Sprawozdanie

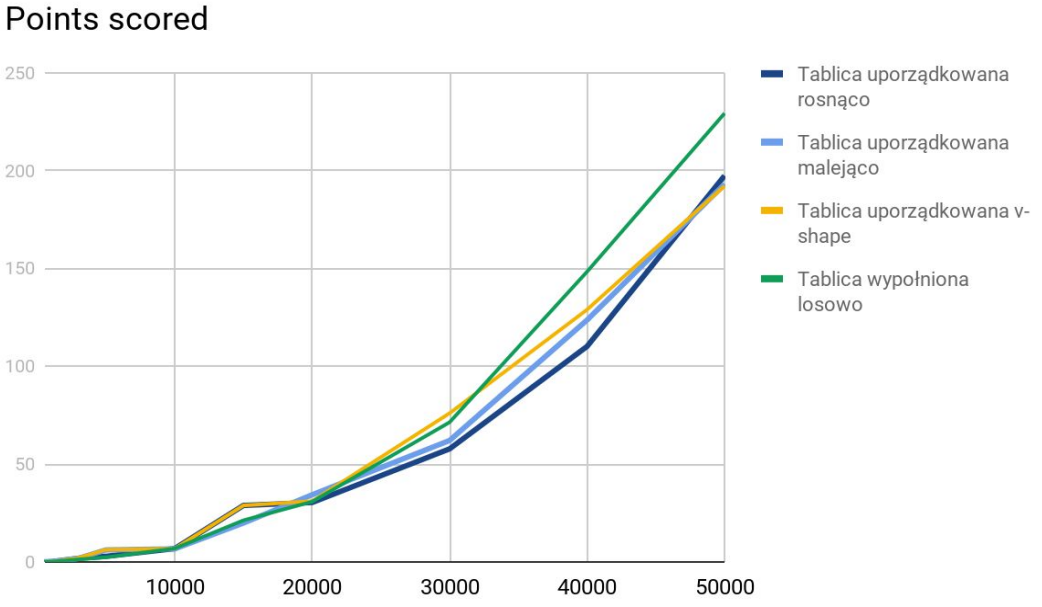
Jakub Królczyk nr.146417
Kamil Lewiński nr. 146437

Język programowania - pyhton

Czas liczony jest w sekundach.
Na górze każdej z tabel napisana jest ilość elementów poddanej algorytmom tablicy.

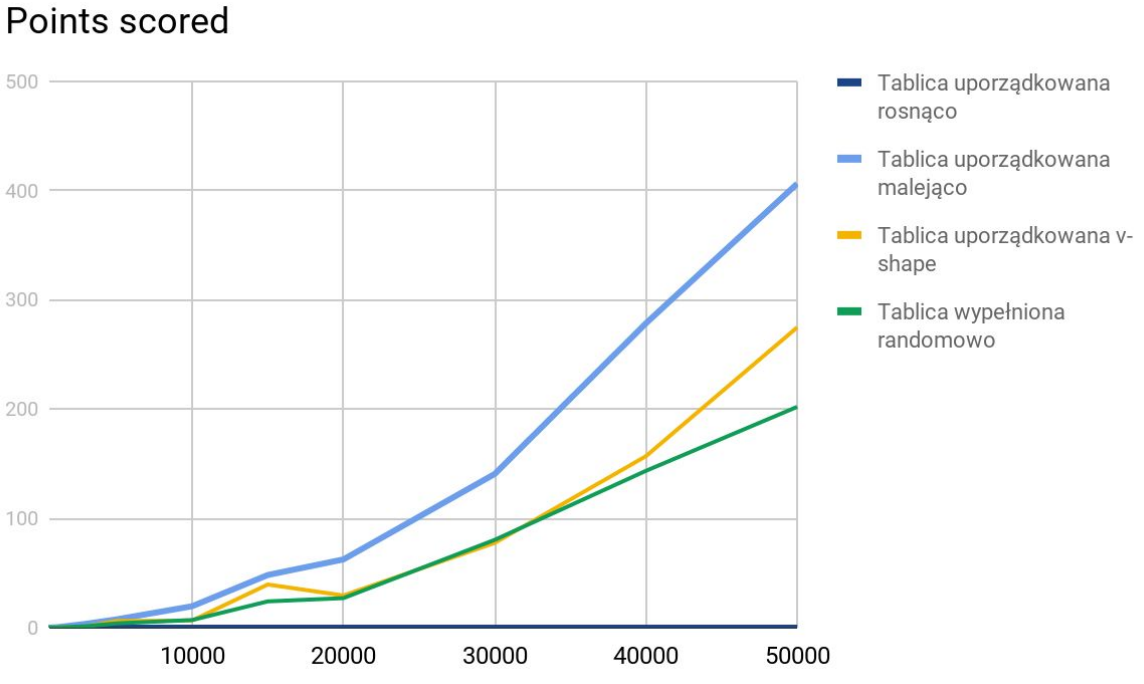
Selection Sort:

