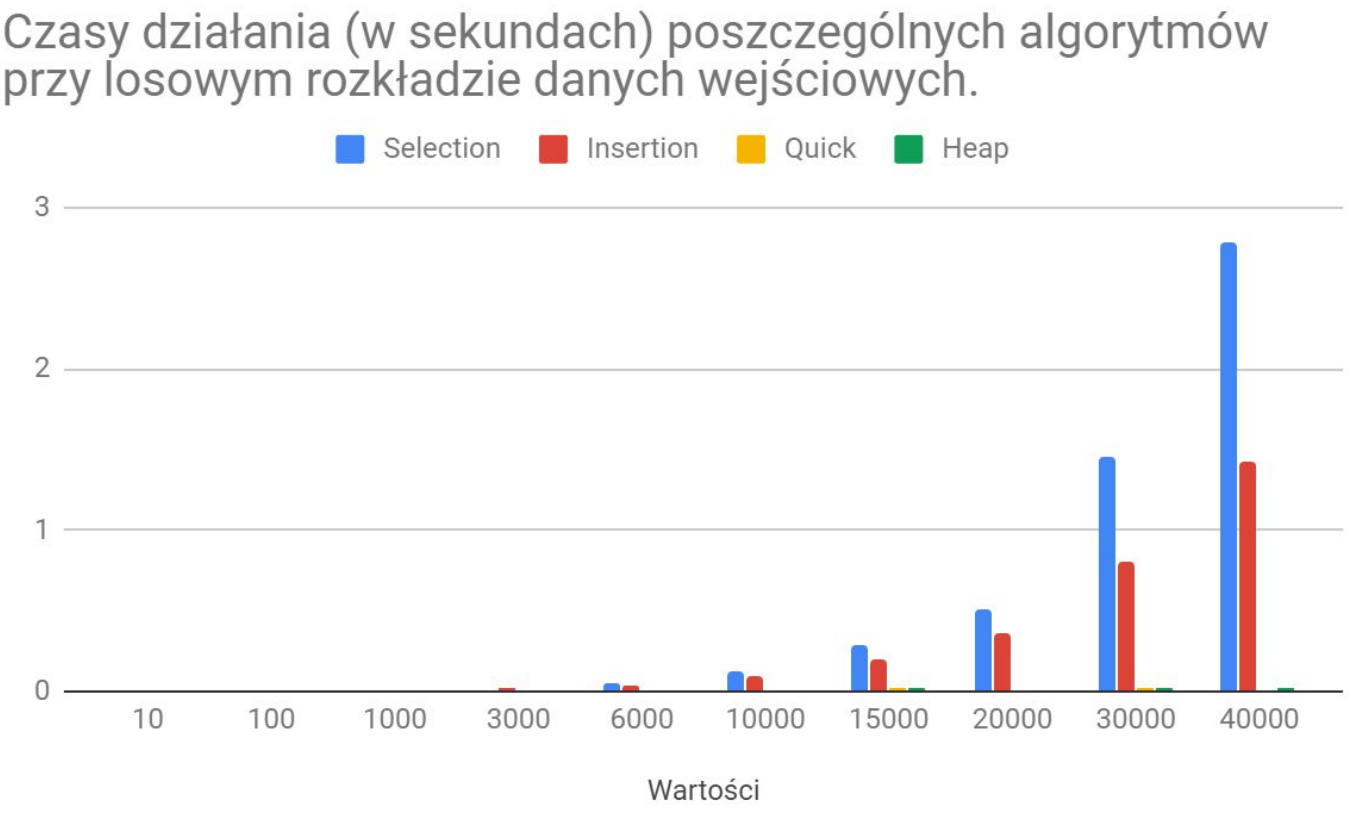
Sprawozdanie:

Michał Szczepanowicz
Jan Kociński

Czasy działania (w sekundach) poszczególnych algorytmów przy **losowym** rozkładzie danych wejściowych.

