quick

	10	100	1000	10000
랜덤	0.017478	0.095615	1.28617	9.703333
정렬	0.01902	0.052948	0.466763	6.05096
역정렬	0.007711	0.064257	0.70837	6.386639

merge

	10	100	1000	10000
랜덤	0.020562	0.105896	1.464033	17.942118
정렬	0.014908	0.185574	1.136065	16.609683
역정렬	0.024161	0.190201	2.290636	12.758885

insertion

	10	100	1000	10000
랜덤	0.001542	0.501719	663.7787	
정렬	0.000514	0.025189	3.347537	944.1205
역정렬	0.001542	0.698603	1028.2772	

selection

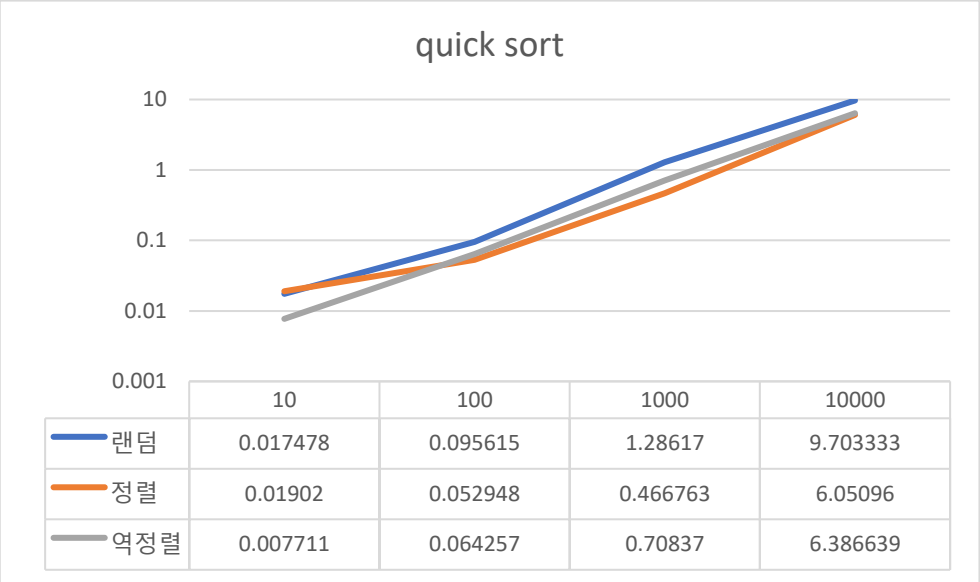
	10	100	1000	10000
랜덤	0.000514	0.015936	1.520579	136.63162
정렬	0.000514	0.014394	1.35968	143.55442
역정렬	0.000514	0.015936	1.354025	168.84807

bubble

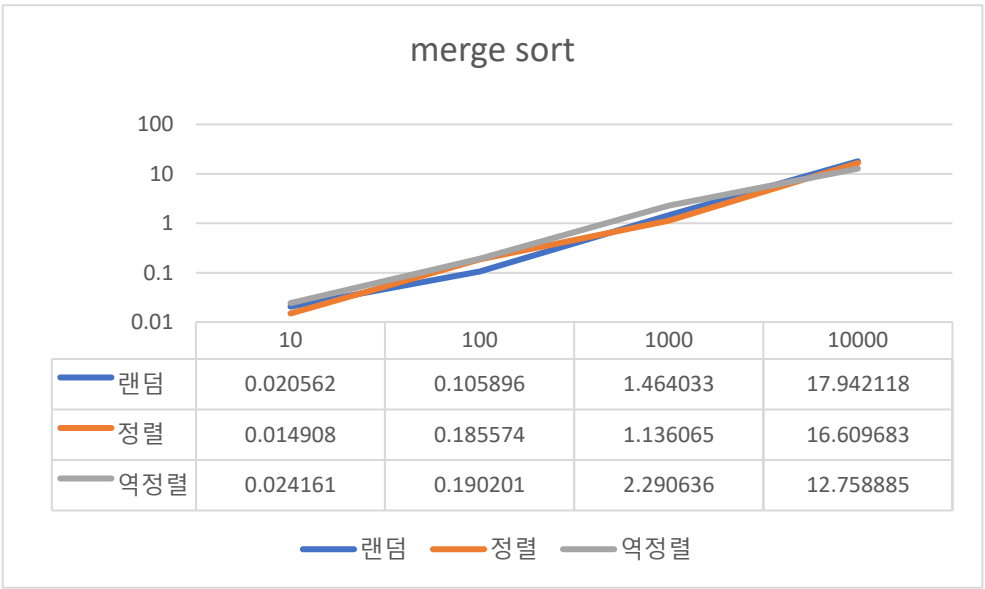
	10	100	1000	10000
랜덤	0.000514	0.026217	2.17446	311.00785
정렬	0.000514	0.015422	1.477399	149.91125
역정렬	0.000514	0.030843	2.564114	292.35634

