SPRAWOZDANIE

Algorytmy sortowania

Czas działania poszczególnych algorytmów w zależności od typu danych wejściowych i wielkości instancji

Wartości losowe					
Wielkość instancji	**Selection Sort**	**Insertion Sort**	**Quick Sort (OW)***	**Quick Sort (LW)***	**Heap Sort ****
100	0.0	0.0	0.00	0.00	-
1000	0.015	0.002	0.00	0.002	-
3000	0.016	0.015	0.16	0.015	-
10000	0.173	0.112	0.145	0.132	-
25000	0.784	0.552	0.813	0.607	-
40000	2.457	1.087	2.115	1.683	-
60000	5.123	3.404	4.497	4.214	-
100000	16.522	13.781	12.660	8.982	-
300000	133.362	68.577	135.441	92.598	-
1000000	1695.603	1307.281	2642.512	1620.413	-
	Czas wykonania algorytmu (s)				

Wartości rosnące					
Wielkość instancji	**Selection Sort**	**Insertion Sort**	**Quick Sort (OW)***	**Quick Sort (LW)***	**Heap Sort ****
100	0.0	0.0	0.00	0.00	-
1000	0.003	0.0	0.00	0.00	-
3000	0.028	0.0	0.015	0.00	-
10000	0.165	0.0	0.274	0.00	-
25000	0.799	0.0	1.648	0.00	-
40000	1.909	0.0	4.269	0.00	-
60000	5.264	0.0	9.454	0.001	-
100000	12.245	0.0	26.575	0.001	-
300000	123.684	0.01	240.217	0.002	-
1000000	1384.134	0.03	2091.388	0.005	-

	Czas wykonania algorytmu (s)				

Wartości malejące

Wielkość instancji	Selection Sort	Insertion Sort	Quick Sort (OW)*	Quick Sort (LW)*	Heap Sort **
100	0.00	0.00	0.00	0.00	-
1000	0.001	0.003	0.003	0.005	-
3000	0.011	0.031	0.019	0.017	-
10000	0.123	0.136	0.144	0.205	-
25000	0.943	0.922	0.975	1.730	-
40000	2.178	2.293	2.892	5.286	-
60000	5.16	5.884	5.914	13.281	-
100000	12.721	14.675	14.966	33.131	-
300000	125.543	123.731	145.605	249.401	-
1000000	1290.052	1281.878	1591.272	3171.557	-

	Czas wykonania algorytmu (s)				

Rozkład V-kształtny ***

Wielkość instancji	Selection Sort	Insertion Sort	Quick Sort (OW)*	Quick Sort (LW)*	Heap Sort **
100	-	-	-	-	-
1000	-	-	-	-	-
3000	-	-	-	-	-
10000	-	-	-	-	-
25000	-	-	-	-	-
40000	-	-	-	-	-
60000	-	-	-	-	-
100000	-	-	-	-	-
300000	-	-	-	-	-
1000000	-	-	-	-	-

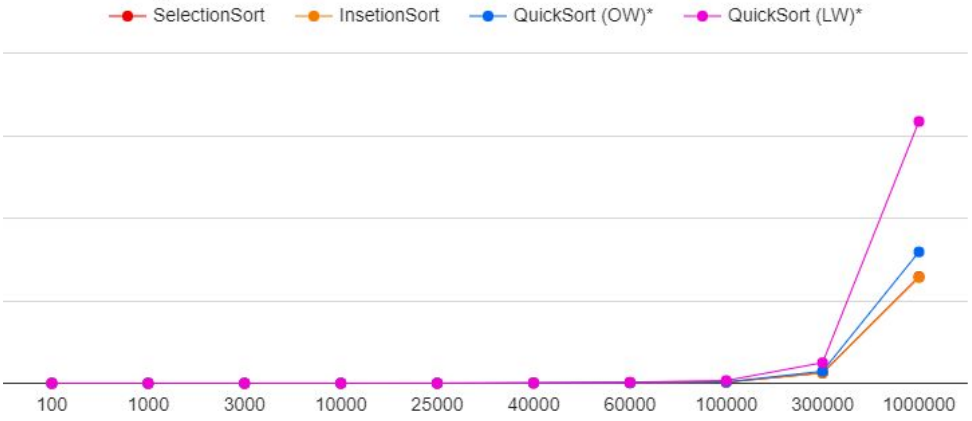
	Czas wykonania algorytmu (s)