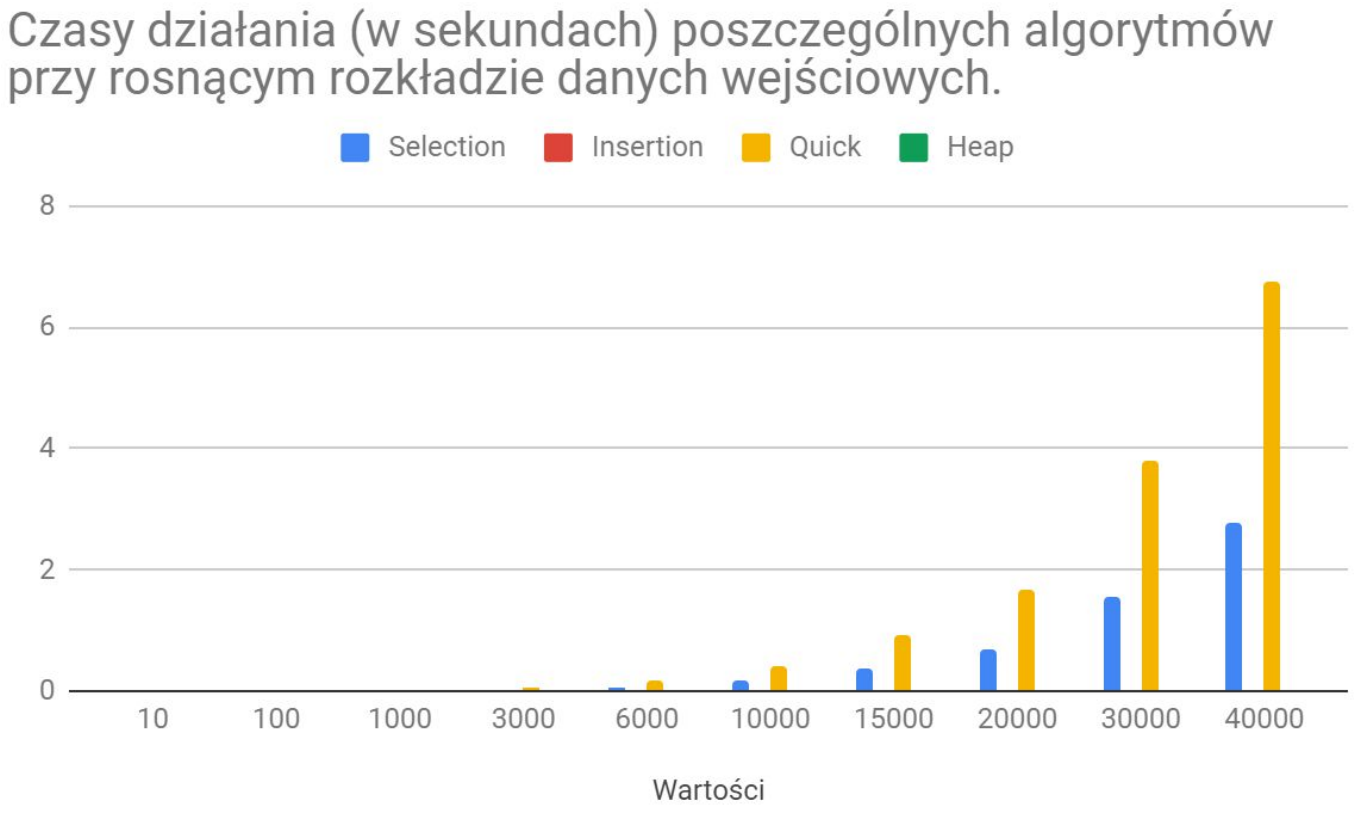
Wartości	10	100	1000	3000	6000	10000	15000	20000	30000	40000
Selection	0	0	0	0	0,047	0,125	0,281	0,516	1,453	2,781
Insertion	0	0	0	0,016	0,031	0,094	0,203	0,359	0,797	1,422
Quick	0	0	0	0	0	0	0,015	0	0,016	0
Heap	0	0	0	0	0	0	0,016	0	0,015	0,016



.

