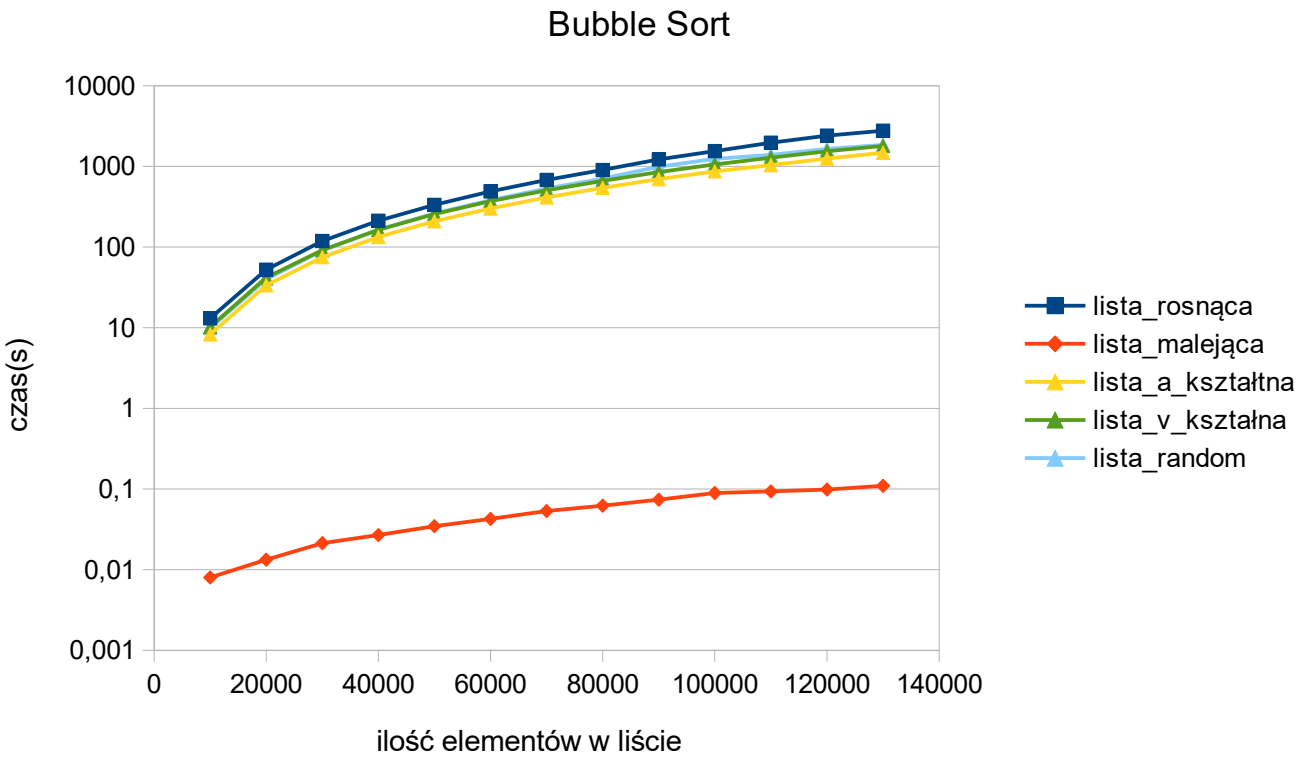
Algorytmy i struktury danych – Sortowanie