* Quick Sort (OW) - Quick Sort, gdzie ostatnia wartość w tablicy jest elementem rozdzielającym
Quick Sort (LW) - Quick Sort, gdzie losowa wartość w tablicy jest elementem rozdzielającym

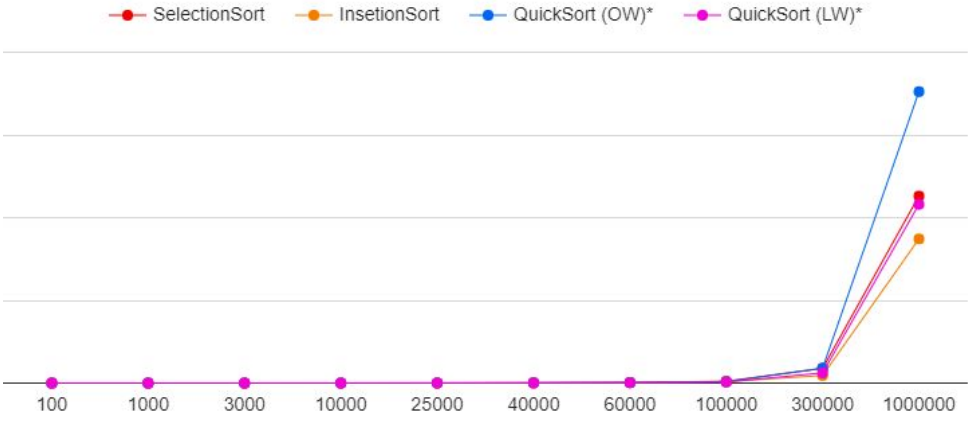
** W związku niepoprawnego działania Heap Sorta został on wyłączony, a co za tym idzie, jest brak wyników czasowych, które byłyby błędne. Kod z tym algorytmem jest dostępny w repozytorium.

*** Podobna sytuacja ma miejsce z wypełnianiem instancji w rozkładzie V-kształtnym. Cały kod jest dostępny w repozytorium.

