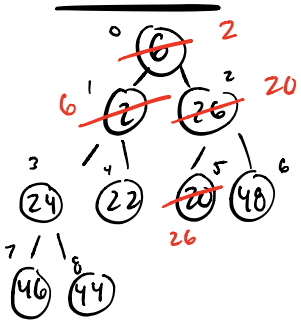
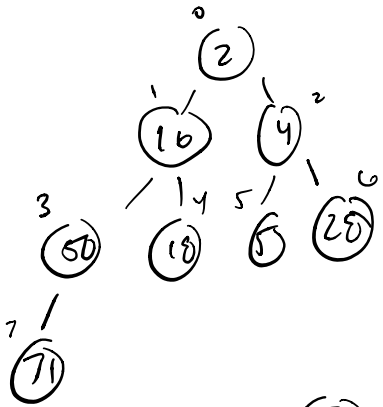
Part (a)



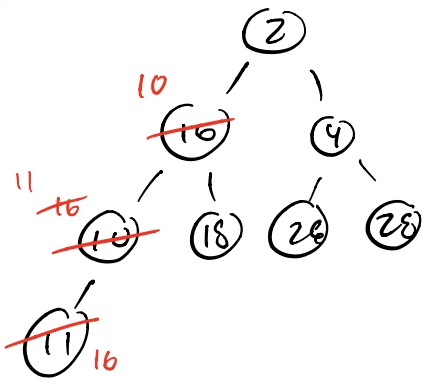
- 2 should be swapped with 6, because a child cannot be smaller than its parent in a min heap.
- 20 should be swapped with 26, because a child is smaller than its parent.

new array:

2	6	20	24	22	26	48	46	44



- this is a valid binary min heap
- every parent is smaller than its child node.

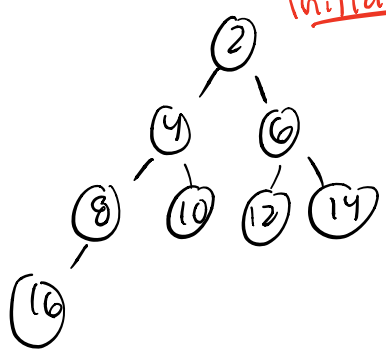


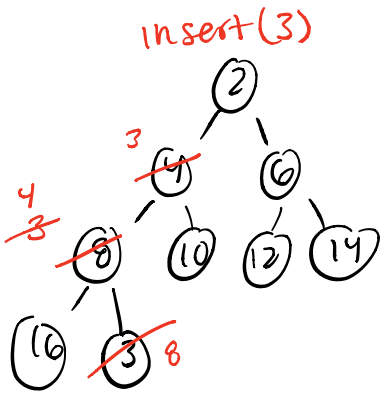
- not a valid min heap.
- 16 and 10 should be swapped
- then 11 and 16 should be swapped.

new array:

2	10	4	11	18	26	28	16

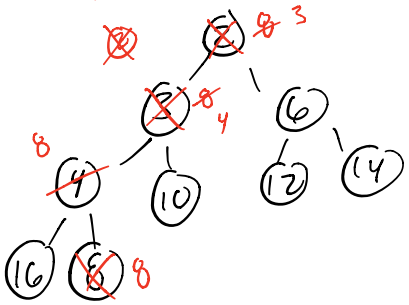
Part (b)





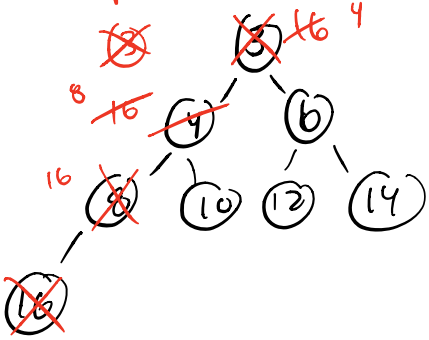
Step 1: insert bottom right
Step 2: swap parent/child 8 and 3
Step 3: swap 3 and 4

pop (top element)



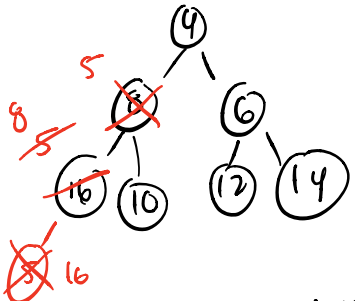
Step 1: remove bottom right
Step 2: remove 2 and replace w/ 8.
Step 3: swap 8 w/ smallest child 3
Step 4: swap 8 w smallest child 4
Step 5: 8 is smaller than 16 so it stays.

pop (top element)



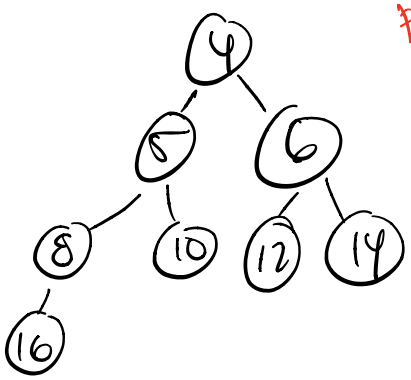
Step 1: take the last element 16. and remove it.
Step 2: make 16 top element and remove 3.
Step 3: swap w/ smallest element until 16 is in right place

Insert (5)