quick : 실행시간이 N에 비례해서 커짐
정렬 여부는 실행시간과 상관 없는 듯 함
피봇을 중앙값으로 짰는데 최악일 때 실행시간이 어떻게 될 지 모르겠음

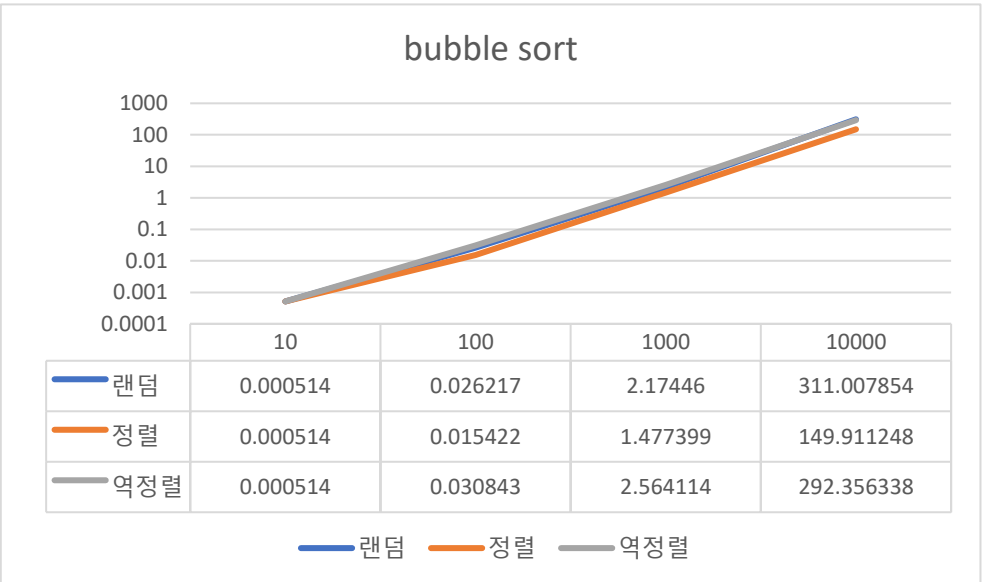
merge : 어떤 데이터를 넣어도 실행시간이 O(nlogn)임
insertion : 정렬된 데이터일 때 최선(10000개부터는 시간측정이 안됨, 잘못 짠듯)
selection : 버블정렬보다는 빨랐지만 최선, 최악, 평균시간 모두 O(n^2)임
bubble : 정렬된 데이터를 넣으면 그마나 시간이 덜 걸리지만 역시 O(n^2)



