CS566 Homework-3

Due – Feb 26, 6:00 PM, at the class beginning

Student's Name: _____

All work must be your own. Upload copy of your homework and any supporting materials (source code and test cases) online to Assignment3 in the CS566_A2 course site
No extensions or late submissions for anything other than major emergency.

(1)[15 pts.]
Design a stack that supports push, pop, and retrieving the minimum element in constant time.

(2) [15 pts] You wish to store a set of n numbers in either a max-heap or a sorted array. For each application below, state which data structure is better, or if it does not matter. Explain your answers.
(a) Want to find the maximum element quickly.
(b) Want to be able to delete an element quickly.
(c) Want to find median element quickly.

(3) [20 pts] Implement an external sort, which uses intermediate files to sort files bigger than main memory. What is the fastest external sort algorithm. What about Mergesort, Heapsort, Quicksort, and B-tree based sort algorithms for external sort. Test your algorithm within limits of time / power, RAM of your computer.

(4) [25 pts.] Given the following array:

26,5,77,1,61,11,59,15,48,19,81

Sort this array by hand using **Heapsort** and answer the following questions:
(a) Draw the initial COMPLETE tree representation of the file BEFORE the HEAP is created.
(b) Draw the initial heap.
(c) Continue with the sort for two iterations PAST the initial heap. How the heap looks like?

(5) [25 pts.] Use **Quicksort** to sort the following array using the first element as the pivot element:

72,20,73,42,11,80,39,30,100,46,88,32,21,1,13

(a) Continue sorting the entire array until totally sorted. During the entire sort – what was the maximum number of inversions removed by a single comparison and swap?
(b) What would be the sequence of the "best" pivot elements for this particular case?
(c) Does balanced Binary Search Tree helps?