CMPT 435 - Fall 2020 - Aaron Green

Assignment 2

Sorts Comparisons

October 3, 2020

1 Quick Sort

1.1 comparisons: 6860

The best asymptotic run time for quick sort is Big Omega 0(n*log n). The run times depends on the whether the partitioning is balanced or unbalanced. In the case of the best scenario which occurs when the pivot is always the middle or near it rather than the end elements.The worst case Big-O:0(n2) occurs if the if either the greatest or the smallest element is chosen. This is best represented by an array that contains n - 1 and the other one sub array is empty.

2 Merge Sort

2.1 comparisons: 5445

Merge Sort which runs on 0(n log n) merge relies on recurrence relation which is implemented through divide and conquer.It begins with a merge functions the run in O(n) time when it merges. The divide steps computes the midpoint of indices.The conquer will recursively sort the sub vectors I constructed with n/2.Then it finally combines giving its 0(n). The merge sort is the sum of merge time for all levels .Thus giving us our comparison equal to our run time. The worst case happens when the merge has to do the number of comparisons.

3 Insertion Sort

3.1 comparison: 114975

The insertion sort normally works best with already sorted data. But its Best Case Complexity O(n) occurs as the outer loop runs through my vector of magic items. the inner loop doesn't run at all.There are only a creating number due to specific insertion works. Worse Case could occur when the algorithm sort a list in the opposite order it is presented.

4 Selection Sort

4.1 comparison: 221445

The selection sort runs on 0(n2) the best case would occur on a item that's already sorted sort so that will explain the number of comparisons for the selection sort. The cycle of comparison for sorting begins with (n-1) and ends with 1 = n(n-1)/2 which would equal n2. The complexity can be analyzed using the loops as well as there are two the complexity is n*n = n2