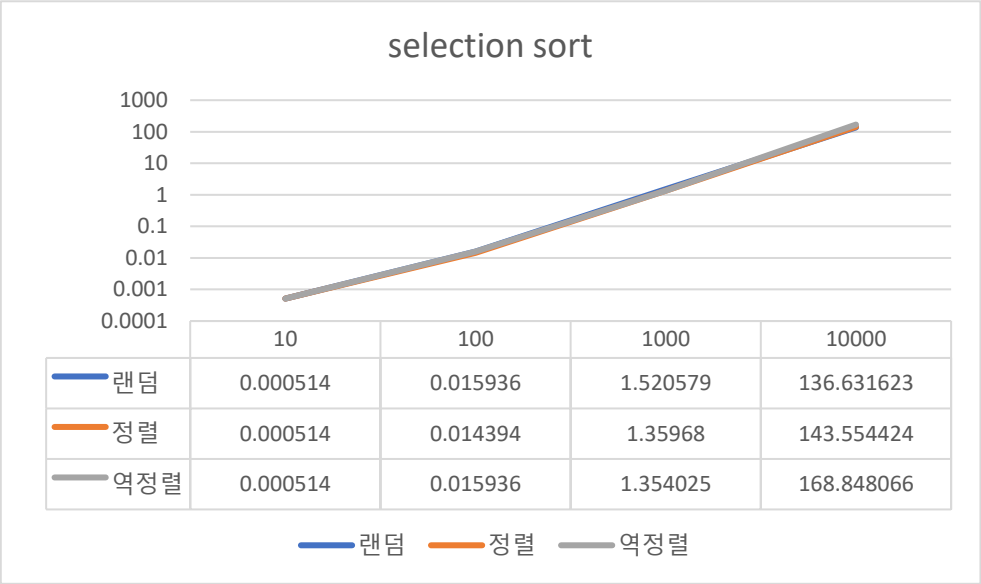
	10	100	1000	10000
랜덤	0.017478	0.095615	1.28617	9.703333
정렬	0.01902	0.052948	0.466763	6.05096
역정렬	0.007711	0.064257	0.70837	6.386639



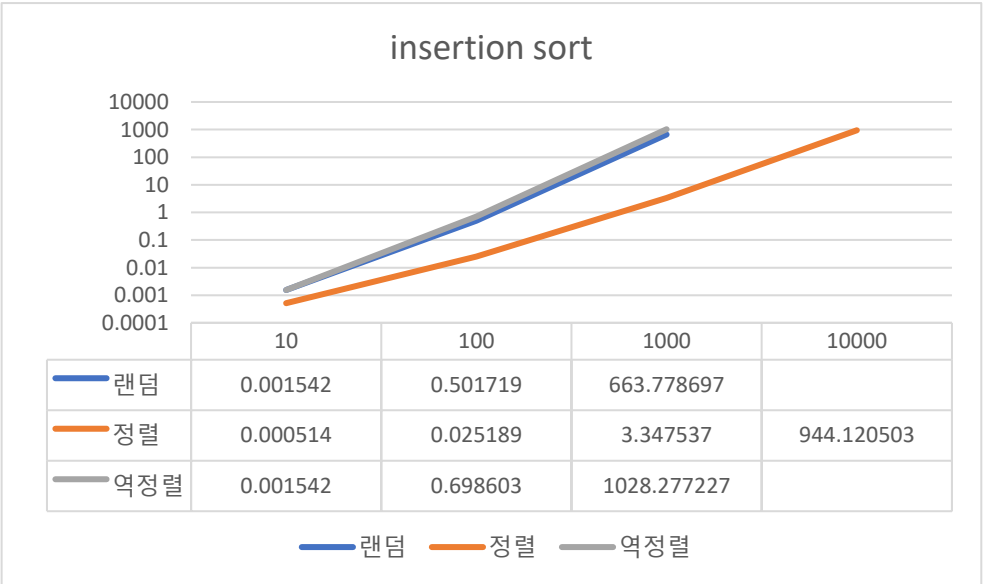
	10	100	1000	10000
랜덤	0.020562	0.105896	1.464033	17.942118
정렬	0.014908	0.185574	1.136065	16.609683
역정렬	0.024161	0.190201	2.290636	12.758885



	10	100	1000	10000
랜덤	0.000514	0.026217	2.17446	311.007854
정렬	0.000514	0.015422	1.477399	149.911248
역정렬	0.000514	0.030843	2.564114	292.356338



	10	100	1000	10000
랜덤	0.000514	0.015936	1.520579	136.631623
정렬	0.000514	0.014394	1.35968	143.554424
역정렬	0.000514	0.015936	1.354025	168.848066



	10	100	1000	10000
랜덤	0.001542	0.501719	663.778697	
정렬	0.000514	0.025189	3.347537	944.120503
역정렬	0.001542	0.698603	1028.277227	