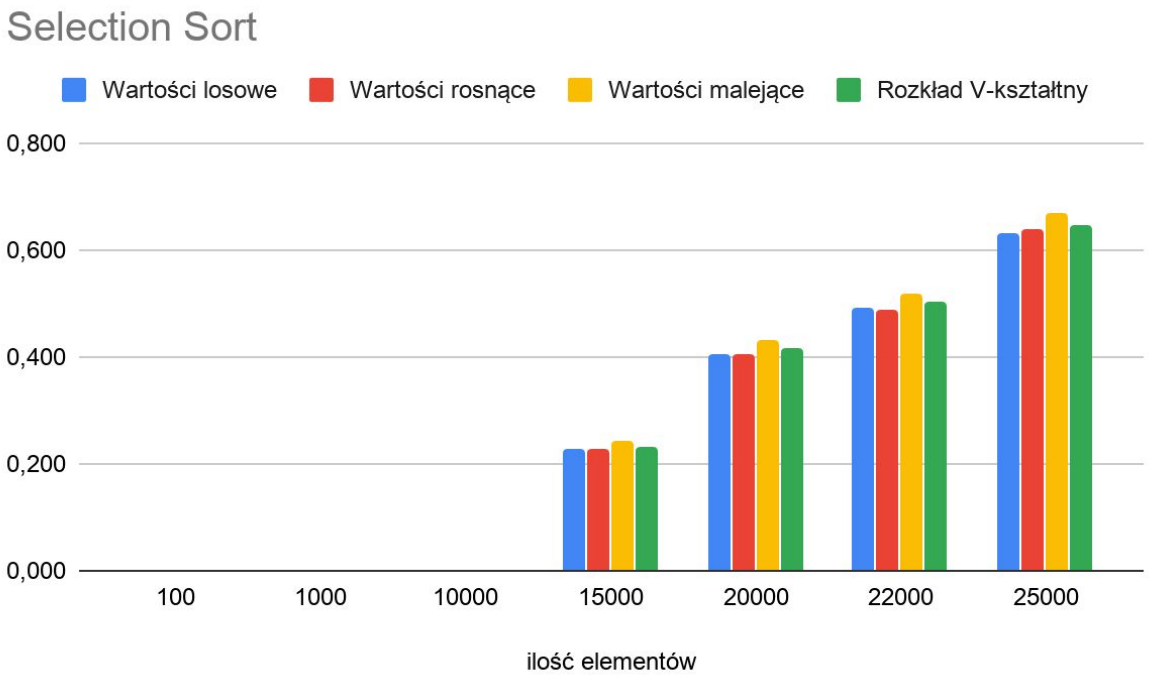
Kierunek: Informatyka	Semestr: 2	Grupa dziekańska: 3a	Data: 28.03.2020
Wykonali: Radosław Rumplewicz Grzegorz Stachowiak	Numer albumu: 131367 146536	Algorytmy i Struktury Danych	Laboratorium 1

1. Algorytm sortowania Selection Sort:
   a. Tabela wyników:

Selection Sort				
ilość elementów	Wartości losowe	Wartości rosnące	Wartości malejące	Rozkład V-kształtny
100	0,000	0,000	0,000	0,000
1000	0.001	0,001	0,001	0,002
10000	0,001	0,001	0,001	0,002
15000	0,228	0,228	0,242	0,233
20000	0,405	0,405	0,430	0,415
22000	0,490	0,488	0,517	0,502
25000	0,632	0,638	0,668	0,647

   b. Wykres wyników:

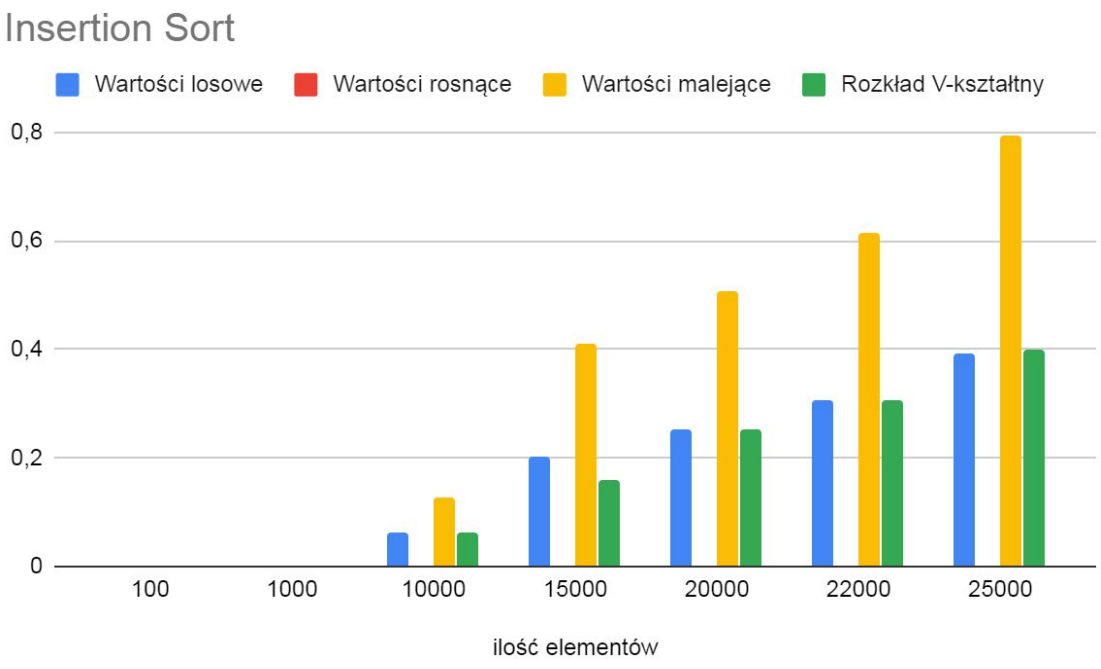


   c. Analiza zachowania algorytmu:
   Czas wykonywania się algorytmu jest zależny głównie od ilości danych. Minimalna różnica występuje przy wartościach malejących, co może wskazywać na najgorszy przypadek.

2. Algorytm sortowania Insertion Sort:
   a. Tabela wyników:

Insertion Sort				
ilość elementów	Wartości losowe	Wartości rosnące	Wartości malejące	Rozkład V-kształtny
100	0,000	0,000	0,000	0,000
1000	0,000	0,000	0,002	0,000
10000	0,063	0,000	0,128	0,063
15000	0,203	0,000	0,409	0,160
20000	0,251	0,000	0,508	0,251
22000	0,308	0,000	0,616	0,307
25000	0,391	0,000	0,795	0,398

   b. Wykres wyników:

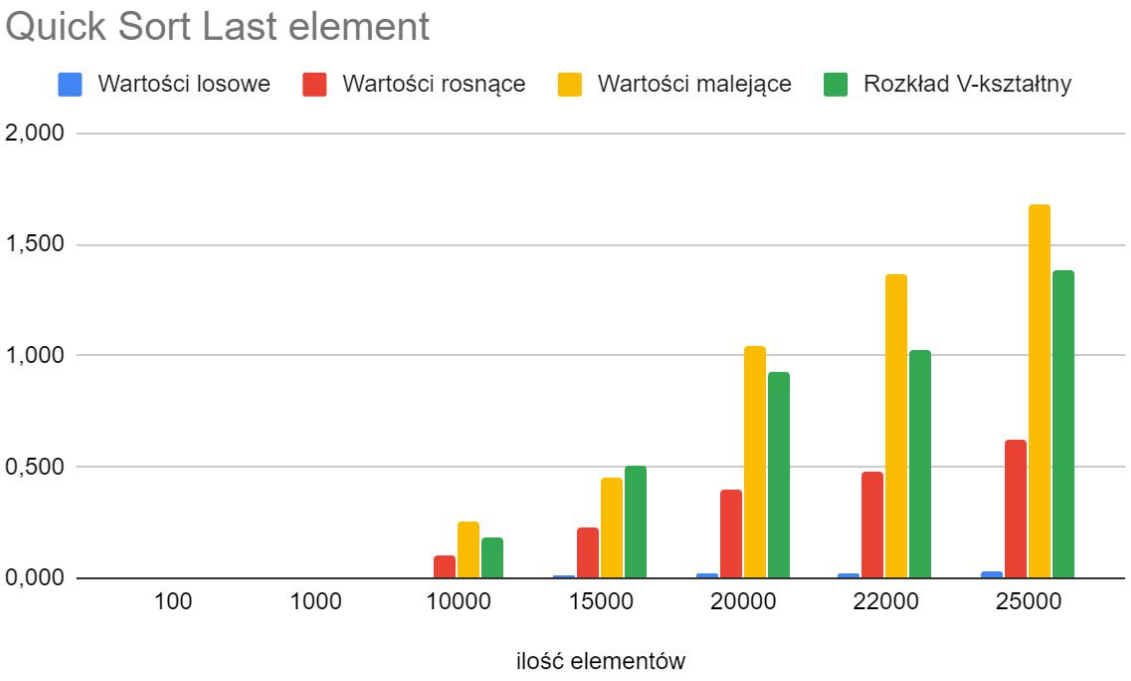


   c. Analiza zachowania algorytmu:
   Dla wartości rosnących Insertion Sort nie wykonuje niepotrzebnych działań na tabeli, przez co przechodzi przez elementy tablicy dokładnie jeden raz i jest to najlepszy przypadek. Najgorszym przypadkiem okazał się być typ danych o wartościach malejących.