	500	1k	3k	5k	10k	15k	20k	30k	40k	50k
Tablica uporządkowana rosnąco	0,0229	0,0919	1,8711	2,8715	6,8209	28,884	30,393	57,815	110,20	197,44
Tablica uporządkowana malejąco	0,0239	0,1279	1,8220	6,1535	6,6852	19,895	34,327	62,110	123,69	193,11
Tablica uporządkowana v-shape	0,0249	0,0999	2,0299	6,2675	6,7582	28,886	31,087	76,048	128,99	192,19
Tablica wypełniona randomowo	0,0229	0,0989	1,3109	2,3639	6,8840	21,276	30,814	71,412	148,40	229,34



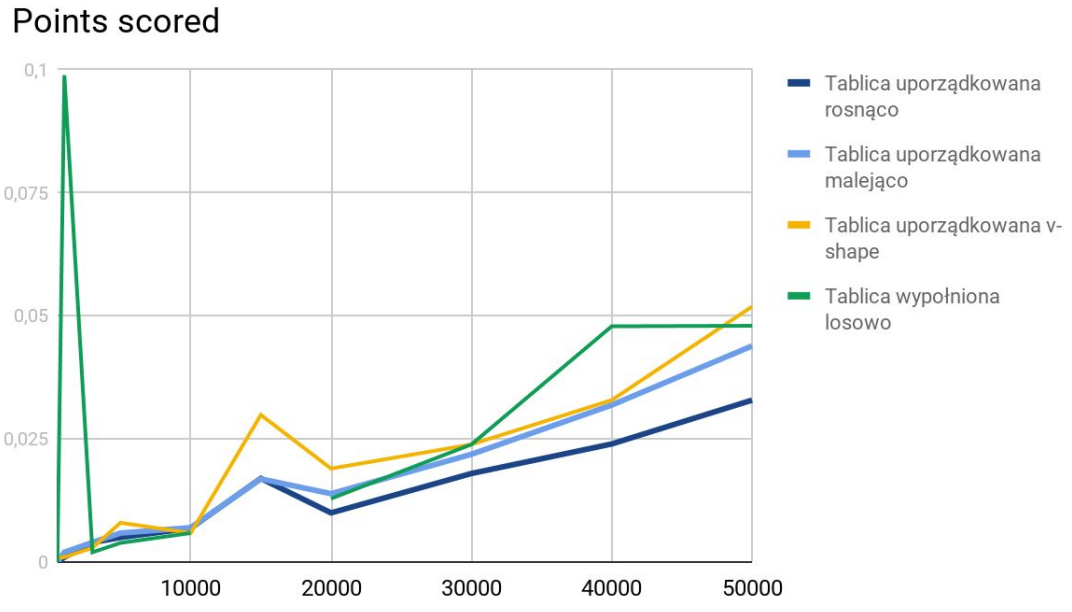
Insertion Sort:

	500	1k	3k	5k	10k	15k	20k	30k	40k	50k
Tablica uporządkowana rosnąco	0,0	0,0	0,0029	0,0020	0,0059	0,0099	0,0060	0,0090	0,0179	0,0190
Tablica uporządkowana malejąco	0,0540	0,2429	3,5249	7,5430	19,582	48,184	62,433	140,65	278,54	406,40
Tablica uporządkowana v-shape	0,0360	0,1089	1,1829	6,0101	6,5223	39,486	29,418	77,545	156,94	274,86
Tablica wypełniona randomowo	0,0270	0,1029	1,3679	3,8949	6,9700	24,028	27,015	,32480	143,53	202,11



