Week 7 Experiment report

Zhengyue LIU z5036602

Ascending ordered data tested on Naïve quick sort

T(n)	Average	Measurements		
10000	0.29	0.285	0.302	0.293
20000	1.00	1	1.004	1
30000	2.18	2.18	2.1	2.25
40000	3.92	3.94	3.97	3.84
50000	6.11	6.13	6.09	6.1
60000	8.56	8.47	8.7	8.5
70000	11.55	11.75	11.3	11.6
80000	15.30	15.28	15.32	15.3
90000	18.79	18.81	19.13	18.443
100000	22.63	22.75	22.34	22.8

OVERSIZE

Fig.1

Analysis:
Fig.1 illustrates that Naïve quick sort shows O(n^2) time complexity on sorting Ascending-ordered data. This is the worst case for naïve quick sort. And the program is very time consuming. When data size up to 100000, the program takes around 22 seconds to finish the sorting.

Descending ordered data tested on Naïve quick sort

T(n)	Average	Measurements		
10000	0.30	0.3	0.302	0.298
20000	1.00	1	0.99	1
30000	2.22	2.275	2.14	2.25
40000	3.88	3.85	3.93	3.87
50000	5.91	5.97	5.8	5.96
60000	8.66	8.62	8.7	8.66
70000	11.67	11.72	11.7	11.6
80000	15.20	15.28	15.11	15.2
90000	18.78	18.81	19.1	18.443
100000	23.41	23.75	23.33	23.14

OVERSIZE

Fig.2

Analysis:
Fig.2 illustrates that Naïve quick sort shows O(n^2) time complexity on sorting Descending-ordered data. And the program shows almost same time complexity behaviour as sorting on Ascending ordered data.

Un-ordered data tested on Naïve quick sort

T(n)	Average	Measurements		
10000	0.00	0	0	0
20000	0.010	0.01	0.01	0.01
30000	0.016	0.016	0.015	0.017
40000	0.024	0.024	0.024	0.0243
50000	0.0297	0.030295	0.028	0.0307
60000	0.0358	0.0368	0.0358	0.0348
70000	0.0415	0.0403	0.04288	0.041366
80000	0.0467	0.0476	0.044788	0.04765
90000	0.0515	0.0496	0.053	0.0519
100000	0.0578	0.058	0.057	0.058366
120000	0.067566667	0.0607	0.064	0.078
140000	0.0773	0.078	0.077	0.077
160000	0.0827	0.083	0.082	0.083
180000	0.0953	0.096	0.095	0.095

OVERSIZE

Fig.3

Analysis:
Fig.3 illustrates that Naïve quick sort shows O(n) time complexity on sorting un-ordered data. And the program becomes much faster on sorting un-ordered data. Even the data size up to 180000, the program takes only 0.1 second to finish the sorting.

Ascending ordered data tested on Median-3 quick sort

T(n)	Average	Measurements		
100000	0.031	0.031	0.031	0.031
200000	0.058	0.058	0.058	0.058
300000	0.079	0.079	0.079	0.08
400000	0.107	0.1	0.11	0.11

600000	0.126	0.126	0.125	0.126
700000	0.153	0.16	0.15	0.15
800000	0.166	0.171	0.168	0.16
900000	0.175	0.174	0.177	0.174
1000000	0.185	0.187	0.185	0.184

OVERSIZE

Fig.4

Analysis:
Fig.4 illustrates that Median-3 quick sort shows O(n) time complexity on sorting Ascending-ordered data. The program is much faster than naïve quick sort on sorting Ascending-ordered data.

Descending ordered data tested on Median-3 quick sort

T(n)	Average	Measurements		
100000	0.0327	0.033	0.033	0.032
200000	0.0603	0.062	0.059	0.06
300000	0.0817	0.084	0.079	0.082
400000	0.1133	0.12	0.11	0.11
600000	0.1400	0.145	0.13	0.145
700000	0.1533	0.14	0.16	0.16
800000	0.1750	0.17	0.177	0.178
900000	0.1853	0.186	0.185	0.185
1000000	0.1893	0.192	0.188	0.188

OVERSIZE

Fig.5

Analysis:
Fig.5 illustrates that Median-3 quick sort shows O(n) time complexity on sorting Descending-ordered data. The program is much faster than naïve quick sort on sorting Descending-ordered data

Un-ordered data tested on Median-3 quick sort

T(n)	Average	Measurements		
100000	0.06	0.062	0.042	0.0616
200000	0.11	0.1	0.11	0.115
300000	0.15	0.14	0.15	0.15
400000	0.18	0.18	0.18	0.17
500000	0.21	0.2	0.21	0.21
600000	0.25	0.24	0.25	0.25
700000	0.28	0.27	0.28	0.283

800000	0.31	0.3	0.31	0.31
900000	0.33	0.33	0.33	0.34
1000000	0.37	0.37	0.38	0.37

OVERSIZE

Fig.6

Analysis:
Fig.6 illustrates that Median-3 quick sort shows O(n) time complexity on sorting un-ordered data. The program has similar speed to naïve quick sort on sorting un-ordered data.

Ascending ordered data tested on Randomised pivot quick sort

T(n)	Average	Measurements		
100000	0.20	0.2	0.2	0.21
200000	0.35	0.29	0.41	0.35
300000	0.43	0.46	0.42	0.418
400000	0.58	0.54	0.68	0.51
500000	0.73	0.63	0.87	0.701
600000	0.86	0.91	0.85	0.82
700000	0.96	0.97	0.85	1.05
800000	1.13	1.12	1.1	1.18
900000	1.21	1.3	1.26	1.07
1000000	1.37	1.3	1.225	1.57

OVERSIZE

Fig.7

Analysis:
Fig.7 illustrates Randomised pivot quick sort shows O(n) time complexity on sorting Ascending-ordered data. The program has similar speed to Median-3 quick sort on sorting Ascending-ordered data. The program is much faster than naïve quick sort on sorting Ascending-ordered data.

Descending ordered data tested on Randomised pivot quick sort

T(n)	Average	Measurements		
100000	0.20	0.2	0.2	0.21
200000	0.34	0.37	0.32	0.34
300000	0.43	0.41	0.43	0.45
400000	0.55	0.527	0.52	0.59

500000	0.71	0.63	0.77	0.72
600000	0.90	0.9	0.98	0.82
700000	0.99	1	0.96	1
800000	1.11	1.11	1.11	1.11
900000	1.25	1.37	1.32	1.07
1000000	1.39	1.38	1.225	1.57

OVERSIZE

Fig.8

Analysis:
Fig.8 illustrates Randomised pivot quick sort shows O(n) time complexity on sorting Descending-ordered data. The program has similar speed to Median-3 quick sort on sorting Descending-ordered data. The program is much faster than naïve quick sort on sorting Descending-ordered data.

Un-ordered data tested on Randomised pivot quick sort

T(n)	Average	Measurements		
100000	0.23	0.22	0.23	0.23
200000	0.38	0.38	0.39	0.38
300000	0.54	0.54	0.55	0.54
400000	0.73	0.711	0.711	0.781
500000	0.87	0.875	0.88	0.866
600000	1.06	1.08	1.04	1.07
700000	1.21	1.21	1.21	1.22
800000	1.41	1.41	1.36	1.46
900000	1.56	1.53	1.57	1.58
1000000	1.72	1.72	1.71	1.72

OVERSIZE

Fig.9

Analysis:
Fig.9 illustrates Randomised pivot quick sort shows O(n) time complexity on sorting un-ordered data. The program has similar speed to Median-3 quick sort and naïve quick sort on sorting, un-ordered data.