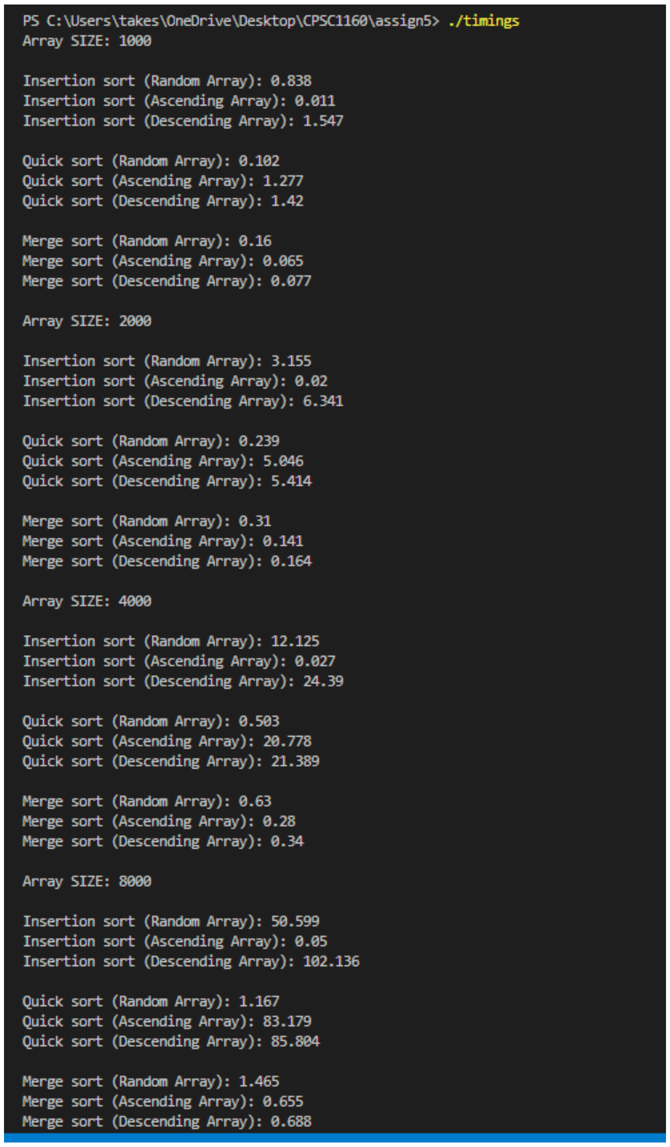
CPSC 1160 Assignment 5 Takeshi Hashimoto



```
PS C:\Users\takes\OneDrive\Desktop\CPSC1160\assign5> ./timings
Array SIZE: 1000

Insertion sort (Random Array): 0.838
Insertion sort (Ascending Array): 0.011
Insertion sort (Descending Array): 1.547

Quick sort (Random Array): 0.102
Quick sort (Ascending Array): 1.277
Quick sort (Descending Array): 1.42

Merge sort (Random Array): 0.16
Merge sort (Ascending Array): 0.065
Merge sort (Descending Array): 0.077

Array SIZE: 2000

Insertion sort (Random Array): 3.155
Insertion sort (Ascending Array): 0.02
Insertion sort (Descending Array): 6.341

Quick sort (Random Array): 0.239
Quick sort (Ascending Array): 5.046
Quick sort (Descending Array): 5.414

Merge sort (Random Array): 0.31
Merge sort (Ascending Array): 0.141
Merge sort (Descending Array): 0.164

Array SIZE: 4000

Insertion sort (Random Array): 12.125
Insertion sort (Ascending Array): 0.027
Insertion sort (Descending Array): 24.39

Quick sort (Random Array): 0.503
Quick sort (Ascending Array): 20.778
Quick sort (Descending Array): 21.389

Merge sort (Random Array): 0.63
Merge sort (Ascending Array): 0.28
Merge sort (Descending Array): 0.34

Array SIZE: 8000

Insertion sort (Random Array): 50.599
Insertion sort (Ascending Array): 0.05
Insertion sort (Descending Array): 102.136

Quick sort (Random Array): 1.167
Quick sort (Ascending Array): 83.179
Quick sort (Descending Array): 85.804

Merge sort (Random Array): 1.465
Merge sort (Ascending Array): 0.655
Merge sort (Descending Array): 0.688
```

Insertion Sort

	Random	Ascending	Descending
Size 1000:	0.838	0.011	1.547
Size 2000:	3.155	0.02	6.341
Size 4000:	12.125	0.027	24.39
Size 8000:	50.599	0.05	102.136

Random:

$0.838 / 1000^2 = 0.00000084 = 8.4 * 10^{-7}$

$3.155 / 2000^2 = 0.00000079 = 7.9 * 10^{-7}$

$12.125 / 4000^2 = 0.00000076 = 7.6 * 10^{-7}$

$50.599 / 8000^2 = 0.00000079 = 7.9 * 10^{-7}$

Insertion Sort with Random Array

$O(n^2)$

$7.9 * 10^{-7} n^2$

Ascending:

$0.011 / 1000 = 0.000011 = 1.1 * 10^{-5}$

$0.02 / 2000 = 0.00001 = 1.0 * 10^{-5}$

$0.027 / 4000 = 0.00000675 = 6.75 * 10^{-6}$

$0.05 / 8000 = 0. 00000625 = 6.25 * 10^{-6}$

Insertion Sort with Ascending Array

$O(n)$

$6.25 * 10^{-6} n$

Descending:

$1.547 / 1000^2 = 0.0000015 = 1.5 * 10^{-6}$

$6.341 / 2000^2 = 0.0000016 = 1.6 * 10^{-6}$

$24.39 / 4000^2 = 0.0000015 = 1.5 * 10^{-6}$

$102.136 / 8000^2 = 0.0000016 = 1.6 * 10^{-6}$

Insertion Sort with Descending Array

$O(n^2)$

$1.6 * 10^{-6} n^2$

Best Case: Ascending $O(n)$,

Worst Case: Descending $O(n^2)$

Average Case: Random $O(n^2)$

Quick Sort

	Random	Ascending	Descending
Size 1000:	0.102	1.277	1.42
Size 2000:	0.239	5.046	5.414
Size 4000:	0.503	20.778	21.389
Size 8000:	1.167	83.179	85.804

Random:

$0.102 / 1000 \log1000 = 0.00001 = 1.0 * 10^{-5}$

$0.239 / 2000 \log2000 = 0.000011 = 1.1 * 10^{-5}$

$0.503 / 4000 \log4000 = 0.000011 = 1.1 * 10^{-5}$

$1.167 / 8000 \log8000 = 0.000011 = 1.1 * 10^{-5}$

Quick Sort with Random Array

O(nlogn)

$1.1 * 10^{-5}$ nlogn

Ascending:

$1. 277 / 1000^2 = 0.0000013 = 1.3 * 10^{-6}$

$5.046 / 2000^2 = 0.0000013 = 1.3 * 10^{-6}$

$20.778 / 4000^2 = 0.0000013 = 1.3 * 10^{-6}$

$83.179 / 8000^2 = 0. 0000013 = 1.3 * 10^{-6}$

Quick Sort with Ascending Array

O(n)

$1.3 * 10^{-6}$ n

Descending:

$1.42 / 1000^2 = 0.0000014 = 1.4 * 10^{-6}$

$5.414 / 2000^2 = 0.0000016 = 1.4 * 10^{-6}$

$21.389 / 4000^2 = 0.0000015 = 1.3 * 10^{-6}$

$85.804 / 8000^2 = 0.0000016 = 1.3 * 10^{-6}$

Quick Sort with Descending Array

$O(n^2)$

$1.3 * 10^{-6} n^2$

Best Case, Average Case: Random O(nlogn),
Worst Case: Ascending, Descending $O(n^2)$

Merge Sort

	Random	Ascending	Descending
Size 1000:	0.16	0.065	0.077
Size 2000:	0.31	0.141	0.164
Size 4000:	0.63	0.28	0.34
Size 8000:	1.465	0.655	0.688

Random:

$0.16 / 1000 \log 1000 = 0.000016 = 1.6 * 10^{-5}$

$0.31 / 2000 \log 2000 = 0.000014 = 1.4 * 10^{-5}$

$0.63 / 4000 \log 4000 = 0.000013 = 1.3 * 10^{-5}$

$1.465 / 8000 \log 8000 = 0.000014 = 1.4 * 10^{-5}$

Merge Sort with Random Array

$O(n\log n)$

$1.4 * 10^{-5}\ n\log n$

Ascending:

$0.065 / 1000 \log 1000 = 0.0000065 = 6.5 * 10^{-6}$

$0.141 / 2000 \log 1000 = 0.0000045 = 6.4 * 10^{-6}$

$0.28 / 4000 \log 4000 = 0.0000059 = 5.9 * 10^{-6}$

$0.655 / 8000 \log 8000 = 0.\ 0000063 = 6.3 * 10^{-6}$

Merge Sort with Ascending Array

$O(n)$

$6.3 * 10^{-6}\ n$

Descending:

$0.077 / 1000 \log 1000 = 0.0000077 = 7.7 * 10^{-6}$

$0.164 / 2000 \log 1000 = 0.0000075 = 7.5 * 10^{-6}$

$0.34 / 4000 \log 4000 = 0.0000071 = 7.1 * 10^{-6}$

$0.688 / 8000 \log 8000 = 0.0000066 = 6.6 * 10^{-6}$

Merge Sort with Descending Array

$O(n^2)$

$6.6 * 10^{-6}\ n^2$

Best Case, Worst Case, Average Case: Random O(nlogn)

Discussion

Insertion Sort

Best case is when the array is already sorted. Because we can stop when the value we are moving don't need to be inserted.
Worst case is the array is reversed. Because we need to insert the value until it goes to the left most point.
Average case is when inserting stops at the middle of the array.

Quick Sort

Best case is when the array is random. Because We want to split the array half by using the left value as a pivot. If the values are random, pivot is likely not too big, too small.
Worst case is when array is sorted regardless of ascending or descending. Because when we choose pivot, we don't want to have extreme values like smallest or largest
Average case is also random case

Merge Sort

Best case, Worse case, and Average case are the same. Because we divide the array until the length will be 1. We don't care about the order of value.

Additionally, when we use random array, the time could be longer because creating random array takes more time than ascending or descending.