Programming Assignment 2 Efficiency Report

Purpose

To track and record the efficiency of four sort types using loop and swap counters.

Implementation

Code was written in Microsoft Visual Studio, on a Windows 10 desktop PC with 16gb of ram, and an Intel i5 4690k processor. Class has an AssortedSorter.h class specification file, an AssortedSorter.cpp class implementation file, and a source.cpp file.

Execution Details

Programming Assignment #2 was executed ten times with values ranging from 2000 to 20000 in increments of 2000 being given as size declarations for arrays. Each of four sort methods was given a copy of this array and efficiency was tracked and recorded for each run, with total inputs, loop counts, and swap counts also being recorded for all sort methods. The sort methods were bubble sort, selection sort, shell sort, and quick sort.

Observations

Selection sort with a total of 109,900 swaps, had the fewest swaps. Bubble sort with a total of 384,855,241 swaps had the largest number of swaps. Quick sort with a total of 1,135,529 loops had the fewest number of loops. Bubble sort with a total of 1,525,401,221 loops had the largest number of loops.

Bubble sort agreed with Big-O notation coming very close to O(n^2) with each run. Selection sort was less than O(n^2) for each run. quick sort also agreed very closely with Big-O, coming very close to O(n log(n)) with each run. Shell sort was about 2 times O(n log(n)).

It’s interesting that with bubble sort, and selection sort having O(n^2), selection sort ran about half as many total loops as bubble sort. This is a similar situation with shell sort and quick sort both having O(n log(n)), but quick sort running about half as many loops as shell sort. Another interesting pattern that emerges is that both shell sort and selection sort will swap counts will be the same across multiple runs for any given number, while the other sorts swap counts vary. In addition, selection sort will have the same loop count across multiple runs for any given number while the other sorts loop counts vary.

Efficiency Ratings

Efficiency of sorts should be rated as such if swaps and loops carry the same weight

1. Quick Sort
2. Shell Sort
3. Selection Sort
4. Bubble Sort

Efficiency of sorts should be rated as such if swaps carry the more weight

1. Selection Sort
2. Quick Sort
3. Shell Sort
4. Bubble Sort

Efficiency of sorts should be rated as such if loops carry the more weight

1. Quick Sort
2. Shell Sort
3. Selection Sort
4. Bubble Sort

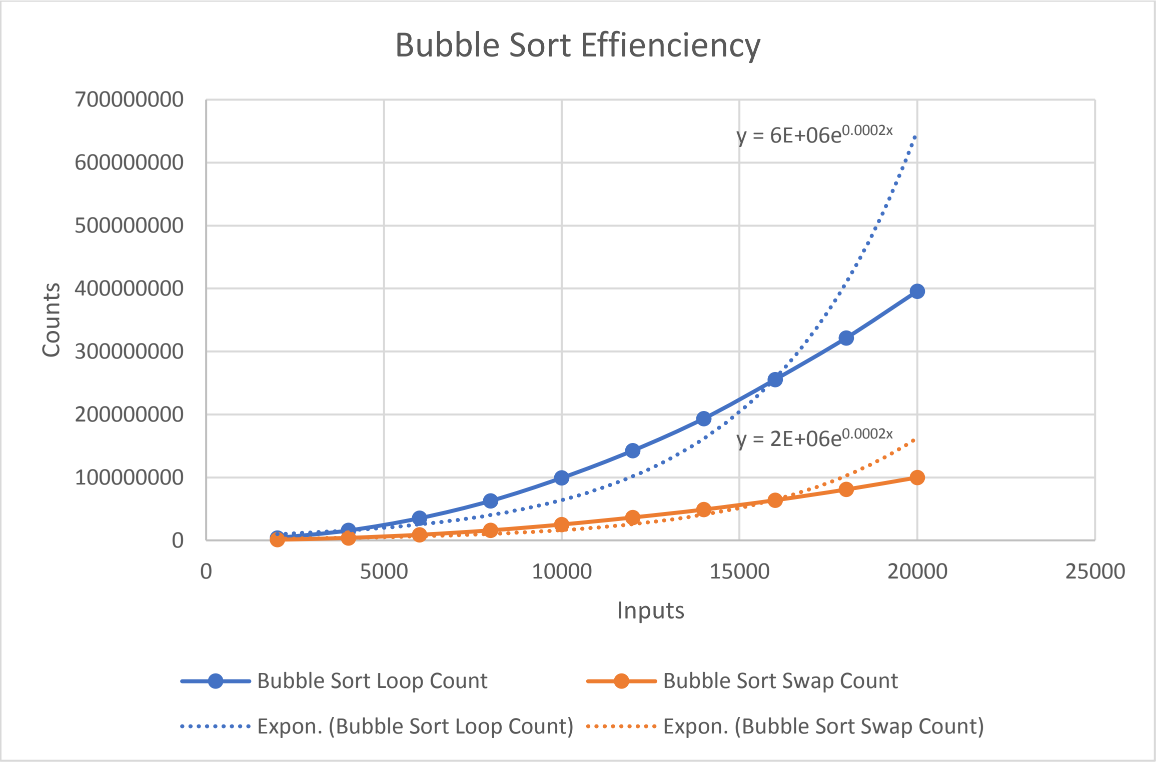
Below is an itemized list of sorts, loop counts, swap counts and charts for each sort method.

