Build Heap

1. Let index = length/2-1. This is the parent of the last node in the tree, i.e. list[index + 1] . . . list[length-1] are leaves
2. Convert the subtree with root of list[index] into a heap.
   a. Given list[a] is root of tree, list[b] is left child (root *2 +1), list[c] is right child (root*2+2), if exists
   b. Compare list[b] with list[c] to determine larger child, list[largerIndex]
   c. Compare list[a] with list[largerIndex]. If list[a] < list[largerIndex], then swap, else already a heap
   d. If swap, repeat step 2 for the subtree of list[largerIndex]
3. Convert the subtree with the root of list[index-1] into a heap, repeat until list[0]

Heap Sort

1. Swap the root with the end of the list.
2. Heapify the list up to but not including the root
3. Repeat until there is only one node in the

Simulate the heapsort algorithm manually to sort the array:

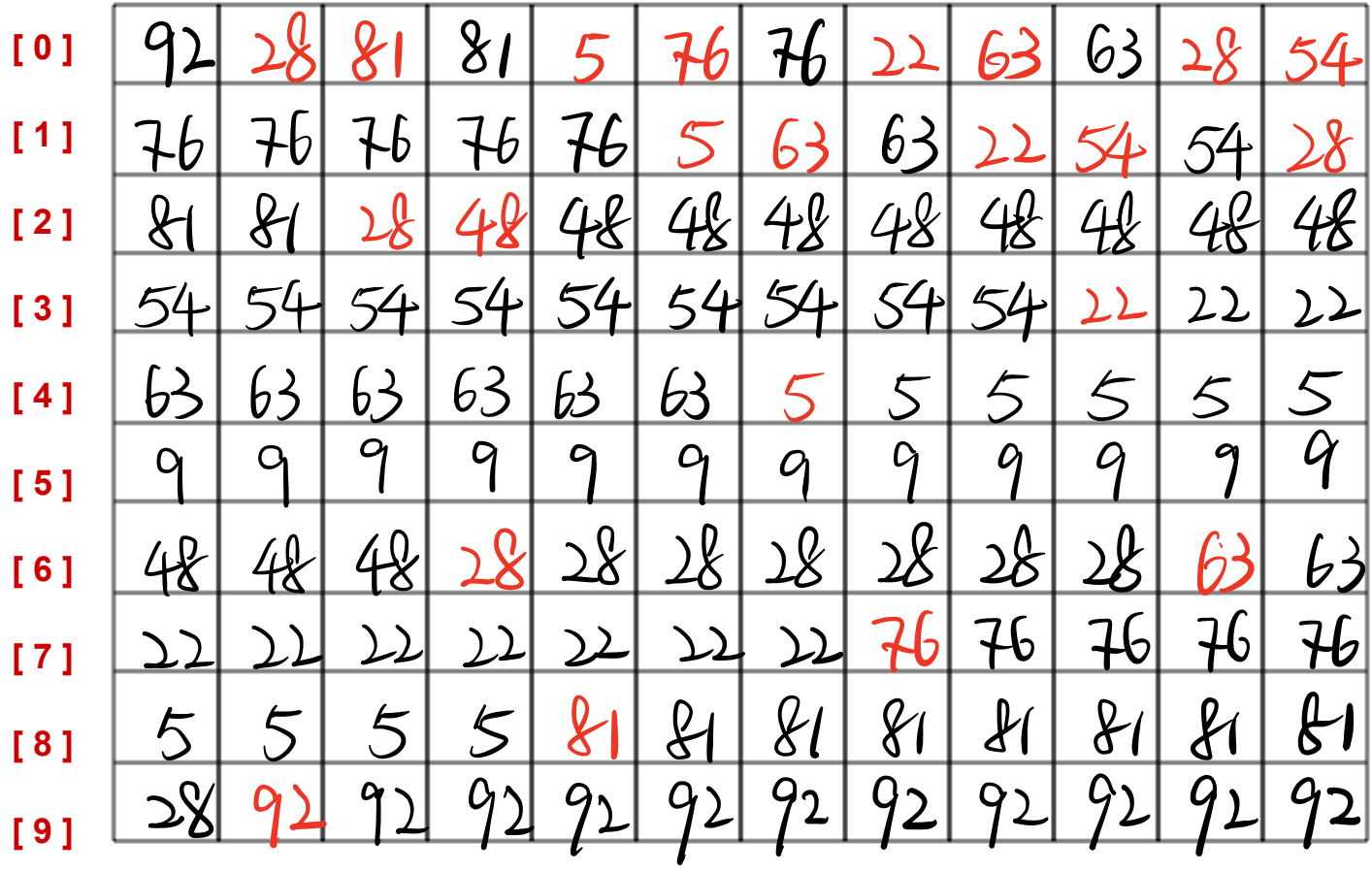
Liqin Wang.

Show all steps

1. Make into a heap
2. Sort

Max-Heap

[0]	5	5	5	5	5	92	92	92	92		
[1]	22	22	22	92	92	5	76	76	76		
[2]	9	9	81	81	81	81	81	81	81		
[3]	76	92	92	22	76	76	5	54	54		
[4]	63	63	63	63	63	63	63	63	63		
[5]	81	81	9	9	9	9	9	9	9		
[6]	48	48	48	48	48	48	48	48	48		
[7]	92	76	76	76	22	22	22	22	22		
[8]	54	54	54	54	54	54	54	5	5		
[9]	28	28	28	28	28	28	28	28	28		


[0] 92 28 81 81 5 76 76 22 63 63 28 54
[1] 76 76 76 76 76 5 63 63 22 54 54 28
[2] 81 81 28 48 48 48 48 48 48 48 48 48
[3] 54 54 54 54 54 54 54 54 54 22 22 22
[4] 63 63 63 63 63 63 5 5 5 5 5 5
[5] 9 9 9 9 9 9 9 9 9 9 9 9
[6] 48 48 48 28 28 28 28 28 28 28 63 63
[7] 22 22 22 22 22 22 22 76 76 76 76 76
[8] 5 5 5 5 81 81 81 81 81 81 81 81
[9] 28 92 92 92 92 92 92 92 92 92 92 92


[0]
[1]
[2]
[3]
[4]
[5]
[6]
[7]
[8]
[9]
9 48 5 28 28 5 22 9 5 5
28 28 28 5 22 22 5 5 9 9
48 9 9 9 9 9 9 22 22 22
22 22 22 22 5 28 28 28 28 28
5 5 48 48 48 48 48 48 48 48
54 54 54 54 54 54 54 54 54 54
63 63 63 63 63 63 63 63 63 63
76 76 76 76 76 76 76 76 76 76
81 81 81 81 81 81 81 81 81 81
92 92 92 92 92 92 92 92 92 92