# Assignment #3 Written Report

The Heap Sort algorithm from lecture slides #17 and #18 was chosen as the sorting algorithm and implemented in this assignment. This algorithm was chosen because according to lecture slide #18, page 2, the Heap Sort algorithm sorts in place, using $O(n \log n)$ operations to sort an array with length $n$ in the worst case. Also, according to lecture slide #18, page 29, the additional storage required by this implementation of Heap Sort is in $O(\log n)$, including the sizes of call stacks whenever recursive methods are being used (and where $n$ is the length of the array to be sorted).

Thus, this implementation of Heap Sort satisfies all the requirements of a sorting algorithm that this assignment asks for, and that is why it was used for this assignment.

The only difference between my implementation of Heap Sort and the implementation shown in the lecture slides is that the insert algorithm from lecture slide #17, page 38 throws a HeapFullException

if the size of the MaxHeap is greater than or equal to the size of the array representing this MaxHeap, while my implementation of insert does not throw any exception in this case.

However, the worst case runtime and storage space used does not change, so the claims about the storage space and runtime bounds made in the lecture slides still hold.

Citation (APA Format):

- Eberly, W. (2019). CPSC 331: Data Structures, Algorithms and their Analysis, Lecture Notes #15, #17, #18 [Powerpoint Slides].