BubbleSort

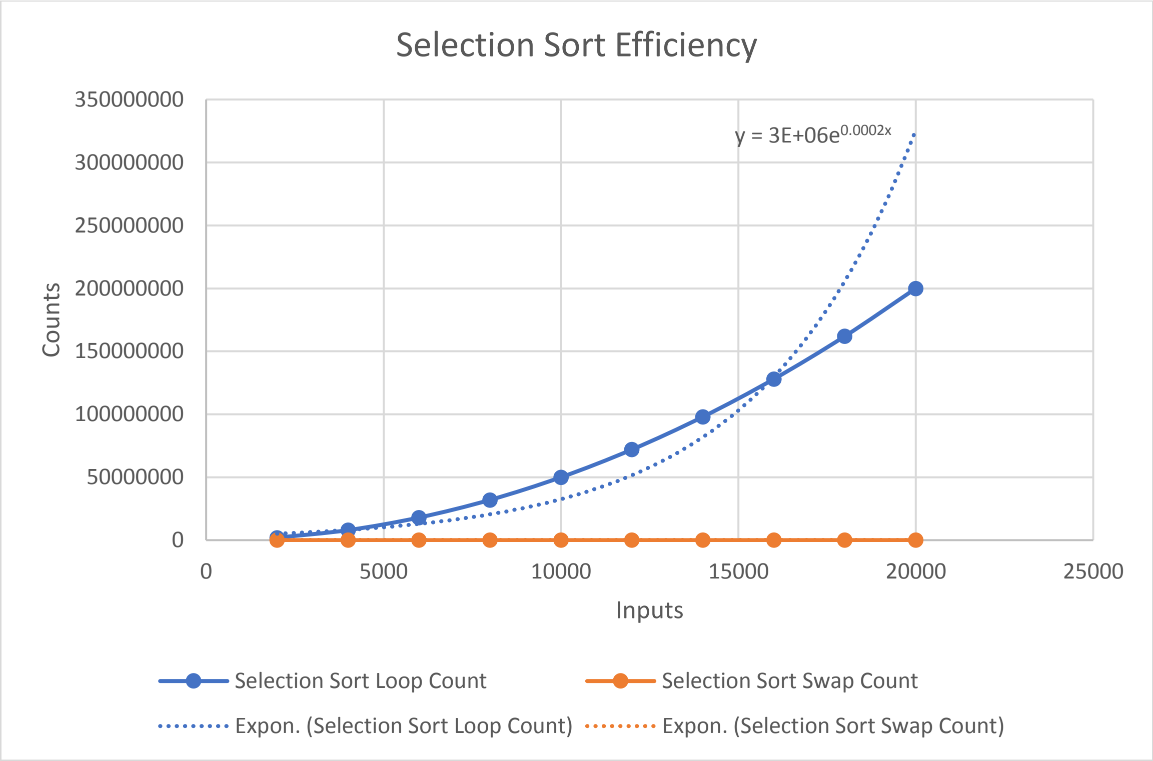
Input Value	Loop Count	Swap Count
2000	3,822,088	1,010,982
4000	15,688,077	3,978,531
6000	35,166,138	8,917,351
8000	62,680,164	15,914,936
10000	99,190,080	25,196,842
12000	142,536,121	36,371,946
14000	193,578,172	48,957,054
16000	255,440,034	63,800,303
18000	321,480,139	80,801,284
20000	395,820,208	99,906,012