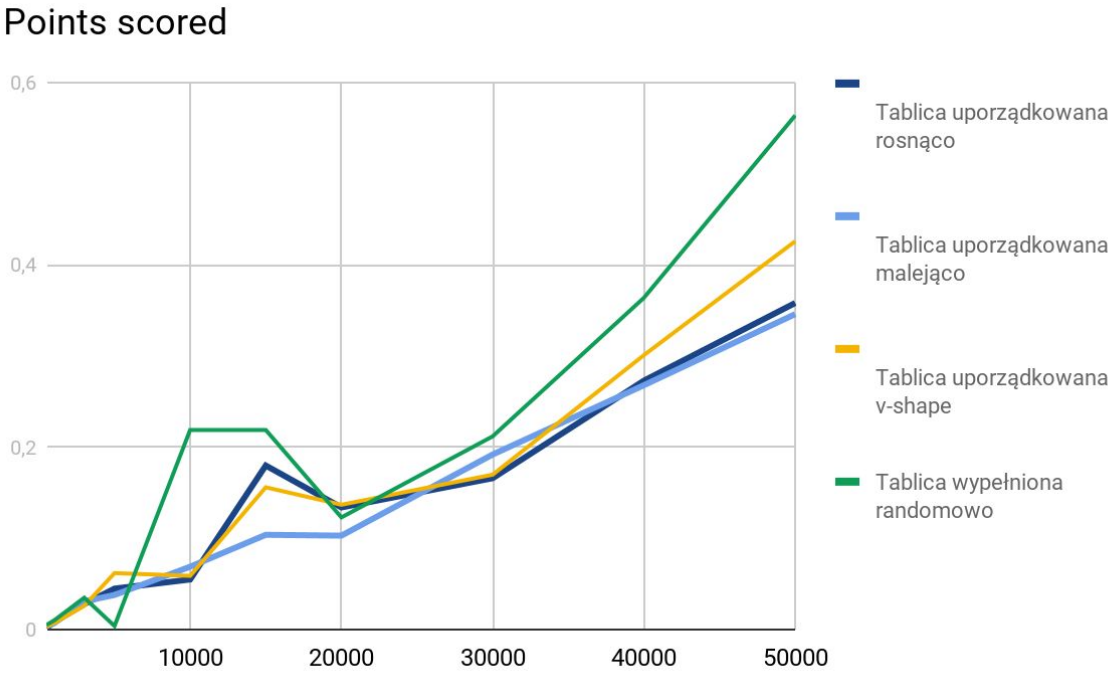
Heap Sort:

	500	1k	3k	5k	10k	15k	20k	30k	40k	50k
Tablica uporządkowana rosnąco	0,0	0,0010	0,0040	0,0050	0,0069	0,0170	0,0100	0,0180	0,0240	0,0329
Tablica uporządkowana malejąco	0,0	0,0020	0,0040	0,0059	0,007	0,0169	0,0139	0,0219	0,0319	0,0439
Tablica uporządkowana v-shape	0,0009	0,0010	0,0029	0,0080	0,0059	0,0299	0,0190	0,0239	0,0329	0,0519
Tablica wypełniona randomowo	0,0	0,0989	0,0020	0,0039	0,0059	0.0189	0,0129	0,0239	0,0479	0,0480



