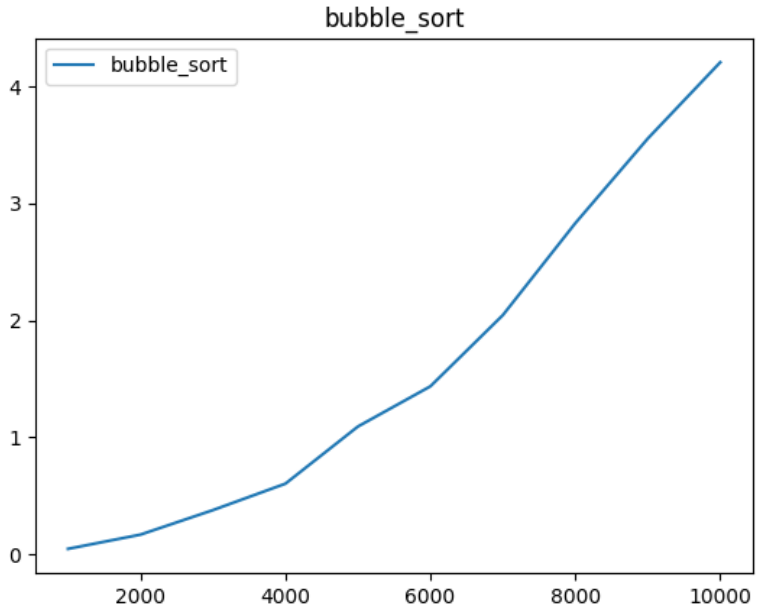
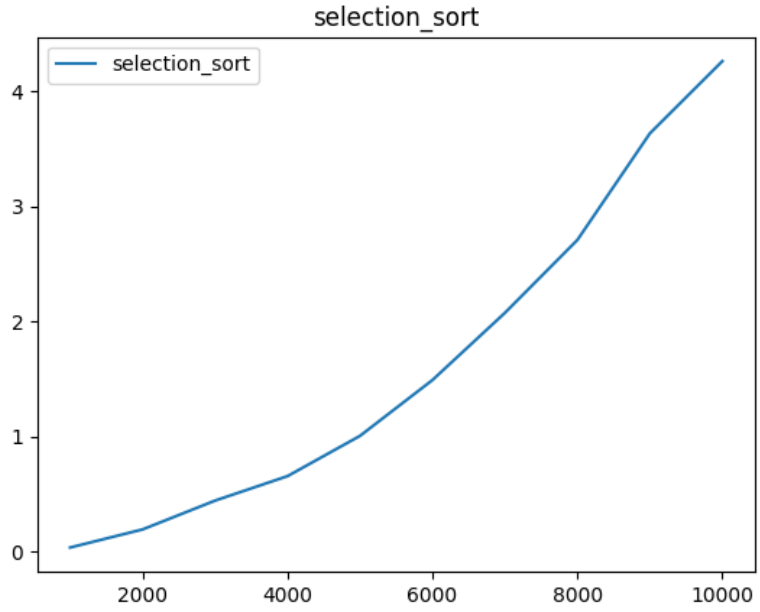
Algorytmy sortujące

Adam Kraś 325177 Krzysztof Król 325178

Sortowania wolne

Z grupy sortowań o złożoności kwadratowej wybraliśmy sortowanie bąbelkowe (ang. bubble sort) i sortowanie przez wybieranie (ang. selection sort). Sortowanie bąbelkowe zostało wykonane przez Adama Kraś, a przez wybieranie przez Krzysztofa Króla. Poniżej przedstawiamy wykresy czasowe tych sortowań dla list ze słowami z pliku pan_tadeusz.txt:

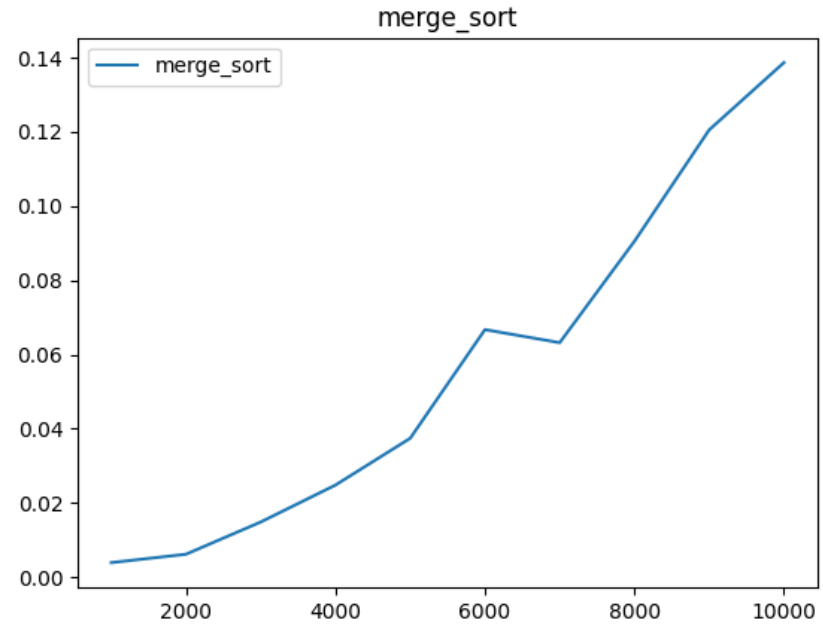


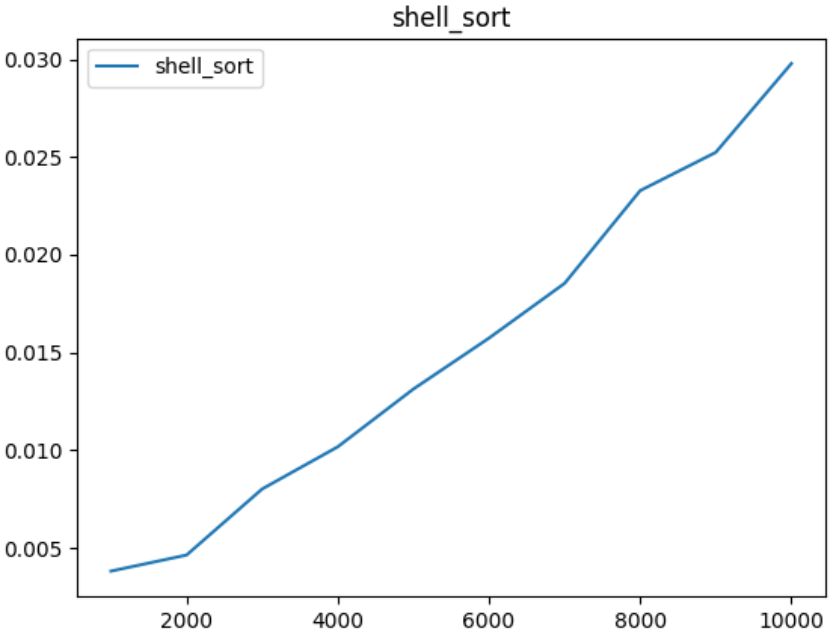


Oba z tych sortowań nie są zbyt przydatne, gdyż z uwagi na ilość wykonywanych przez nie porównań nie są one zbyt wydajne, a ich złożoność obliczeniowa to $O(n^2)$.

Sortowania szybkie

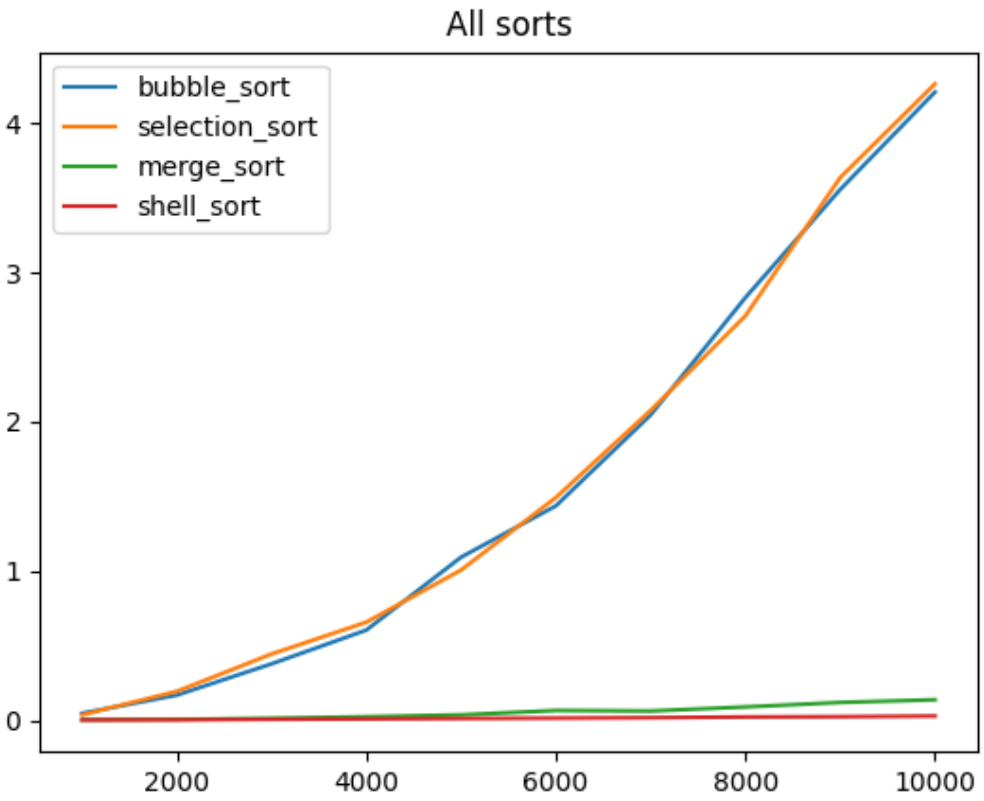
Z grupy sortowań szybkich zdecydowaliśmy się zaimplementować sortowanie przez scalanie (ang. merge sort) oraz sortowanie Shella (ang. shellsort). Adam Kraś wykonał sortowanie Shella, natomiast Krzysztof Król sortowanie przez scalanie. Poniżej przedstawiamy wykresy czasowe tych sortowań dla takich samych list jak w przypadku sortowań o złożoności kwadratowej.