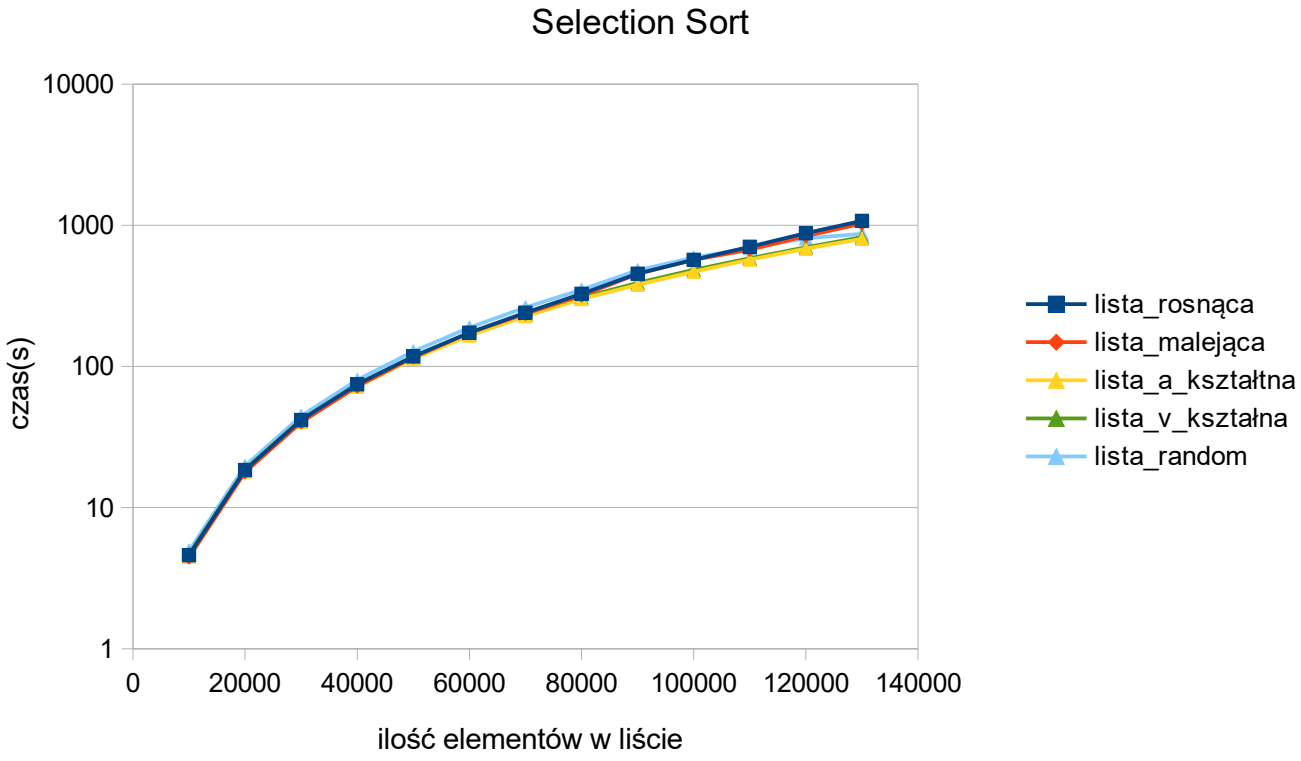
Filip Rosiak

1. Wykresy czasu obliczeń dla każdej metody sortowania

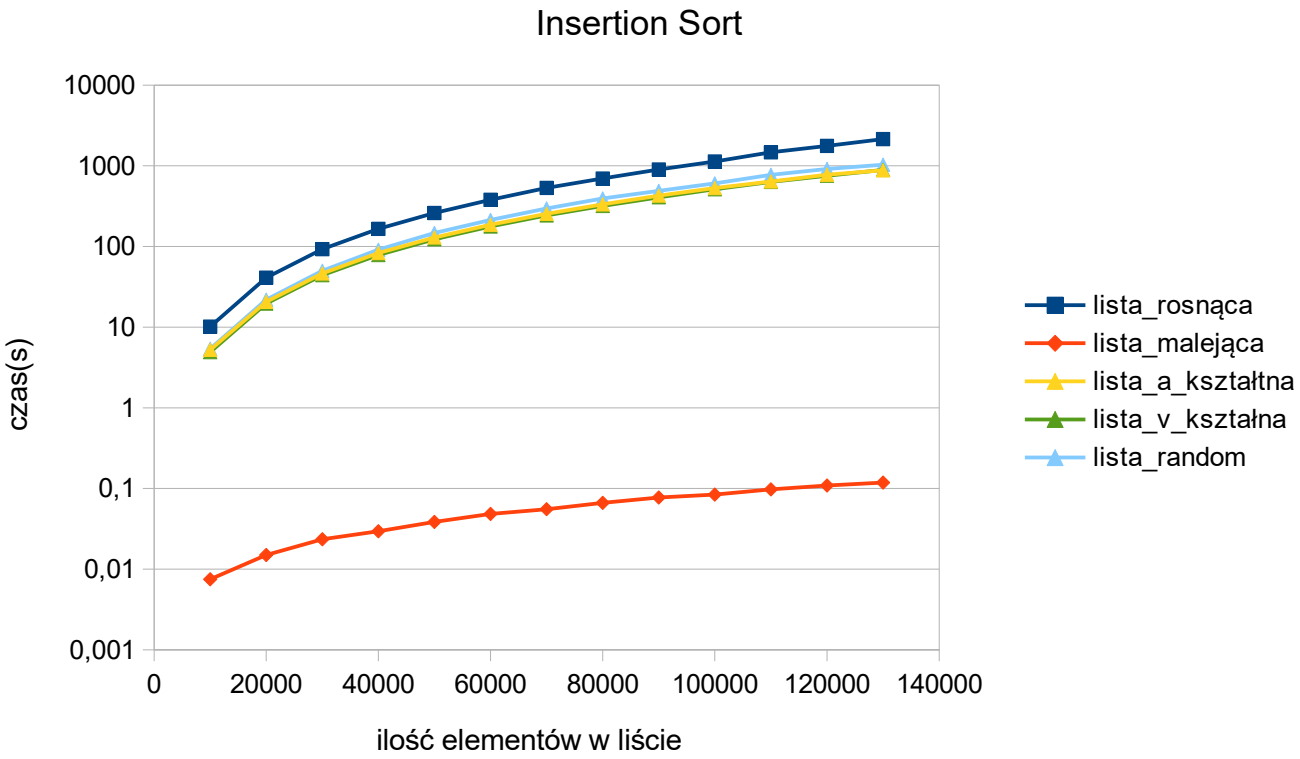
a) Bubble Sort



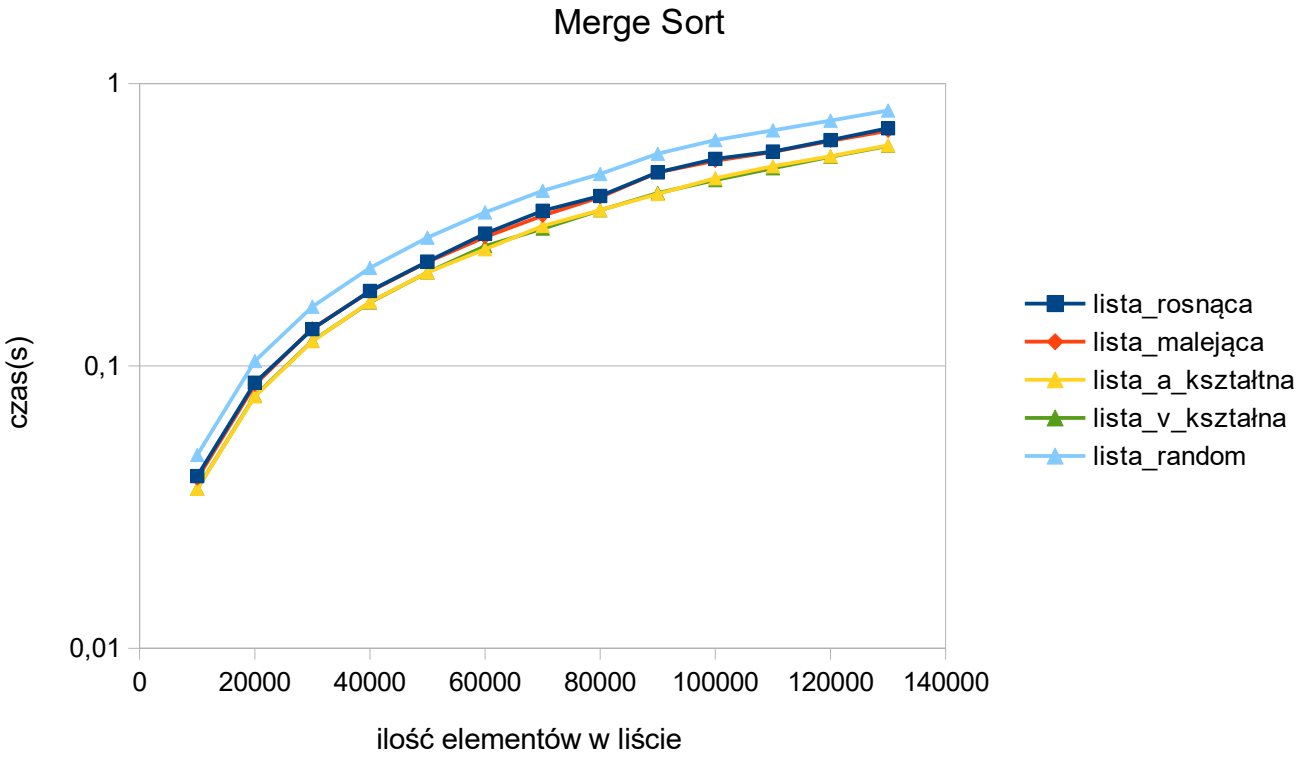
b) Selection Sort



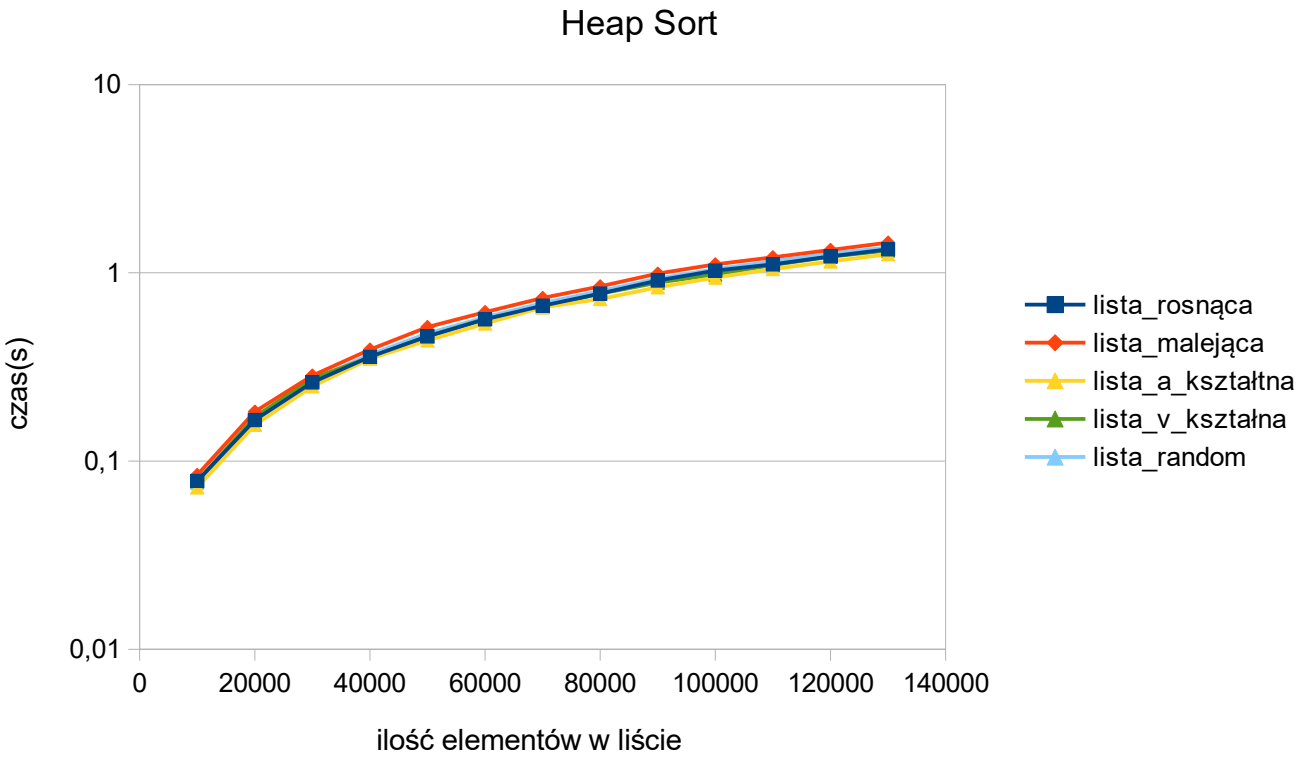
c) Insertion Sort



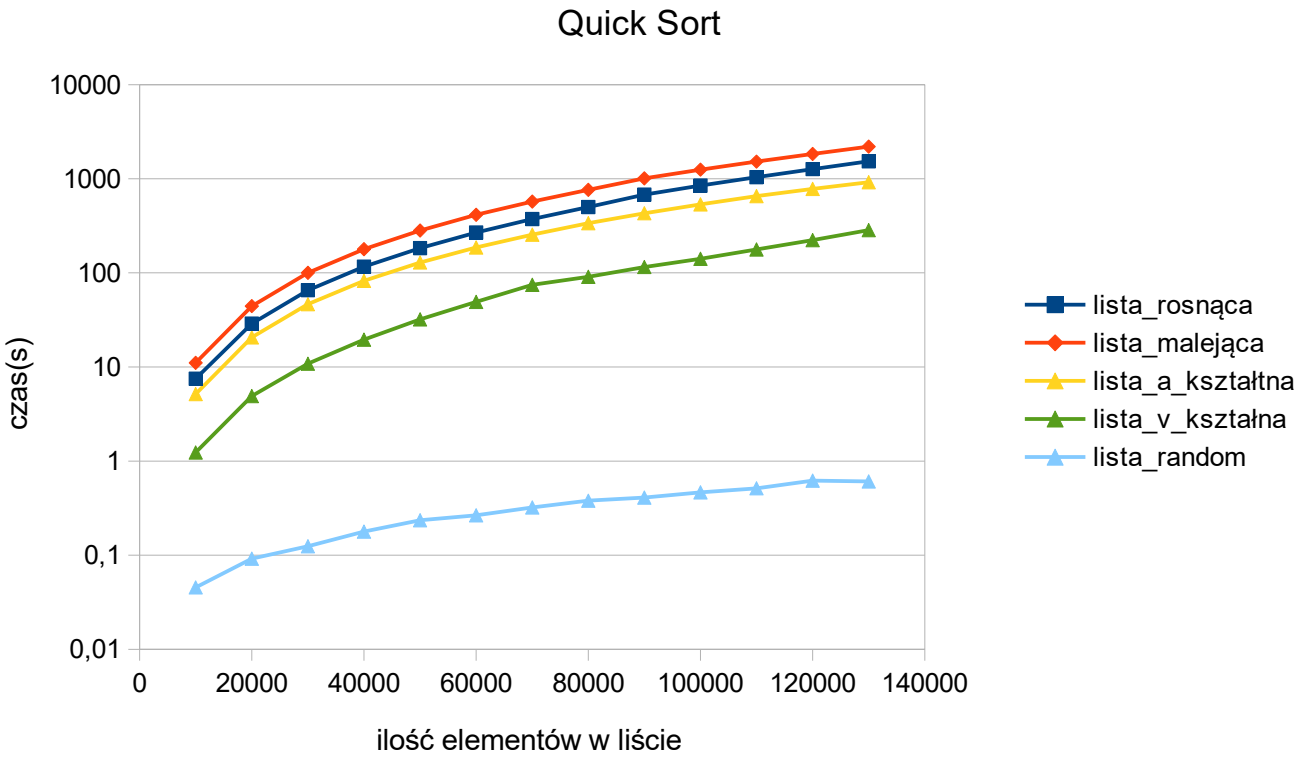
d) Merge Sort



e) Heap Sort

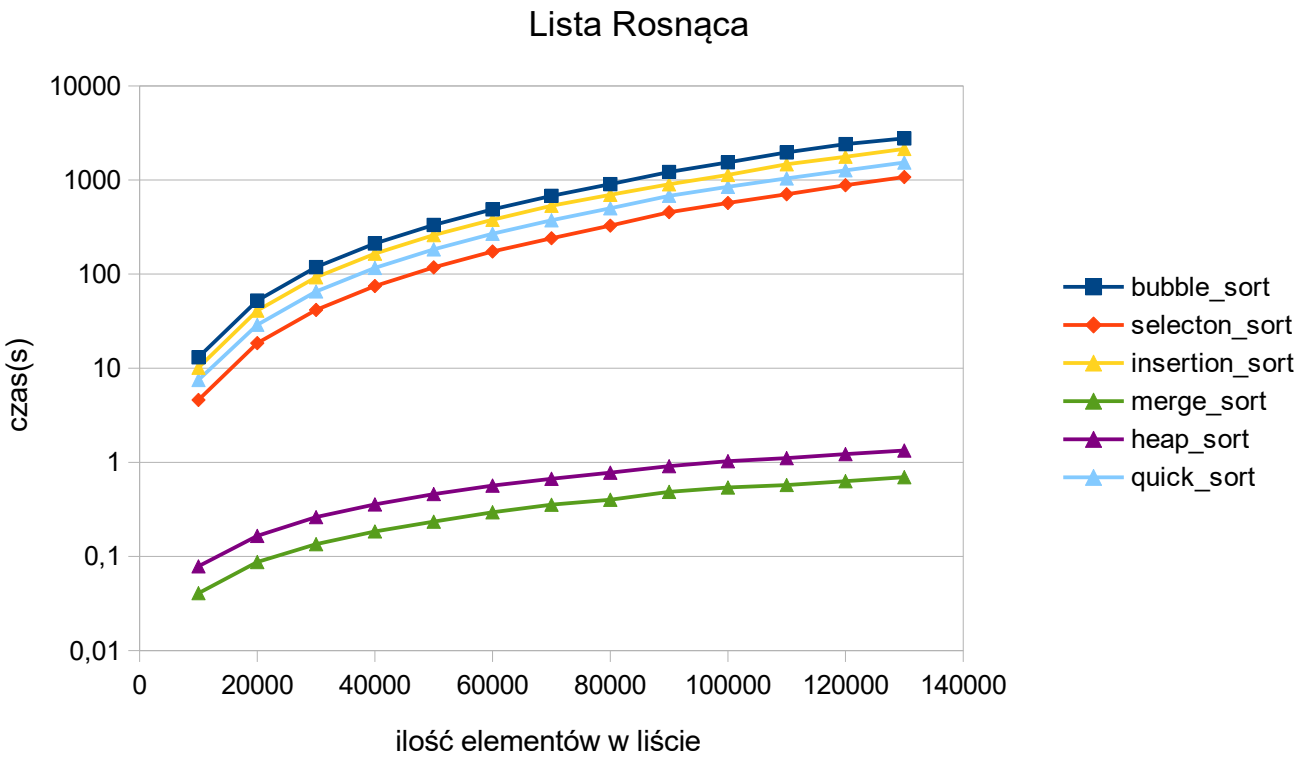


f) Quick Sort

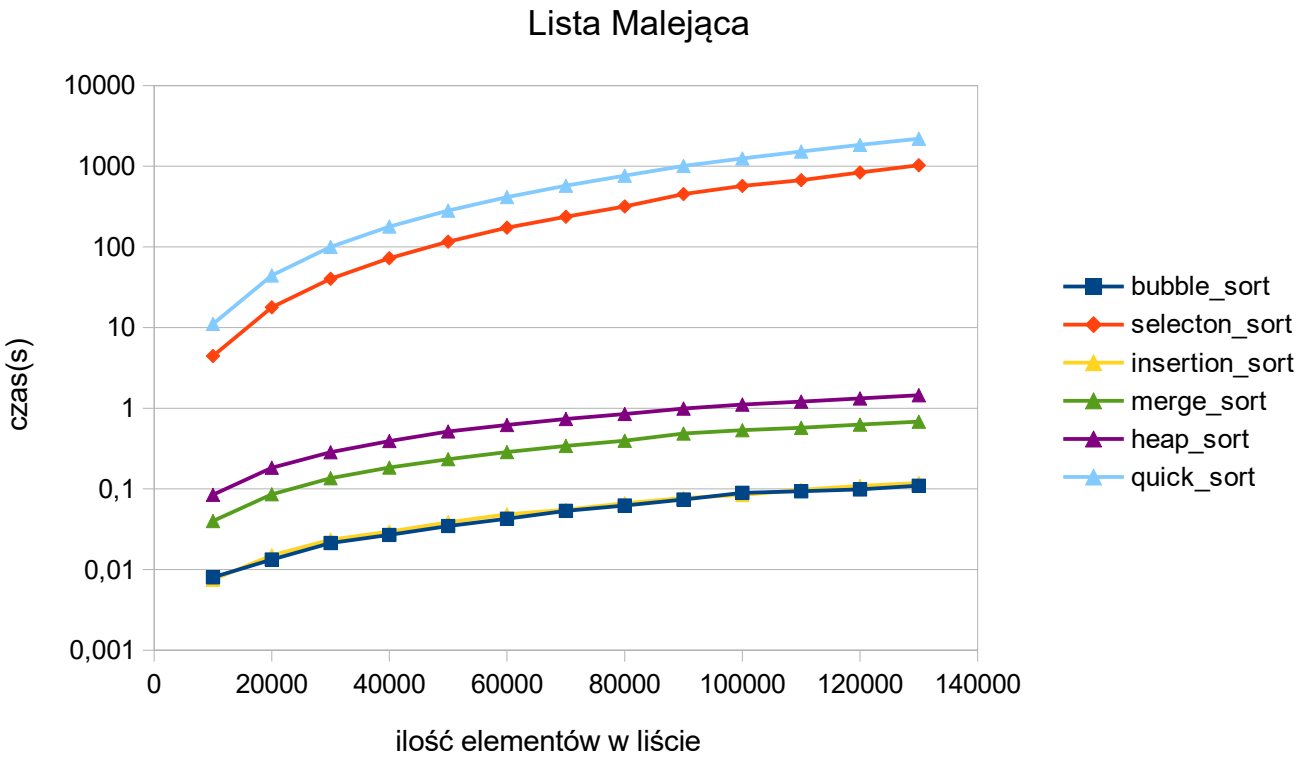