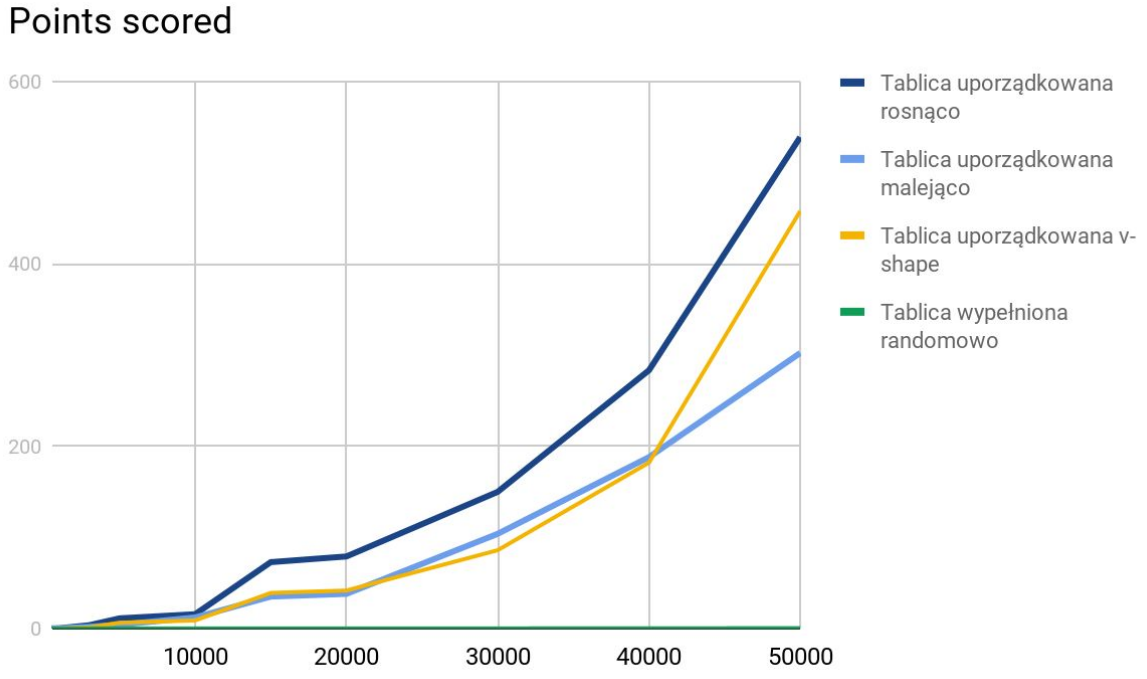
Quick Sort random partition

	500	1k	3k	5k	10k	15k	20k	30k	40k	50k
Tablica uporządkowana rosnąco	0,0019	0,0080	0,0280	0,0449	0,0549	0,1799	0,1339	0.1660	0,2730	0,3582
Tablica uporządkowana malejąco	0,0039	0,0069	0,0309	0,0379	0,0689	0,1039	0,1030	0.1920	0,2680	0,3460
Tablica uporządkowana v-shape	0,0030	0,0070	0,0260	0,0619	0,0589	0,1560	0,1369	0.1699	0,3010	0,4259
Tablica wypełniona randomowo	0,0050	0,0099	0,0349	0,0039	0,0619	0,2189	0,1230	0.2119	0,3639	0,5639