Totals

Input	Loops	Swaps
110,000	1,525,401,221	384,855,241



Selection Sort		
Input Value	Loop Count	Swap Count
2000	1,999,000	1,999
4000	7,998,000	3,999
6000	17,997,000	5,999
8000	31,996,000	7,999
10000	49,995,000	9,999
12000	71,994,000	11,999
14000	97,993,000	13,999
16000	127,992,000	15,999
18000	161,991,000	17,999
20000	199,990,000	19,999
Totals		
Input	Loops	Swaps
110,000	769,945,000	109,990

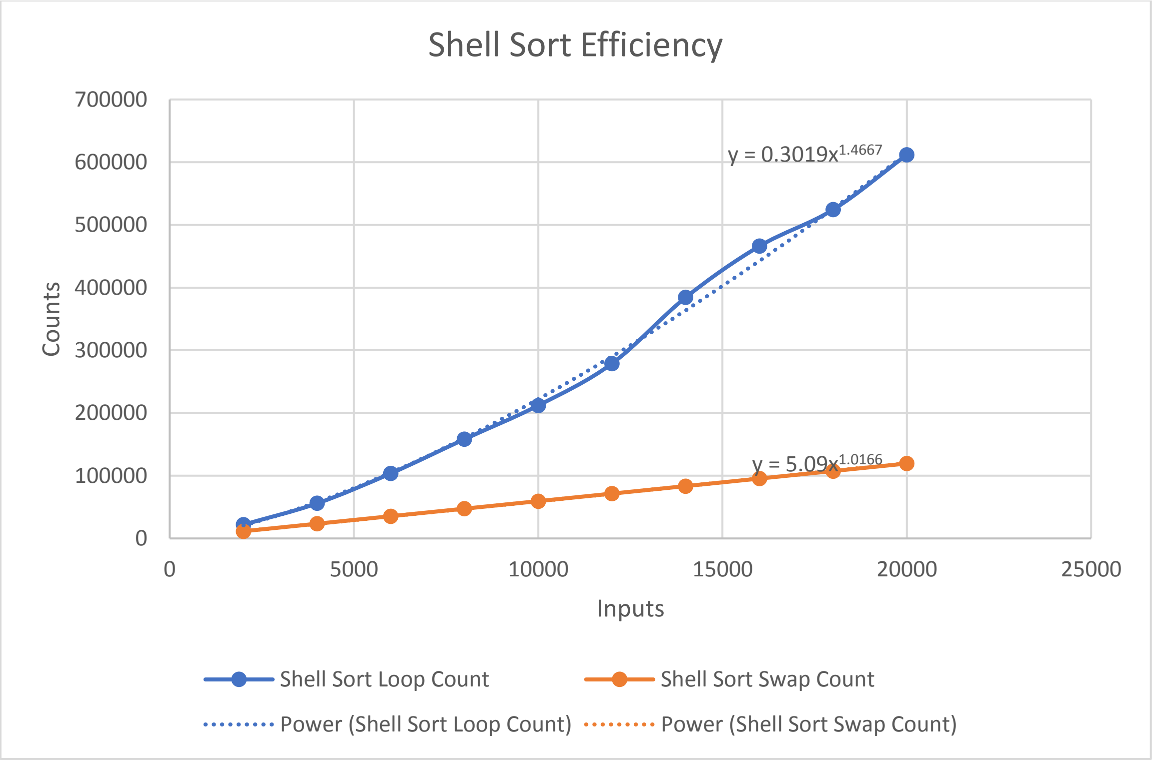


Shell Sort

Input Value	Loop Count	Swap Count
2000	21,926	11,457
4000	55,912	23,457
6000	103,713	35,457
8000	158,310	47,457
10000	212,011	59,457
12000	278,812	71,457
14000	384,284	83,457
16000	466,042	95,457
18000	524,269	107,457
20000	611,751	119,457