2. Wykresy czasu obliczeń dla każdej postaci danych wejściowych

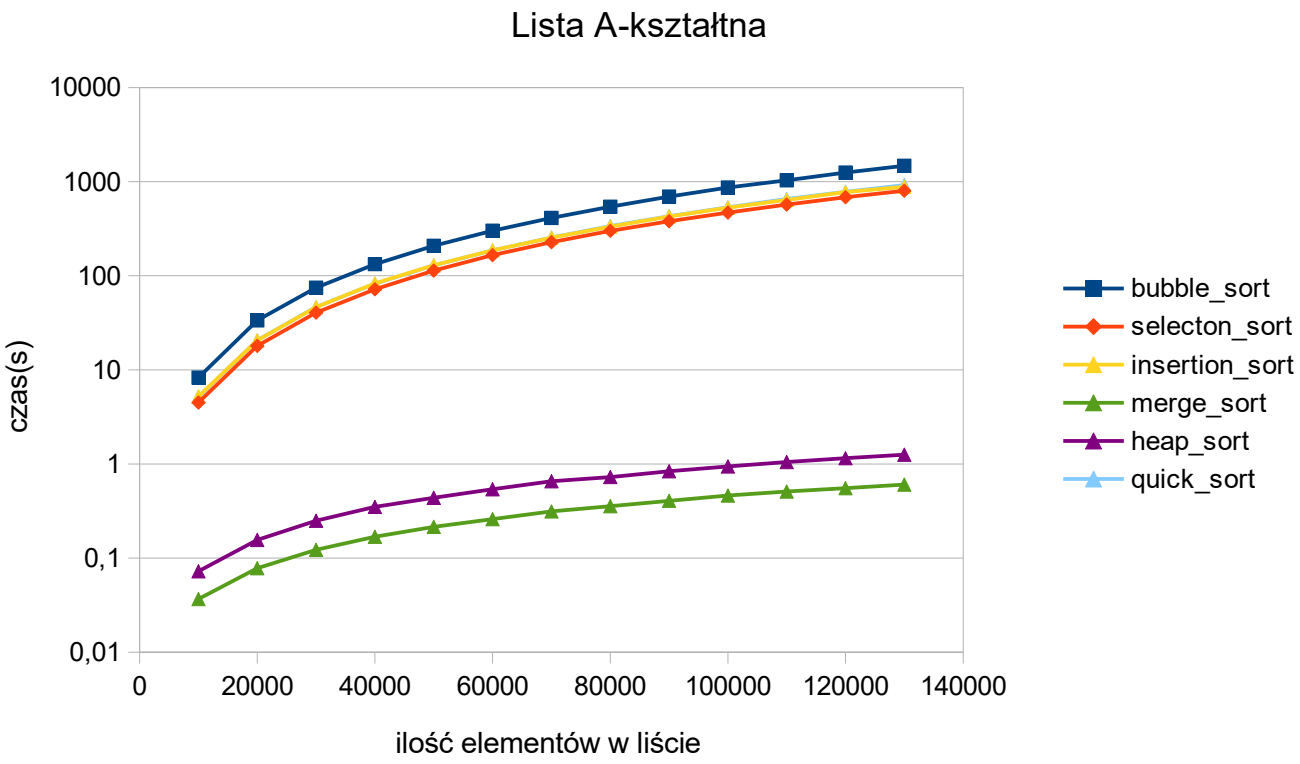
a) Lista Rosnąca



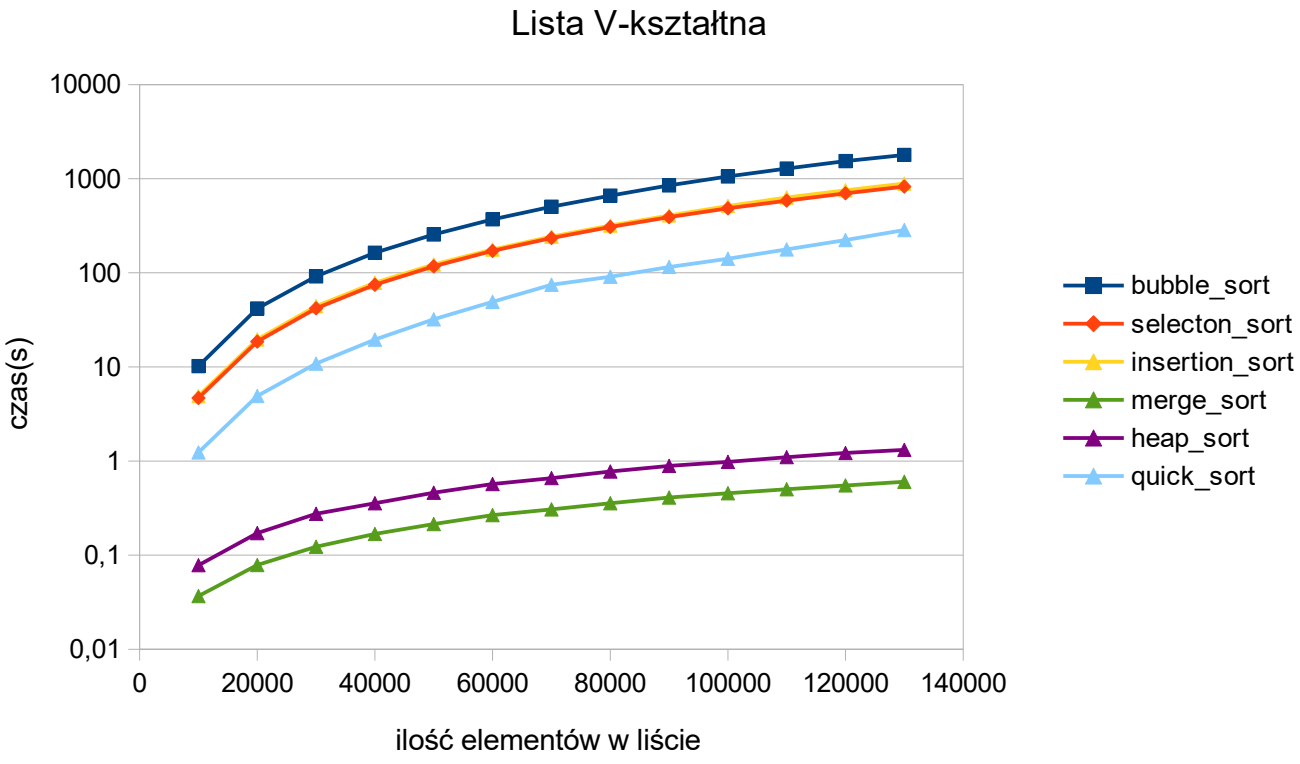
b) Lista Malejąca



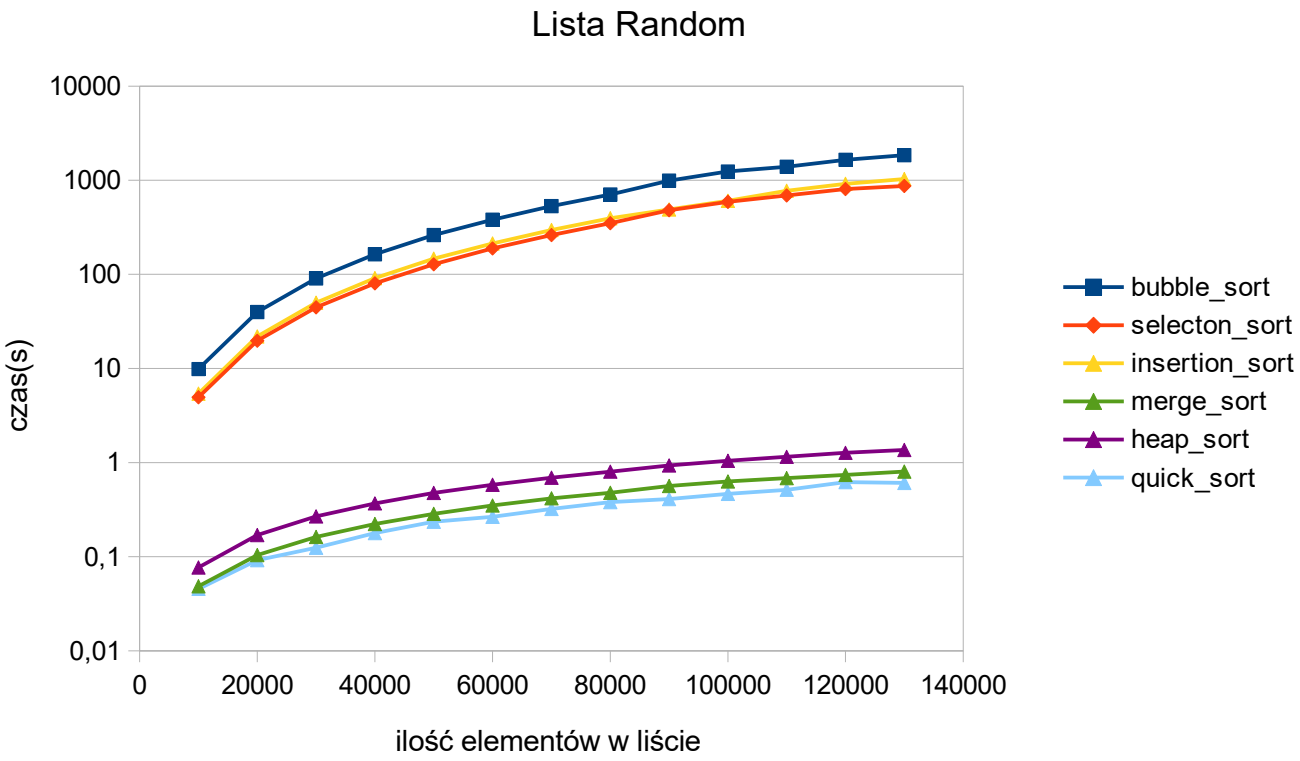
c) Lista A-kształtna



d) Lista V-kształtna

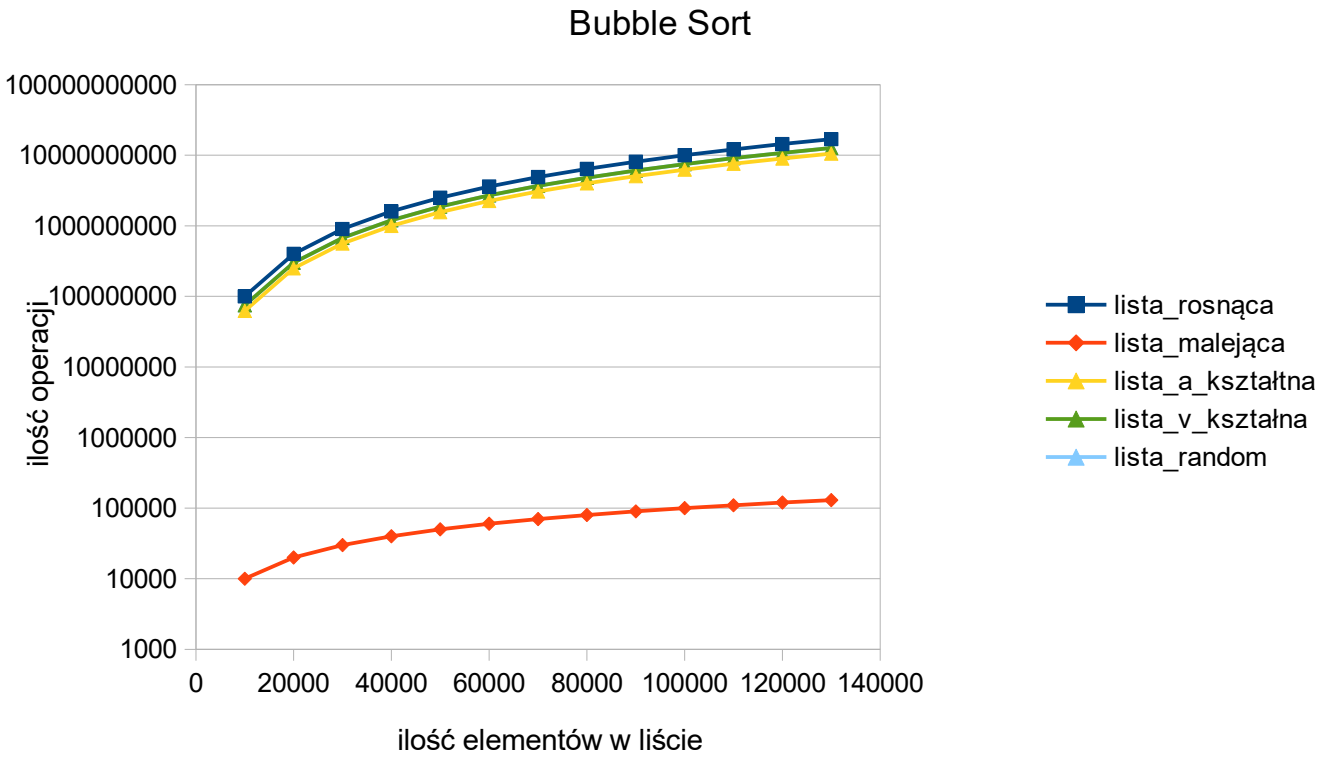


e) Lista Losowa

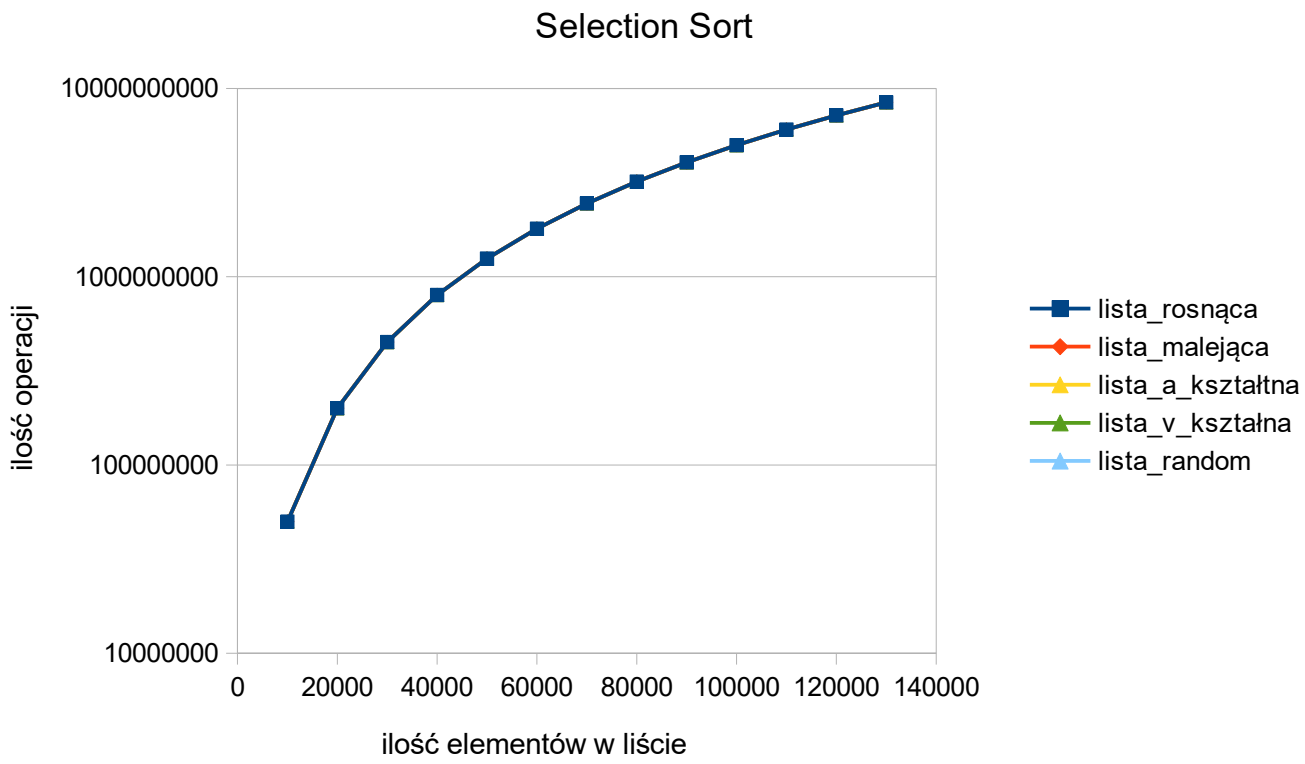


3. Wykresy ilości operacji dla każdej metody sortowania

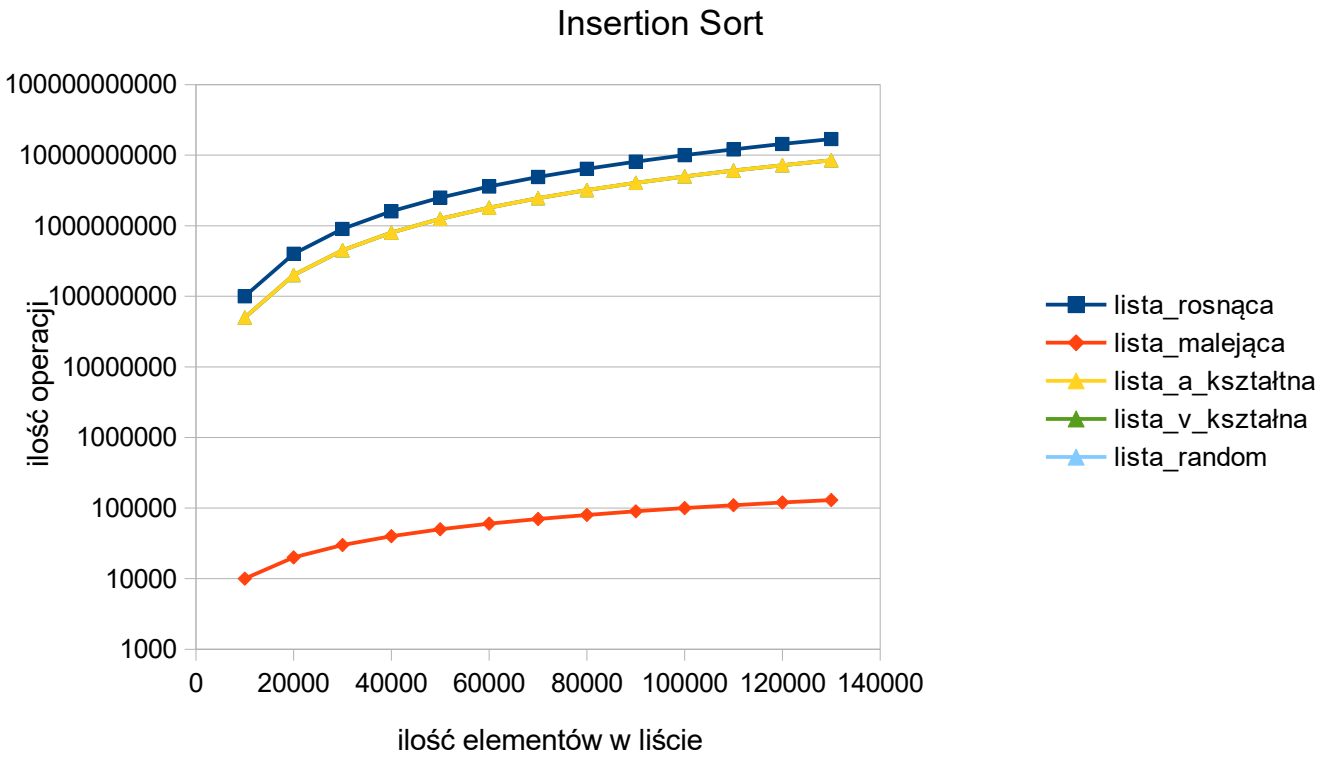
a) Bubble Sort



