Behlül Gülmez

150140113

BLG 335E Project 1 Report

Q1.

PART 1: FULL SORT TIMES	INSERTION SORT	MERGE SORT
10K	1.23 seconds	0.01 seconds
100K	182.1 seconds	0.2 seconds
1M		2.96 seconds

PART 2: FILTER SORT TIMES	INSERTION SORT	MERGE SORT
10K	1.52 seconds	0 seconds
100K		0.009 seconds
1M		1.33 seconds

```
[gulmezbe@ssh Proje 1]$ g++ 150140113.cpp -o project1
[gulmezbe@ssh Proje 1]$ ./project1 -full -m hs-set-10k.txt hs-set-10k-full-m.tx
t
Time elapsed: 0.01 seconds
[gulmezbe@ssh Proje 1]$ ./project1 -full -m hs-set-100k.txt hs-set-100k-full-m.
txt
Time elapsed: 0.2 seconds
[gulmezbe@ssh Proje 1]$ ./project1 -full -m hs-set-1M.txt hs-set-1M-full-m.txt
Time elapsed: 2.96 seconds
[gulmezbe@ssh Proje 1]$ ./project1 -filter -m hs-set-10k.txt hs-set-10k-filter-
m.txt
Time elapsed: 0 seconds
[gulmezbe@ssh Proje 1]$ ./project1 -filter -m hs-set-100k.txt hs-set-100k-filte
r-m.txt
Time elapsed: 0.09 seconds
[gulmezbe@ssh Proje 1]$ ./project1 -filter -m hs-set-1M.txt hs-set-1M-filter-m.
txt
Time elapsed: 1.33 seconds
[gulmezbe@ssh Proje 1]$ ./project1 -full -i hs-set-10k.txt hs-set-10k-full-i.tx
t
Time elapsed: 1.23 seconds
```

Q2.

With a noticeable difference, merge sort is always superior because the worst case of the insertion sort algorithm is n ^ 2, but the worst case of the merge sort algorithm is nlogn. The merge sort algorithm is doing fewer operations also the difference between them increases as the size of the data increases. Thus the merge sort is much faster.

Q3.

Because there are so many kinds of names, more processing will be required in sorting by name. However it will be similar to type when sorting by rarity and set.

Q4.

Merge sort and insertion sort are stable algorithms. When we sort by full sort, if class, cost and name are same, other all features are same so there will be no change if we use different algorithm.