Assignment 3 report - CMPT 276 - Duc Duy Pham

Testing Algorithms: QuickSort and HeapSort
In testing the effectiveness of sorting algorithms, two algorithms were implemented and tested in this assignment: QuickSort and HeapSort, as they satisfy the requirement for O(log n) run time and in-place sorting.

I. Functional Testing

For both QuickSort and HeapSort, tests were designed to cover a variety of input conditions, including: arrays containing negative numbers, mixed positive and negative numbers, duplicates, elements of the same value, large numbers, arrays in sorted and reverse order, and strings.

Coverage Results:
For only functional testings, the coverage results for the entire maven project was 84.4%, with 13 different tests for both HeapSort and QuickSort. This shows that functional testing covered a broad variety of functionality that both sorting algorithms should cover. However, the remaining 16.6% lies on the structure of the code itself, hence, structural testing is needed.

II. Structural Testing

QuickSort: Structural tests for quickSort focus on the pivot as the smallest and largest elements, providing deeper coverage of the recursive partitioning algorithm. This ensured that the edge cases, which could reduce performance and correctness, were tested.

HeapSort: Structural tests for HeapSort target its heap construction and heapify process, crucial for its correctness and efficiency. Tests focus on handling diverse array conditions, including repeated elements and alternating values, ensuring the algorithm effectively maintains the heap property and performs reliably across various tests.

Coverage results and conclusion:
After adding the structural tests for Quicksort, the coverage for the entire project went up to 98.9%, showing that structural tests for HeapSort will fill up the rest of the test cases. After adding structural tests for HeapSort, the coverage went up to 98.8%, which shows that the tests covered almost every case in the project. As all the test cases were passed without any failures, we can conclude that the implementation of Quick and Heap sort is correct and can efficiently sort any given array.