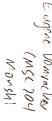


Build Heap

- 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 list



Simulate the heapsort algorithm manually to sort the array:

Show all steps

- 1. Make into a heap
- 2. Sort

										IVE	x-Hea
[0]	5					92			92		
[1]	22		· · ·	92		5	76	n Marry	76	110	
[2]	9		81				5	54	54		
[3]	76	92		22	76				76		
[4]	63		Þ						63	547	***
[5]	81		9		-	-1	7-		9		-
[6]	48				N .				48		
[7]	92	76			22				22		
[8]	54					1. 632		5	5		
[9]	28								28		

[0]	92	28	81		5	76		22	63		28	54
[1]	76					5	63		22	54	neth ma	28
[2]	81		28	48					- 1-			48
[3]	54		,							22		22
[4]	63				-		5	_				5
[5]	9					-						9
[6]	48	H	-	28)		63	63
[7]	12							76				76
[8]	5				81							81
[9]	28	92										92

											END	
[0]	54	9	48	5	28		5	22	9	5	5	_
[1]	28				5	22		5		9	9	
[2]	48		9			_			12		22	
[3]	22					5	28				28	
[4]	5			48							48	-
[5]	9	54									54	
[6]	63	-				_		-			63	
[7]	76							,			76	
[8]	81										81	
[9]	92							f 5			92	