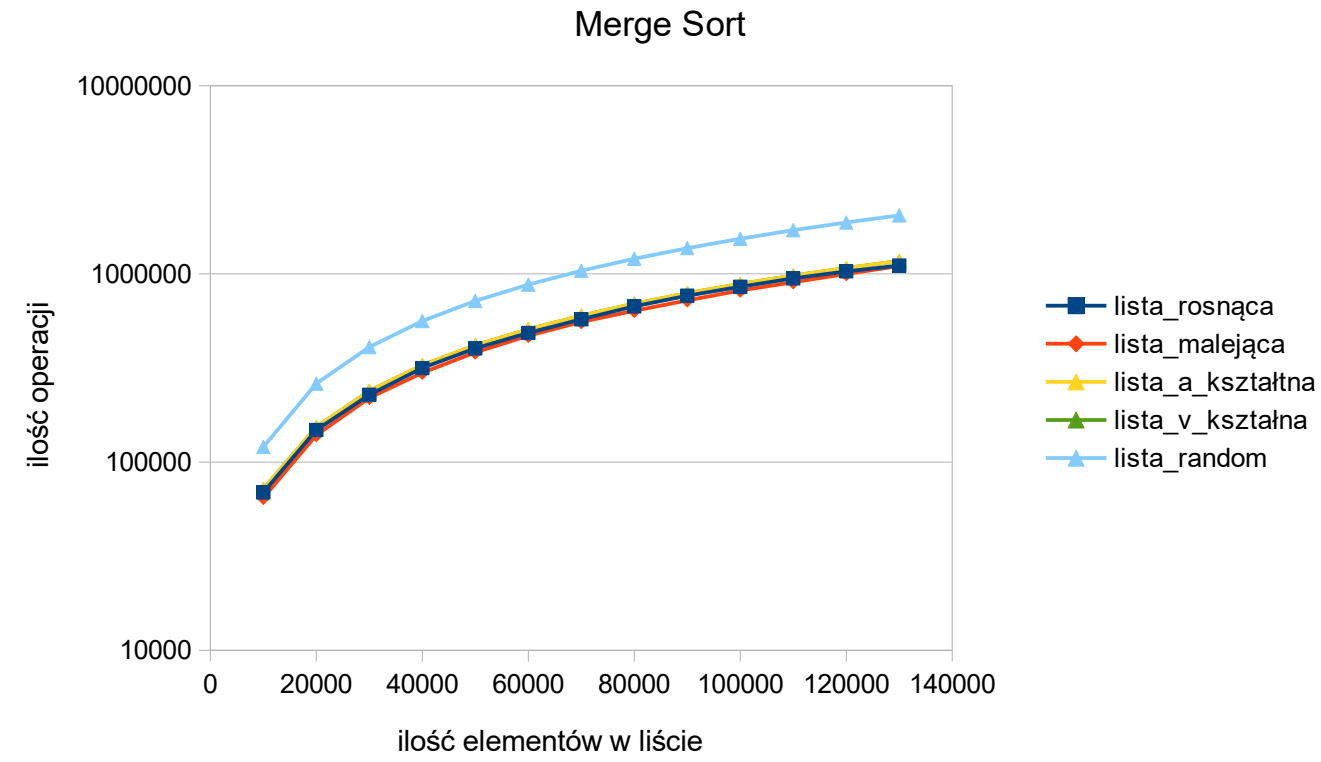
b) Selection Sort



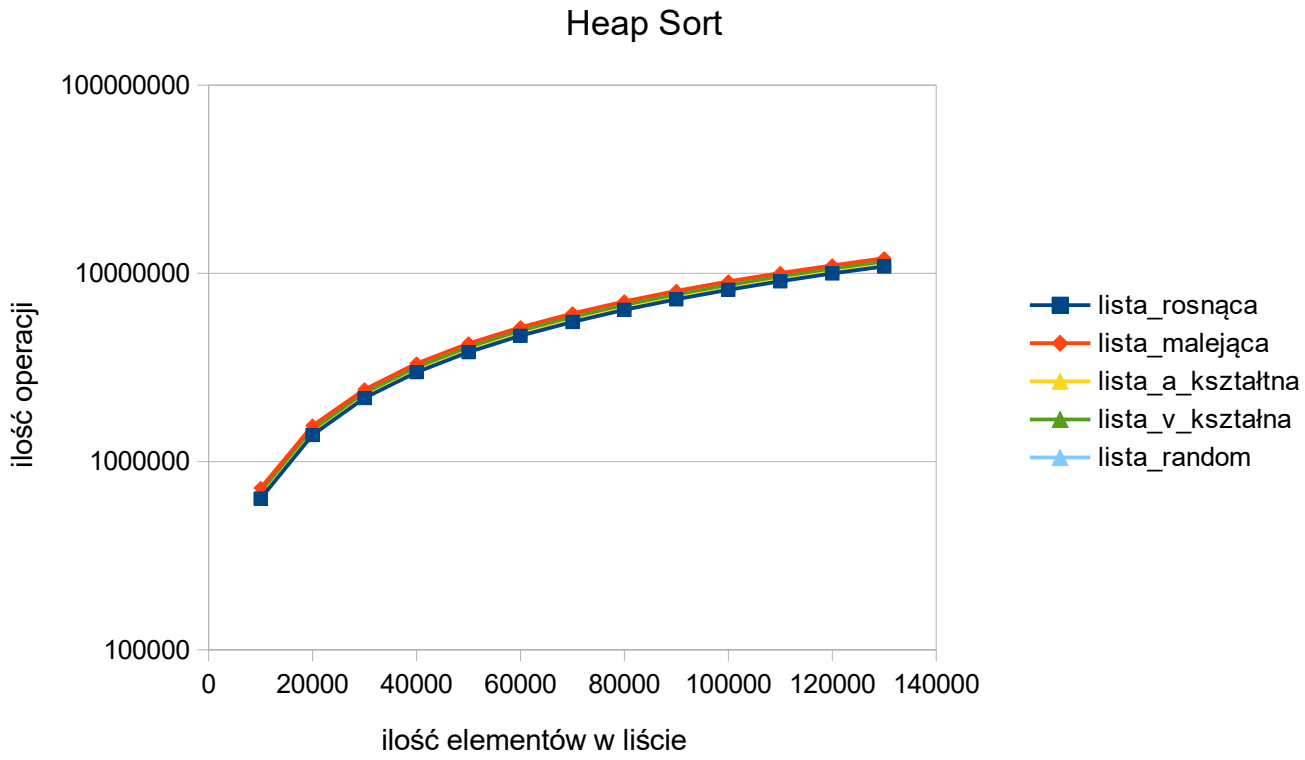
c) Insertion Sort



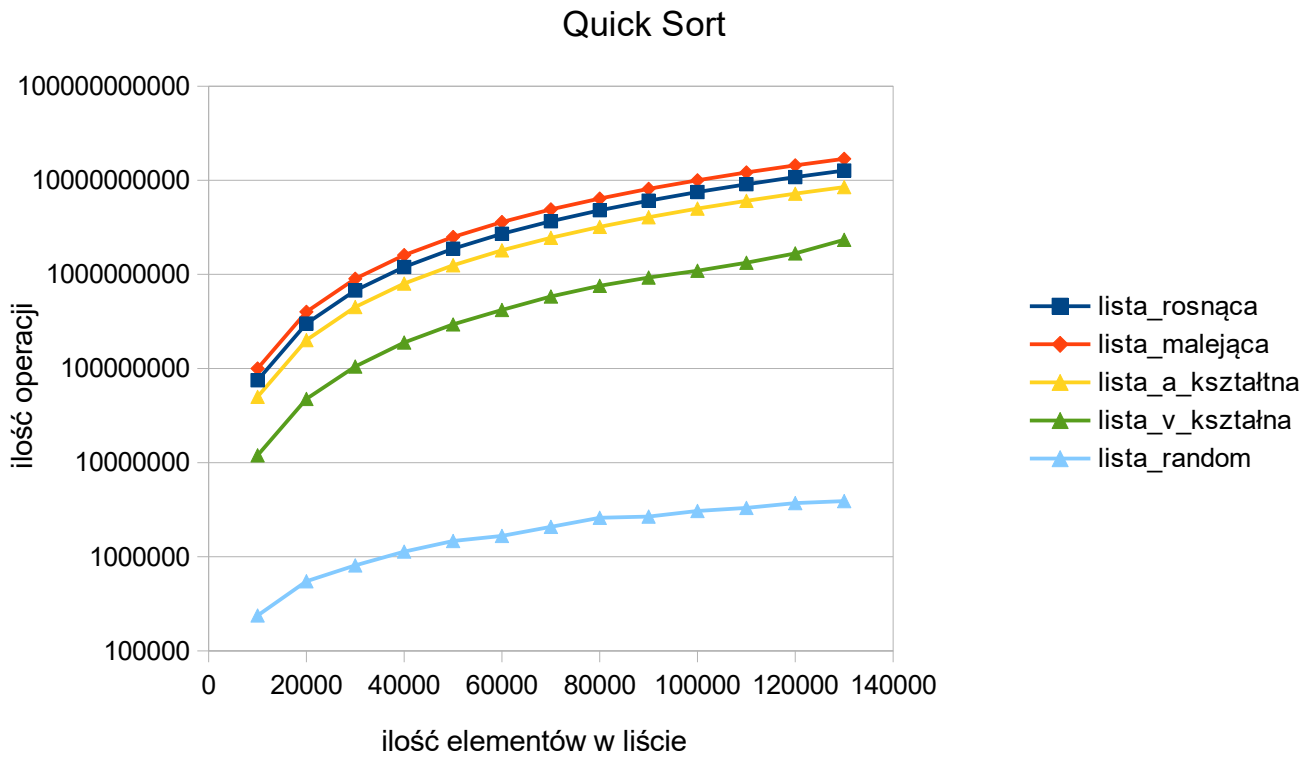
d) Merge Sort



e) Heap Sort



f) Quick Sort



4. Złożoności obliczeniowe algorytmów

L.p.	Nazwa algorytmu	klasa złożoności		
		optymistyczna	średnia	pesymistyczna
1	Bubble Sort	O(n)	O(n2)	O(n2)
2	Selection Sort	O(n2)	O(n2)	O(n2)
3	Insertion Sort	O(n)	O(n2)	O(n2)
4	Merge Sort	O(n log n)	O(n log n)	O(n log n)
5	Heap Sort	O(n log n)	O(n log n)	O(n log n)
