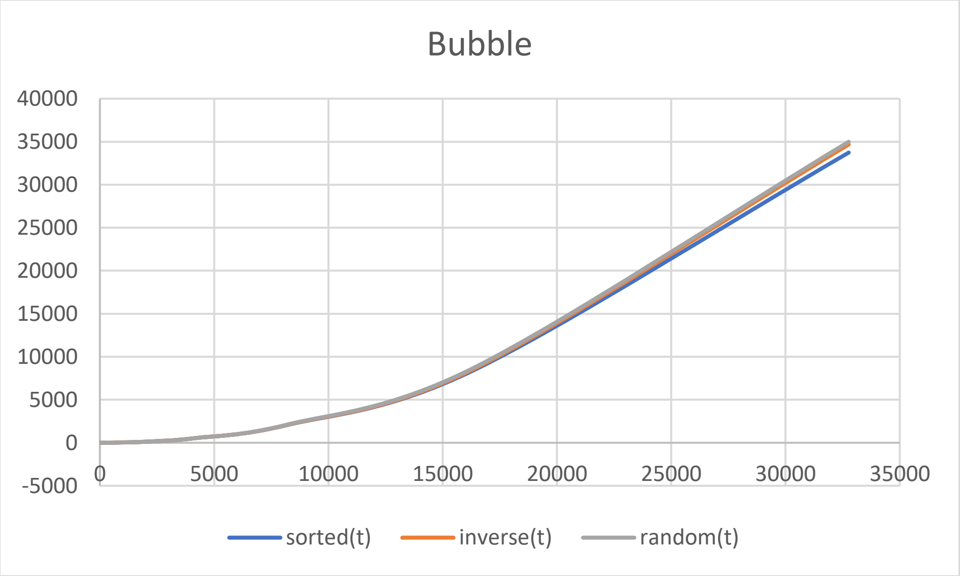
Sorting Laboratory

Emilio Cortina Labra UO257322

Bubble Algorithm

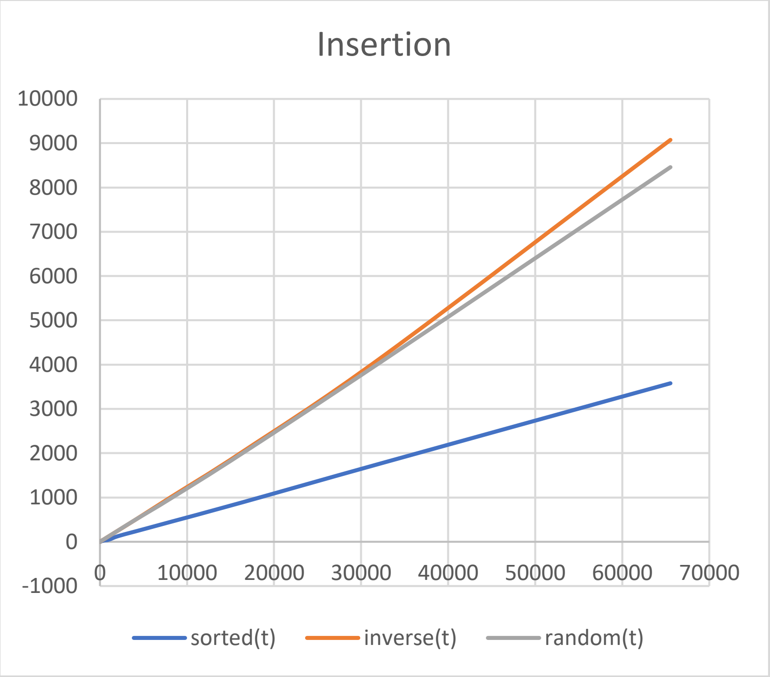
n	sorted(t)	inverse(t)	random(t)
2	0	0	0
4	0	0	0
8	0	0	0
16	0	0	0
32	0	0	0
64	0	0	0
128	0	0	0
256	0	0	0
512	0	0	0
1024	47	47	32
2048	125	125	140
4096	516	516	516
8192	2062	2079	2109
16384	8467	8607	8743
32768	33734	34674	34995



As it can be seen, the times are very similar in every situation, and the curves seems to follow the theoretical complexity of the algorithm, O(n²).