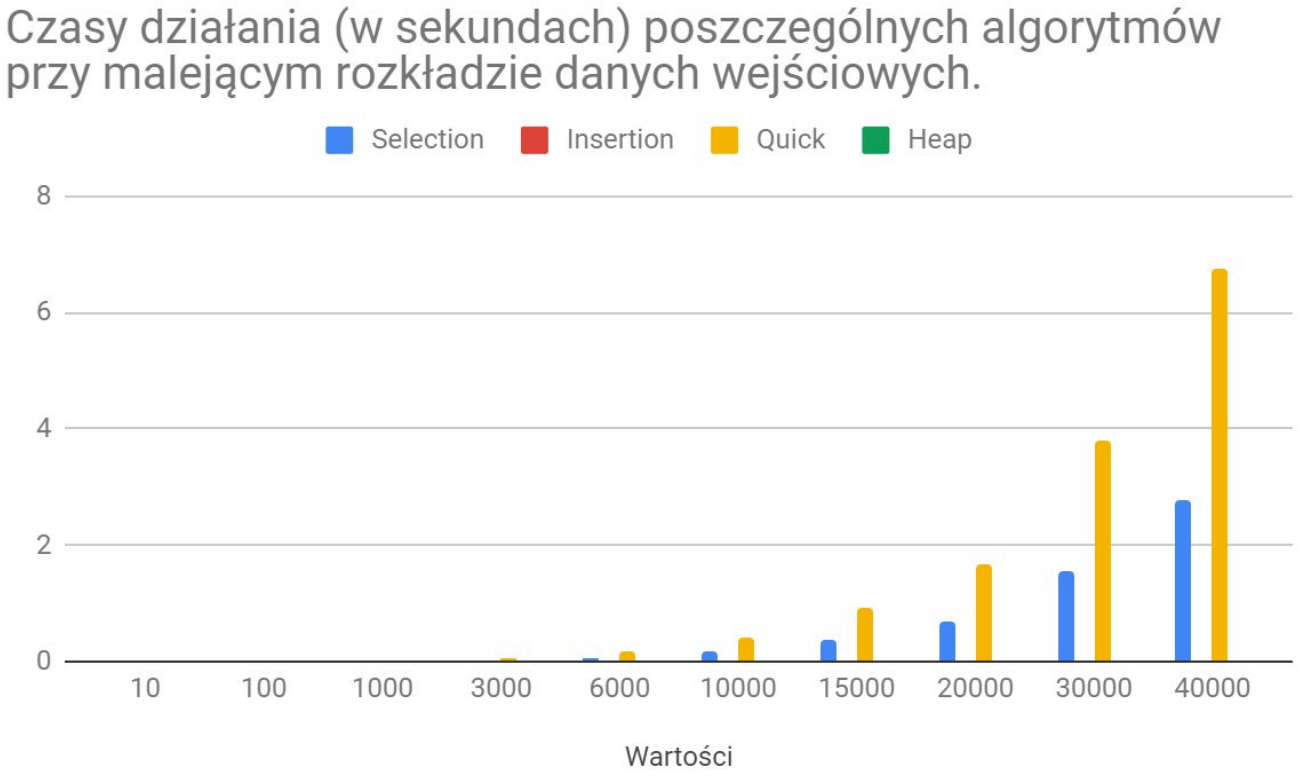
Czasy działania (w sekundach) poszczególnych algorytmów przy **rosnącym** rozkładzie danych wejściowych.

Wartości	10	100	1000	3000	6000	10000	15000	20000	30000	40000
Selection	0	0	0,015	0,016	0,063	0,171	0,391	0,703	1,562	2,781
Insertion	0	0	0	0	0	0	0	0	0	0
Quick	0	0	0	0,047	0,157	0,406	0,938	1,672	3,799	6,737
Heap	0	0	0	0	0	0	0	0,016	0	0