Totals

Input	Loops	Swaps
110000	2,817,030	654,570

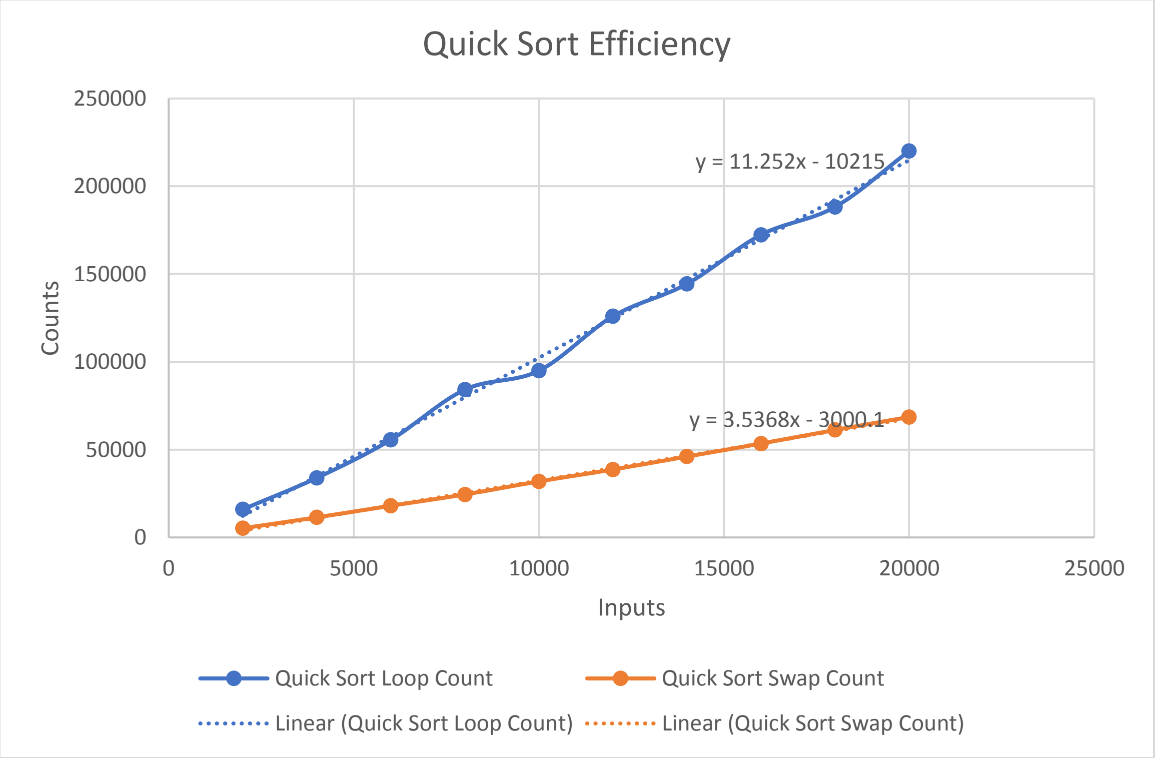


Quick Sort

Input Value	Loop Count	Swap Count
2000	15,959	5,241
4000	33,878	11,426
6000	55,572	18,036
8000	84,178	24,470
10000	94,932	31,899
12000	125,976	38,678
14000	144,381	46,065
16000	172,358	53,458
18000	188,216	61,188
20000	220,079	68,591

Totals

Input	Loops	Swaps
110,000	1,135,529	359,052


Quick Sort Efficiency
Counts
Inputs
y = 11.252x - 10215
y = 3.5368x - 3000.1
Quick Sort Loop Count
Quick Sort Swap Count
Linear (Quick Sort Loop Count)
Linear (Quick Sort Swap Count)