Insertion Algorithm

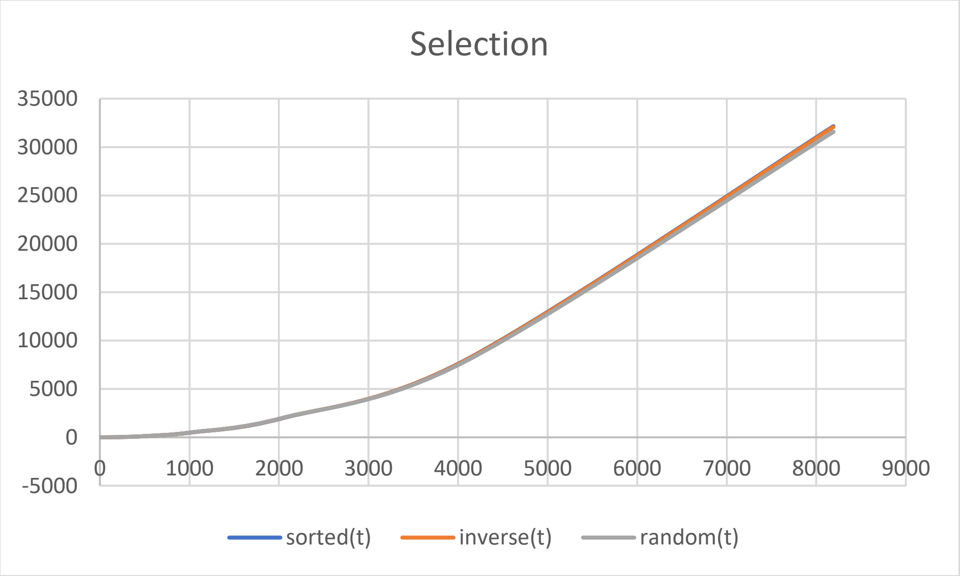
n	sorted(t)	inverse(t)	random(t)
2	0	0	0
4	0	0	0
8	0	0	0
16	0	0	0
32	0	0	0
64	0	16	16
128	15	15	15
256	16	32	31
512	31	62	63
1024	47	125	125
2048	125	250	250
4096	235	500	500
8192	453	1014	985
16384	892	2033	2002
32768	1797	4225	4125
65536	3579	9075	8459



As it can be seen, while the random and inverse scenarios share very similar times, the best case (sorted) is significantly faster, this is because the first are both O(n^2) and the latter is O(n).

Selection Algorithm

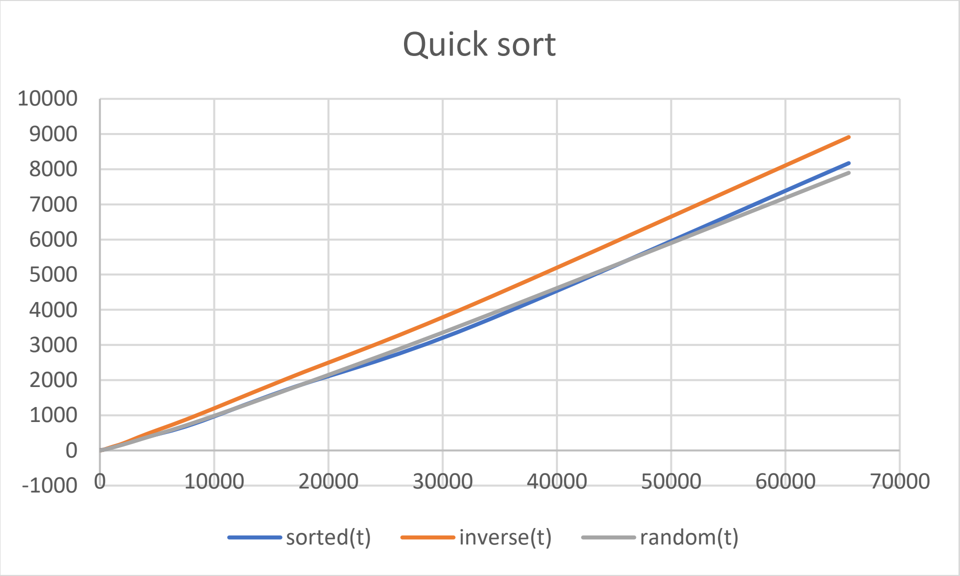
n	sorted(t)	inverse(t)	random(t)
2	0	0	0
4	0	0	0
8	0	0	0
16	1	0	0
32	2	1	1
64	2	2	2
128	8	8	8
256	34	32	32
512	127	130	128
1024	506	506	507
2048	1994	2017	2027
4096	8060	8024	7916
8192	32153	32064	31569



In this method, as it can be seen there are no differences between times in any scenario, each case has the same complexity as the others, O(n²).

Quick sort with central element Algorithm

n	sorted(t)	inverse(t)	random(t)
2	1	0	0
4	2	0	0
8	2	0	0
16	4	1	0
32	8	3	1
64	6	7	3
128	14	7	6
256	18	16	16
512	36	41	33
1024	87	96	77
2048	183	202	169
4096	382	467	371
8192	755	962	785
16384	1735	2042	1723
32768	3554	4169	3699
65536	8171	8912	7896



Finally, in the Quick Sort algorithm with central element all three cases share the same complexity O(nlogn).