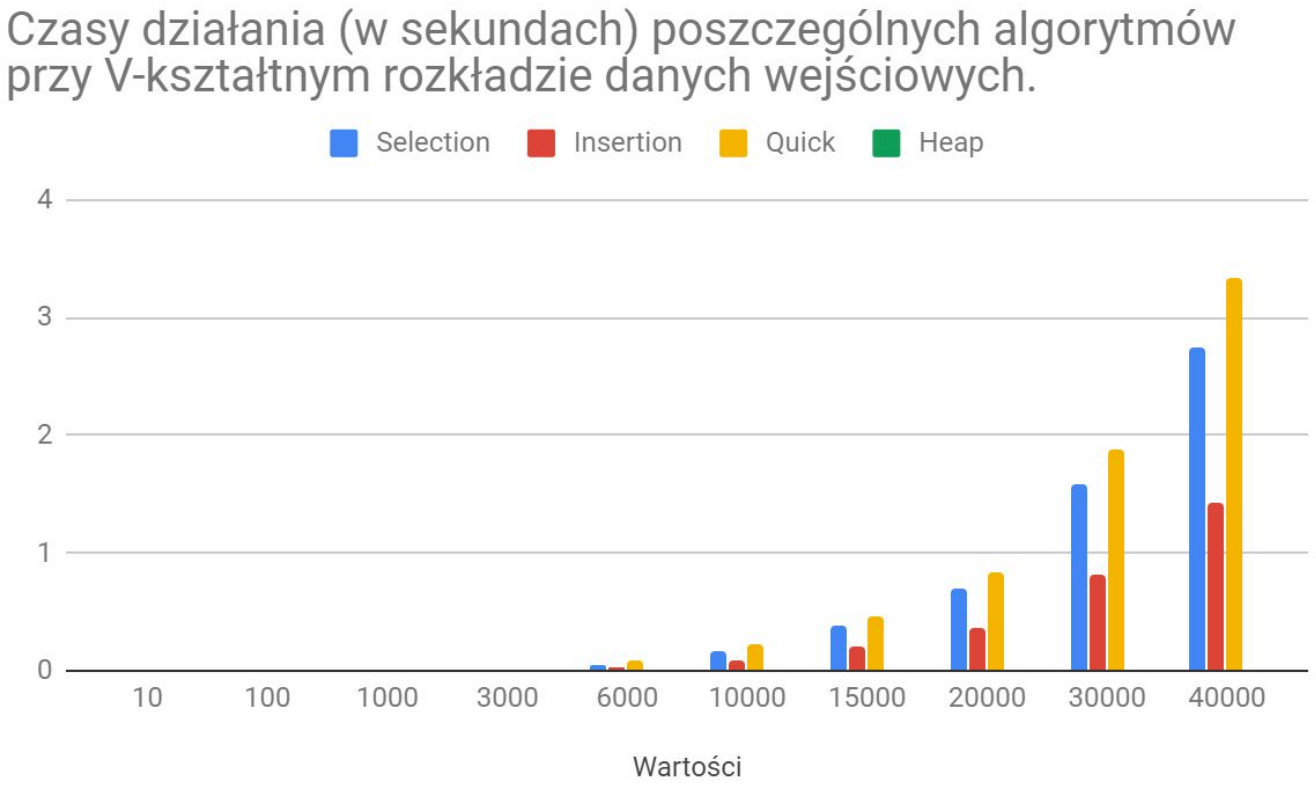
Czasy działania (w sekundach) poszczególnych algorytmów przy **malejącym** rozkładzie danych wejściowych.

Wartości	10	100	1000	3000	6000	10000	15000	20000	30000	40000
Selection	0	0	0	0,015	0,063	0,172	0,375	0,671	1,565	2,690
Insertion	0	0	0,016	0,015	0,063	0,172	0,407	0,718	1,594	2,875
Quick	0	0	0	0,031	0,110	0,297	0,656	1,172	2,641	4,702
Heap	0	0	0	0	0,016	0	0	0	0,016	0,016



Czasy działania (w sekundach) poszczególnych algorytmów przy **V-kształtnym** rozkładzie danych wejściowych.

Wartości	10	100	1000	3000	6000	10000	15000	20000	30000	40000
Selection	0	0	0	0,015	0,048	0,172	0,375	0,703	1,578	2,750
Insertion	0	0	0	0	0,031	0,094	0,203	0,359	0,809	1,422
Quick	0	0	0	0,016	0,078	0,218	0,469	0,828	1,875	3,328
Heap	0	0	0	0	0,015	0	0	0	0,016	0,015



Rodzaj algorytmu	Sposób najlepszy	Sposób najgorszy
Selection	losowy	rosnący
Insertion	rosnący	malejący
Quick	losowy	rosnący

Heap	rosnący	malejący

Złożoność obliczeniowa :

Algorytm	przypadek najlepszy	przypadek najgorszy
Selection	O(n^2)	O(n^2)
Insertion	O(n^2)	O(n^2)
Quick	O(n*log n)	O(n^2)
Heap	O(n)	O(n*log n)

Wnioski:

Najlepszym algorytmem okazał się być Heap sort, który dla wszystkich rodzajów danych działał bez wątpienia najszybciej. Algorytm Quicksort wypadł zaskakująco słabo przy rosnących, v-kształtnych i malejących wartościach. Natomiast okazał się być najlepszy jeśli chodzi o losowe wartości. Z przedstawionych przez nas danych wynika, że niezależnie od rodzaju podanych wartości najlepszym wyborem jest heapsort, który nawet przy dużych liczbach sprawdza się dobrze. Natomiast najgorszym wyborem jest selectionsort ,który jest mało efektywny w przypadku liczb większych wartości liczbowych. Poniższa tabela przedstawia sposób najlepszy i najgorszy dla poszczególnych algorytmów:

Rodzaj algorytmu	Sposób najlepszy	Sposób najgorszy
Selection	losowy	rosnący
Insertion	rosnący	malejący
Quick	losowy	rosnący
Heap	rosnący	malejący