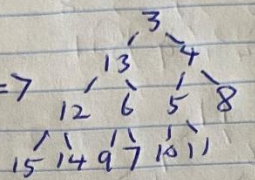




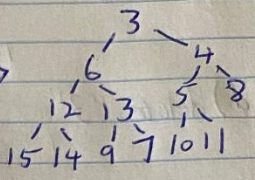
c) delete "2"



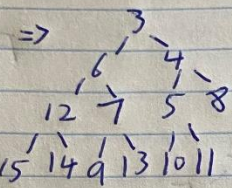
=>



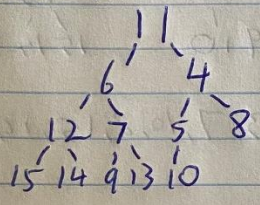
=>



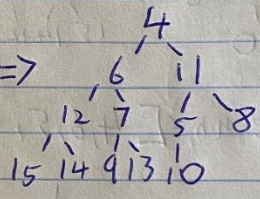
=>



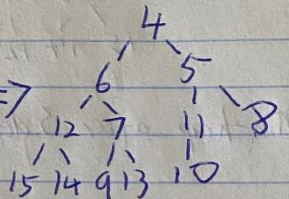
d) delete "3"



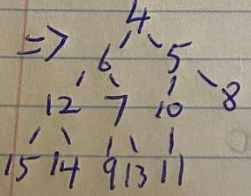
=>



=>



=>



Final tree after 3 deletions.

[4, 6, 5, 12, 7, 10, 8, 15, 14, 9, 13, 11]