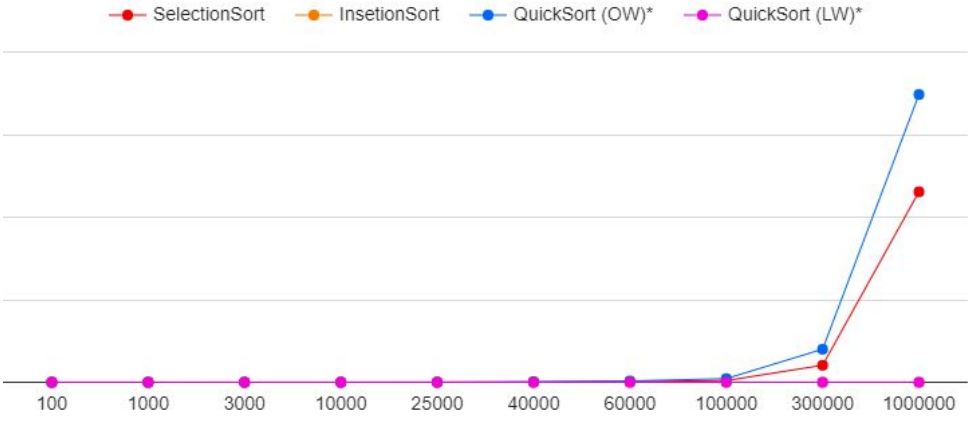
Wartości malejące



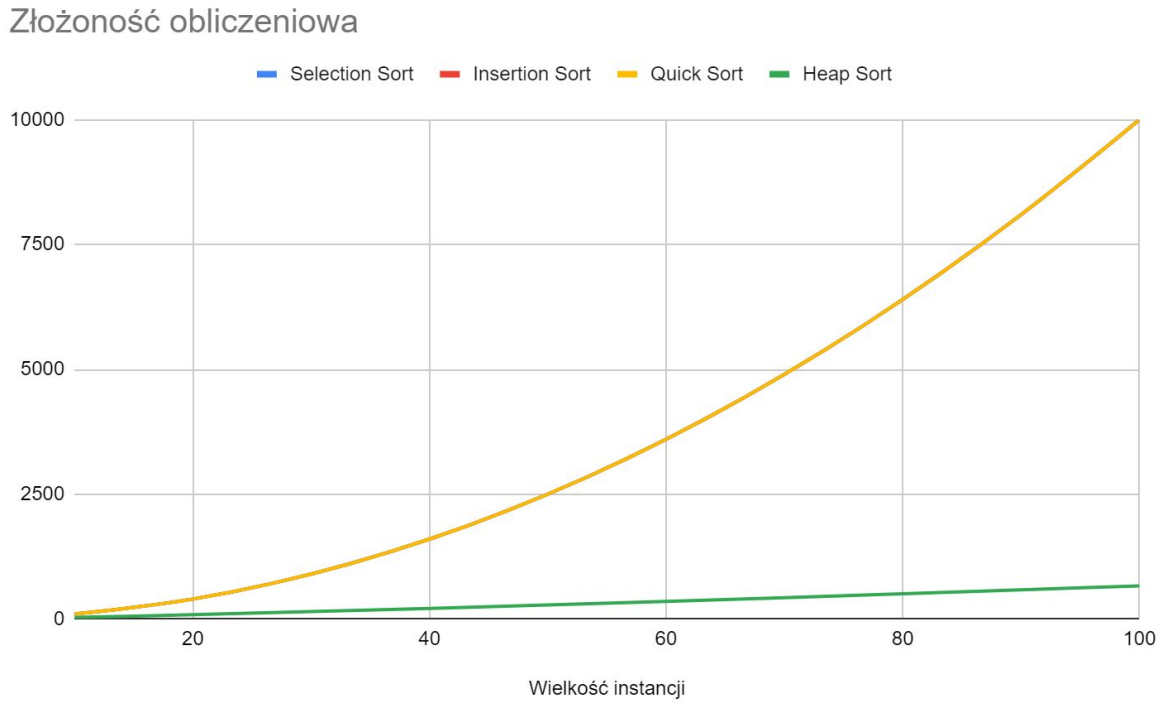
Wartości losowe



Wartości rosnące



Złożonosc obliczeniowa w najgorszym przypadku dla danego algorytmu				
Wielkość instancji	**Selection Sort**	**Insertion Sort**	**Quick Sort**	**Heap Sort**
10	100	100	100	33
20	400	400	400	86
30	900	900	900	147
40	1600	1600	1600	213
50	2500	2500	2500	282
60	3600	3600	3600	354
70	4900	4900	4900	429
80	6400	6400	6400	506
90	8100	8100	8100	584
100	10000	10000	10000	664
	Liczba wykonanych operacji			



Najgorszy przypadek dla Heap Sort to O(n log n)
Najgorszy przypadek dla Selection Sort, Insertion Sort oraz Quick Sort to $O(n^2)$, dlatego linie na wykresie dla tych algorytmów się pokrywają

W przypadku instancji z wartościami rosnącymi niektóre algorytmy tylko sprawdzały poprawność ułożenia wartości, natomiast algorytmy na przykład takie jak Selection Sort przechodziły cały proces sortowania.

Instancja milion elementowa w porównaniu do około trzy razy mniejszej instancji trzystu tysięczno elementowej jest sortowana kilkunastokrotnie dłużej, nie jest więc to równomierny wzrost i stosunek wielkości instancji do czasu sortowania.

Algorytm Heap Sort jest najszybszym algorytmem. Jego złożoność obliczeniowa dla przypadku najlepszego jak i najgorszego wynosi O(n log n).
Dla Quick Sorta oraz Insertion Sorta złożoność obliczeniowa dla najlepszego przypadku wynosi O(n) a dla najgorszego $O(n^2)$.
Najgorzej wypada algorytm Selection Sort, ponieważ jego złożoność obliczeniowa w obu przypadkach to $O(n^2)$