6	Quick Sort	O(n log n)	O(n log n)	O(n2)

Powyższe wykresy zostały wykonane w skali logarytmicznej przy użyciu tablic o długości 10000 – 130000 elementów z różnicą 10000, czyli 13 punktów pomiarowych, z czego każdy jest średnią 7 pomiarów czasu dla identycznych list.

Po prześledzeniu wykresów można dojść do wniosku, że ich kształty we wszystkich przypadkach pokrywają się z reprezentacjami graficznymi ich złożoności obliczeniowych.

Ciekawymi wyjątkami są bubble sort oraz insertion sort dla tablicy malejącej gdzie ich złożoność jest znacznie niższa od tej dla najgorszego przypadku. Jest to spowodowanie ich budową – dla bubble sorta wewnętrzna pętla wykonuje się tylko raz, natomiast dla insertion sorta takie ułożenie danych sprawia, że wewnętrzna pętla nie wykonuje się, ponieważ już pierwszy sprawdzany wyraz nie spełnia warunku wywołania, co zmniejsza złożoność z O(n2) do O(n).

Ciekawy okazują się też wykresy dla selection sorta, które są praktycznie takie same. Spowodowane jest to tym, iż algorytm ten nie uwzględnia posortowania zbioru.

Dla algorytmu merge sort najdłużej trwa sortowanie tablicy nieuporządkowanej. Jednakże czasy nie różnią się zbyt wiele, co może sugerować, że algorytmu tego nie specjalnie zakłóca rozkład danych wejściowych.

W algorytmie heap sort czasy również nie różnią się zbytnio od siebie, jednak potrzebuje on odrobinę więcej czasu na posortowanie niż merge sort. Zaletą sortowanie w miejscu – poprzedni algorytm wymaga dodatkowej pamięci, co może być problematycznie przy sortowaniu dużych zbiorów danych.

W algorytmie quick sort, który jako pivot przyjmował skrajnie prawy element praktycznie tylko jeden z pięciu przypadków osiągnął najlepszą możliwą złożoność tj. O(n) i była to lista losowych elementów. Dla danych rosnących bądź malejących, quick sort trafiał na najmniejszy oraz największy element co sprawia, że algorytm odejmie tylko jedną liczbę w kolejnym wywołaniu rekurencyjnym.