MTH 4320 Homework 8

Yaohui Wu

April 9, 2024

Problem 1

Solution. We can use a min-heap and a global counter starting at zero for the key. When we push we increment the counter by one then assign it to be the key of the element and insert it into the heap. When we pop we decrement the counter by one then we remove the last node which is the top of the stack. The heap will always be in order since we maintain a counter for the keys. Therefore, the time complexity of push and pop is $O(1)$. ■

Problem 2

Solution. We can start at the parent of the last leaf in the max-heap. For every parent in the level we run a modified heapify for a min-heap so we swap the parent with the smallest children. We repeat this algorithm until we reach the root. Since the max-heap is ordered from greatest to least so after we run the algorithm then every parent of the new min-heap will be smaller than or equal to its children. The running time of building a heap is $O(n)$. There is no asymptotically faster algorithm because we have to visit all n nodes of the heap. Therefore, the time complexity of the algorithm is $O(n)$. ■

Problem 3

Solution. We use a max-heap and a min-heap, each containing about half of the elements, to implement the data structure.

- Insert: The time complexity is $O(\log n)$.
- Remove: The time complexity is $O(\log n)$.
- Find median: The time complexity is $O(1)$.

■