3. Algorytm sortowania Quick Sort (Ostatnia wartość w tablicy jest elementem rozdzielającym):
   a. Tabela wyników:

Quick Sort Last element				
ilość elementów	Wartości losowe	Wartości rosnące	Wartości malejące	Rozkład V-kształtny
100	0,000	0,001	0,000	0,000
1000	0,000	0,001	0,002	0,001
10000	0,005	0,101	0,259	0,186
15000	0,013	0,225	0,448	0,508
20000	0,018	0,399	1,046	0,928
22000	0,021	0,483	1,370	1,028
25000	0,033	0,623	1,680	1,386

   b. Wykres wyników:

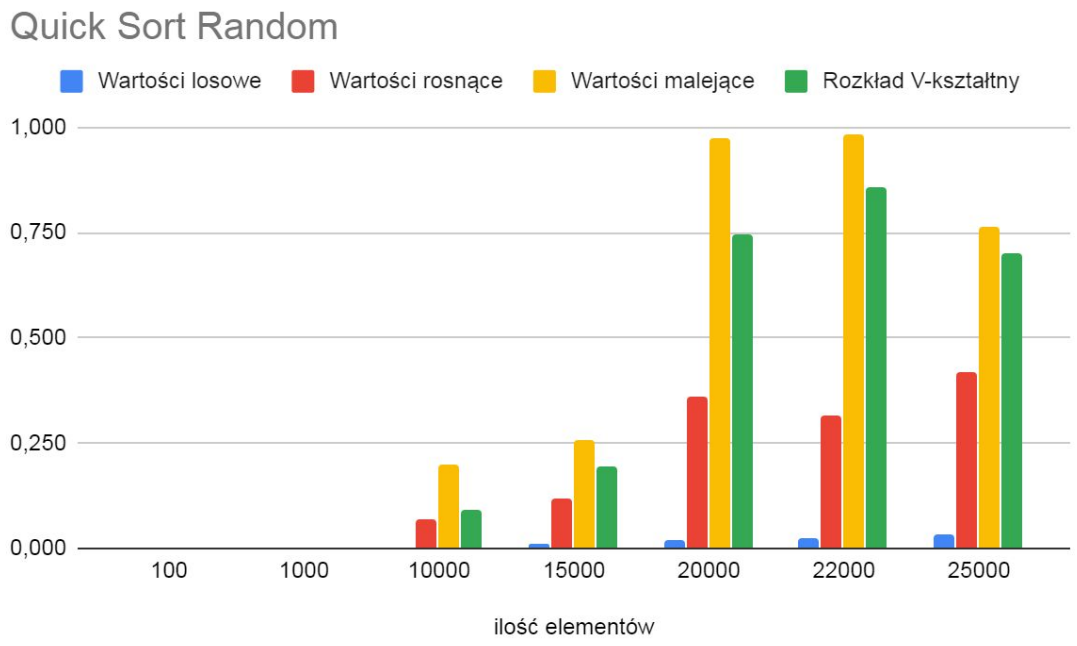


   c. Analiza zachowania algorytmu:
   W przypadku algorytmu Quick Sort z ostatnią wartością tablicy użytą jako element rozdzielający najlepszym przypadkiem są instancje z danymi losowymi. Natomiast najgorszymi instancjami były te z wartościami malejącymi.

4. Algorytm sortowania Quick Sort (Losowa wartość z tablicy jest elementem rozdzielającym):
   a. Tabela wyników:

Quick Sort Random				
ilość elementów	Wartości losowe	Wartości rosnące	Wartości malejące	Rozkład V-kształtny
100	0,000	0,000	0,000	0,000
1000	0,000	0,001	0,001	0,000
10000	0,004	0,071	0,201	0,093
15000	0,011	0,118	0,256	0,196
20000	0,020	0,362	0,976	0,745
22000	0,025	0,314	0,985	0,860
25000	0,032	0,420	0,766	0,701

   b. Wykres wyników:



   c. Analiza zachowania algorytmu:
   W przypadku algorytmu Quick Sort z losową wartością tablicy użytą jako element rozdzielający najlepszym przypadkiem są instancje z danymi losowymi. Natomiast najgorszymi instancjami były ponownie te z wartościami malejącymi. Dodatkowo można zauważyć, że ten algorytm w testowanych przypadkach wykonał sortowanie nawet o 50% szybciej niż poprzedni.

5. Algorytm sortowania Heap Sort:
   a. Tabela wyników:

Heap Sort				
ilość elementów	Wartości losowe	Wartości rosnące	Wartości malejące	Rozkład V-kształtny
100	0,000	0,000	0,000	0,000
1000	0,000	0,000	0,000	0,000
10000	0,002	0,002	0,002	0,001
15000	0,003	0,003	0,003	0,003
20000	0,003	0,003	0,003	0,003
22000	0,004	0,003	0,003	0,003
25000	0,005	0,004	0,004	0,005

   b. Wykres wyników:

   c. Analiza zachowania algorytmu:
   Algorytm Heap Sort wykonał wszystkie sortowania poniżej jednej setnej sekundy. Różnice w czasie wykonywania się sortowania w zależności od typu danych są na tyle niskie, że mieszczą się w granicy błędu pomiarowego.

6. Złożoność obliczeniowa w najgorszym przypadku dla poszczególnych algorytmów:

Selection Sort	**$O(n^2)$**
Insertion Sort	**$O(n^2)$**
Quick Sort Last element	**$O(n^2)$**
Quick Sort Random	**$O(n^2)$**
Heap Sort	**O(nlogn)**