**Mason Shepherd**

**Homework 6: due March 8th 11:59PM.**

R-9.2 Suppose you set the key for each position *p* of a binary tree *T* equal to its preorder rank. Under what circumstances is *T* a heap? [5 points].

Ans:

So long as the Complete Binary Tree property is satisfied, *T* will be a heap since the Heap-Order property will be satisfied regardless via the preorder ranking of its keys.

// specify that each level but the last must be complete

R-9.7 Illustrate the execution of the selection-sort algorithm on the following input sequence: (22, 15, 36, 44, 10, 3, 9). [5 points].

Ans: Sequence S Priority Queue P

Input: (22, 15, 36, 44, 10, 3, 9) ()

Phase1:

(a) (15, 36, 44, 10, 3, 9) (22)

(b) (36, 44, 10, 3, 9) (22, 15)

…

(g) () (22, 15, 36, 44, 10, 3, 9)

Phase2:

(a) (3) (22, 15, 36, 44, 10, 9)

(b) (3, 9) (22, 15, 36, 44, 10)

(c) (3, 9, 10) (22, 15, 36, 44)

…

(f) (3, 9, 10, 15, 22, 36) (44)

(g) (3, 9, 10, 15, 22, 36, 44) ()

Ans:

W/ in-place selection sort...

(22, 15, 36, 44, 10, 3, 9)

(3, 22, 15, 36, 44, 10, 9)

(3, 9, 22, 15, 36, 44, 10)

…

(3, 9, 10, 15, 22, 36, 44)

R-9.19 Bill claims that a preorder traversal of a heap will list its keys in nondecreasing order. Draw an example of a heap that proves him wrong. [10 points].

Ans:

C-9.39 Explain how the *k* largest elements from an unordered collection of size *n* can be found in time *O*(*n* + *k* log *n*) using a maximum-oriented heap. [10 points].

Ans:

Use bottom-up construction to convert the list to a maximum-oriented heap where the element values = key { O(n) } and removeMax() *k* times { O(*k* log *n*) } … ergo O(*n* + *k* log *n*) time.

R-9.13 Illustrate the execution of the in-place heap-sort algorithm on the following input sequence: (2, 5, 16, 4, 10, 23, 39, 18, 26, 15). [10 points].

Ans:

(a) (2, 5, 16, 4, 10, 23, 39, 18, 26, 15)

(b) (2, 4, 5, 16, 10, 23, 39, 18, 26, 15)

(c) (2, 4, 5, 10, 16, 23, 39, 18, 26, 15)

(d) (2, 4, 5, 10, 16, 18, 23, 39, 26, 15)

(e) (2, 4, 5, 10, 16, 18, 23, 26, 39, 15)

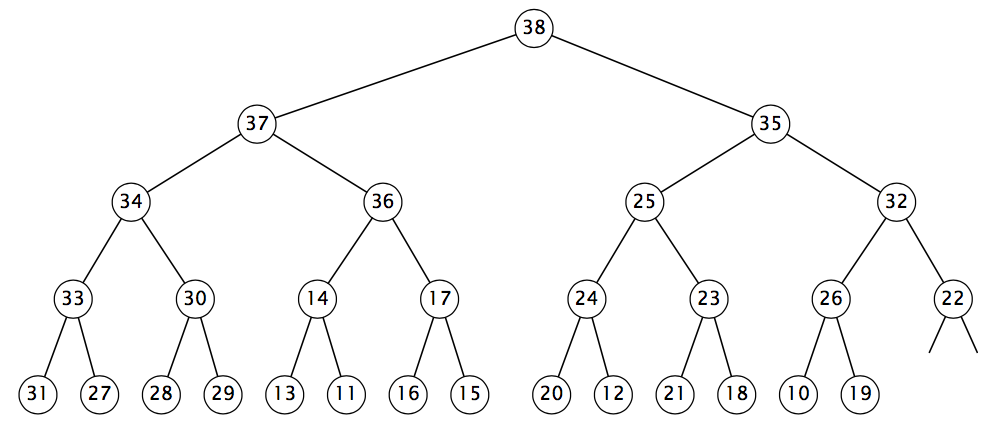
(f) (2, 4, 5, 10, 15, 16, 18, 23, 26, 39)

Ans:

Phase1: Construct maximum-oriented heap

Phase2: Remove root and down-heap

Additional question [10 points] Consider the following binary heap.



**(a)** Suppose that the last operation performed in the binary heap above was inserting the key x. Circle (highlight) all possible value of x:

10 11 12 13 14 15 16 17 18 19 20 21 22 23 24

25 26 27 28 29 30 31 32 33 34 35 36 37 38 39

// not 38 b/c 35 would have had to have been the root… < 37…

**(b)** Suppose that you delete the maximum key from the binary heap above. Circle (highlight) all keys that are involved in one (or more) compares.

10 11 12 13 14 15 16 17 18 19 20 21 22 23 24

25 26 27 28 29 30 31 32 33 34 35 36 37 38 39