Lab 9, Problems 2,3,4,5

Problem 2. Carry out the array-based version of HeapSort on the input array

[1, 4, 3, 9, 12, 2, 4]

Show steps and outputs along the way. Make sure to distinguish between Phase I and Phase II of the algorithm.

Solution.

Phase I

[1 | 4 3 9 12 2 4]

[1 4 | 3 9 12 2 4]

upheap

[4 1 | 3 9 12 2 4]

[4 1 3 | 9 12 2 4]

upheap (nothing to do)

[4 1 3 9 | 12 2 4]

upheap

[4 9 3 1 | 12 2 4] => [9 4 3 1 | 12 2 4]

[9 4 3 1 12 | 2 4]

upheap

[9 12 3 1 4 | 2 4] => [12 9 3 1 4 | 2 4]

[12 9 3 1 4 2 | 4]

upheap (nothing to do)

[12 9 3 1 4 2 4]

upheap

[12 9 4 1 4 2 3]

Phase II

[12 9 4 1 4 2 3]

[3 9 4 1 4 2 | 12]

downheap

[9 3 4 1 4 2 | 12] => [9 4 4 1 3 2 | 12]

[2 4 4 1 3 | 9 12]

downheap

[4 2 4 1 3 | 9 12] => [4 3 4 1 2 | 9 12]

[2 3 4 1 | 4 9 12]

downheap

[4 3 2 1 | 4 9 12]

[1 3 2 | 4 4 9 12]

downheap

[3 1 2 | 4 4 9 12]

[2 1 | 3 4 4 9 12]

downheap (nothing to do)

[1 | 2 3 4 4 9 12]

[1 2 3 4 4 9 12]

Problem 3. Carry out the steps of the recursive algorithm BottomUpHeap for the input sequence 11, 5, 2, 3, 17, 24, 1

Solution.

BUH([11, 5, 2, 3, 17, 24, 1])
k = 11 A = [5, 2, 3] B = [17, 24, 1]

BUH([5, 2, 3])
k = 5 A = [2] B = [3]

BUH([2]) => [2]

BUH([3]) => [3]

⇨ [5, 2, 3]
downheap
[5, 2 , 3]

BUH([17, 24, 1])
k = 17 A = [24] B = [1]

BUH([24]) => [24]

BUH([1]) => [1]

⇨ [17, 24, 1]
downheap
[24, 17 , 1]

=> [11, 5, 24, 2, 3, 17, 1]

downheap

[24, 5, 11, 2, 3, 17, 1] => [24, 5, 17, 2, 3, 11, 1]

4. Draw an example of a MaxHeap whose keys are all the odd numbers lie in [1, 21] (with no repeats), such that the insertion of an item with key 14 would cause up-heap to proceed all the way up to a child of the root (replacing that child's key with 14).