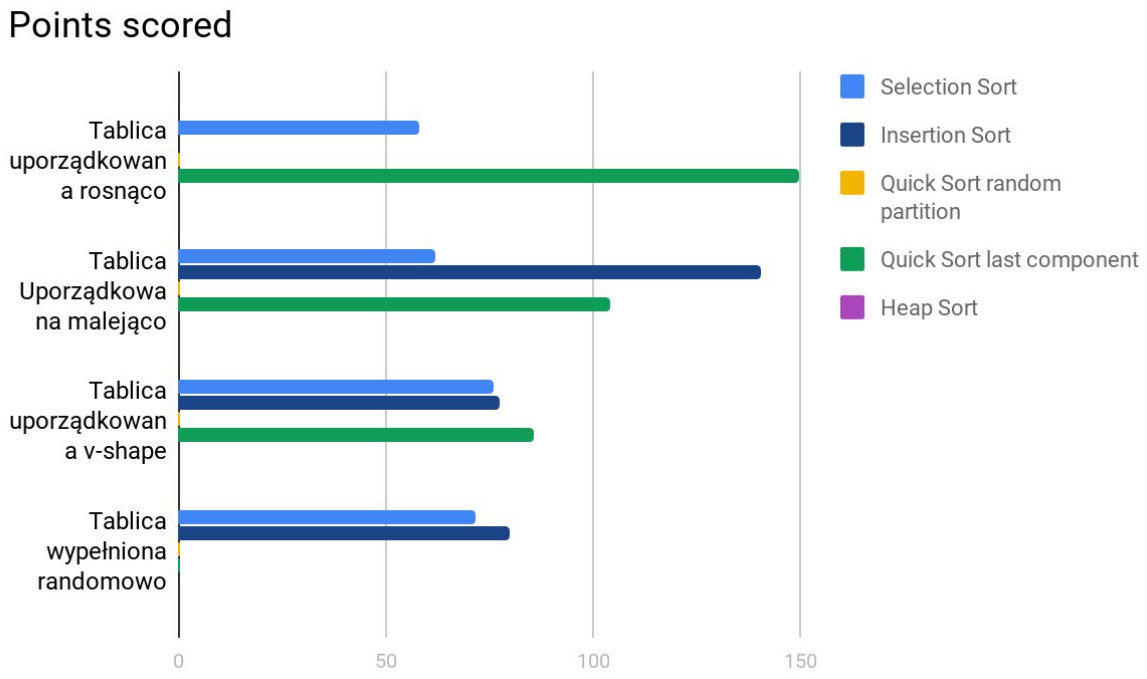


Quick Sort Last component

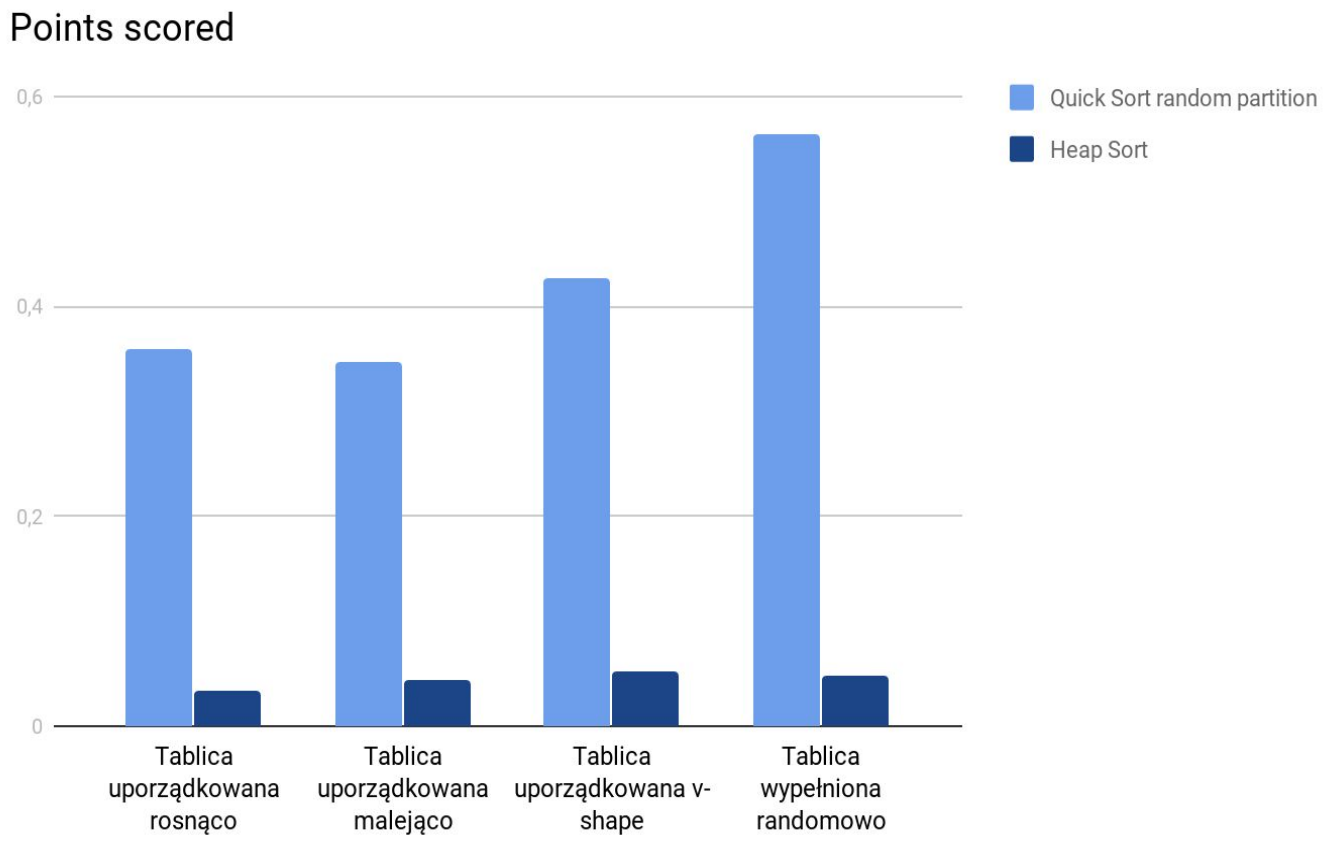
	500	1k	3k	5k	10k	15k	20k	30k	40k	50k
Tablica uporządkowana rosnąco	0,0569	0,2409	3,6339	11,163	15,696	72,835	79,059	149,92	283,28	538,99
Tablica uporządkowana malejąco	0,0399	0,1560	1,8399	4,1285	12,111	35,049	38,022	103,88	187,96	302,23
Tablica uporządkowana v-shape	0,0009	0,1179	1,2550	6,3699	8,8739	39,069	41,710	85,906	181,92	457,84
Tablica wypełniona randomowo	0,0229	0,0040	0,0219	0,0699	0,0360	0,1560	0,0769	0,1469	0,2359	0,3659



Porównanie algorytmów dla 30000 elementowej tablicy:



Porównanie Heap Sort i Quick Sort random partition dla tablicy 50000 elementowej:



Wnioski:

Z badanych algorytmów najbardziej optymalnym zdaje się być korzystanie z algorytmu Heap Sort ponieważ działa on najszybciej w największej ilości przypadków. Jego złożoność czasowa to O(n log(n)).

Quick Sort random partition radzi sobie znacznie lepiej z sortowaniem uporządkowanych tablic niż Quick Sort last component, który musi wykonać znacznie więcej operacji aby posortować uporządkowaną tablicę.

Dla Insertion Sort najlepszym przypadkiem jest tablica zupełnie uporządkowana i radzi on sobie z nią najszybciej ze wszystkich algorytmów. Najgorszym przypadkiem dla tego algorytmu jest tablica uporządkowana malejąco ponieważ musi on wtedy wykonać największą ilość operacji $O(n^2)$