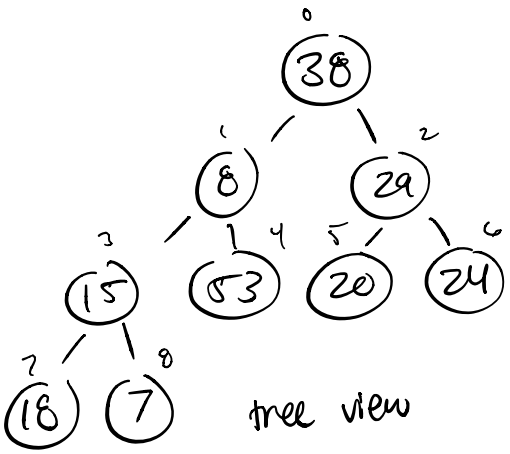


Step 1: make 5 last element.
Step 2: swap 16 and 5 because $5 < 16$
Step 3: swap 5 and 8 because $5 < 8$.

Final result



Part (C)



tree view

original array

38	8	29	15	53	20	24	18	7

Step 1: Start at first non-leaf node.
(7, 8, 4, 5, 6 all leaf nodes)

Step 2: heapify (3)
=> swap 15 and 7

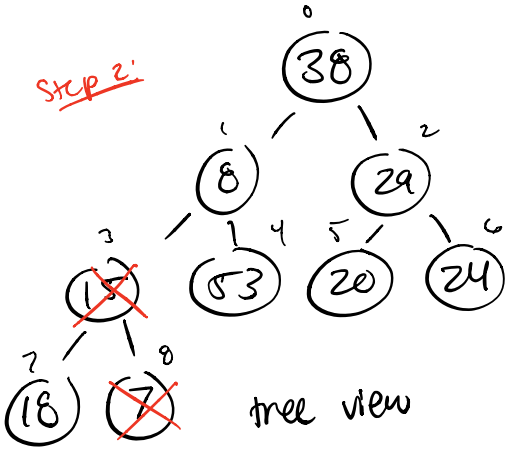
38	8	29	7	53	20	24	18	15

new array

Step 3: heapify (2)
=> swap 29 w/ 20

38	8	20	7	53	29	24	18	15

new array

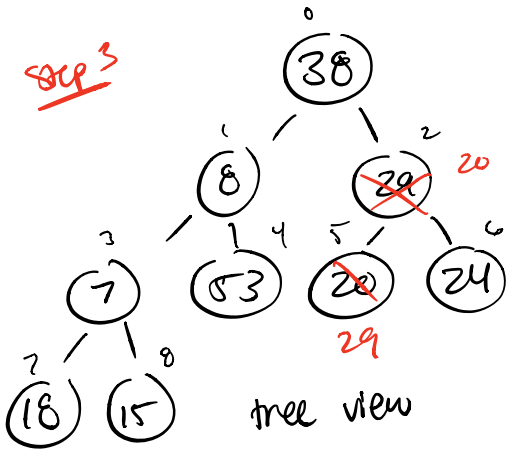


tree view

Step 4: heapify (1)
=> swap 7 and 8

38	7	20	8	53	29	24	18	15

new array

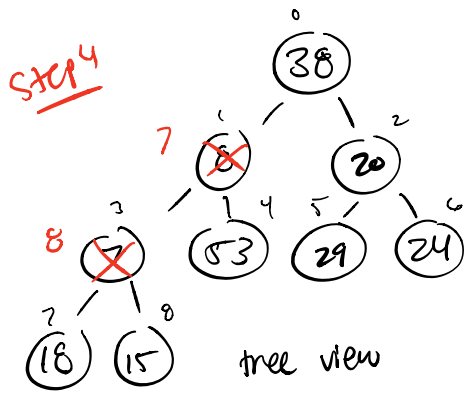


tree view

Step 5: heapify (0)
=> swap 38 w/ 7
=> swap 38 w/ 8
=> swap 38 w/ 15

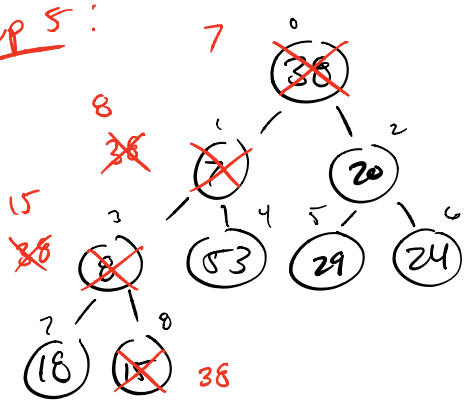
38	7	20	8	53	29	24	18	15

new array

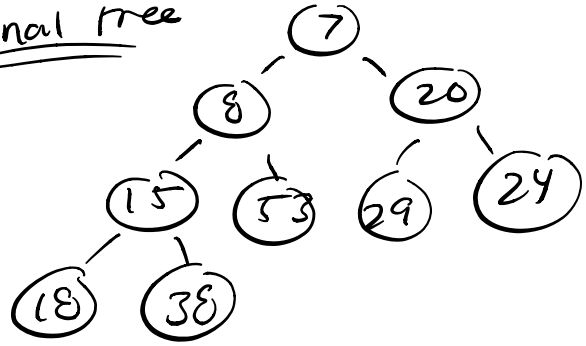


tree view

Step 5:



Final tree



Final array

7	8	20	15	53	29	24	18	38