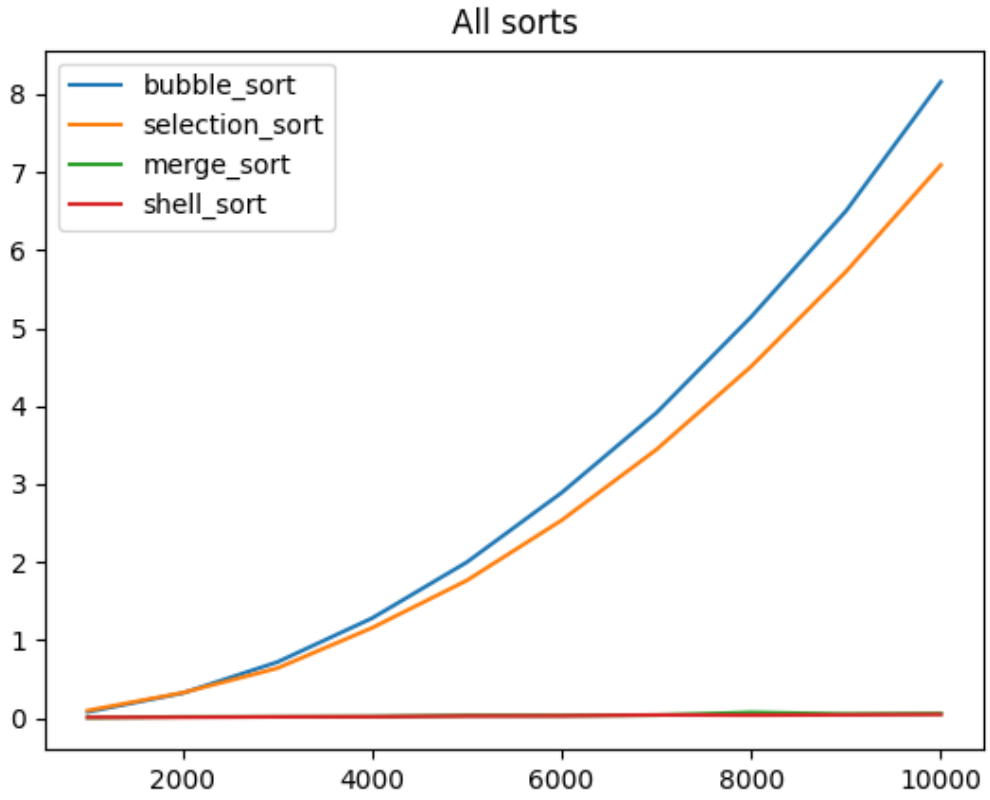


Jak widać przy dokładnie takich samych listach te sortowania poradziły sobie znacząco lepiej, co widać po znacząco krótszym czasie potrzebnym do posortowania tychże list. Nie jest to zaskoczeniem, gdyż oba mają w przybliżeniu złożoność obliczeniową O(n log n).

Porównanie

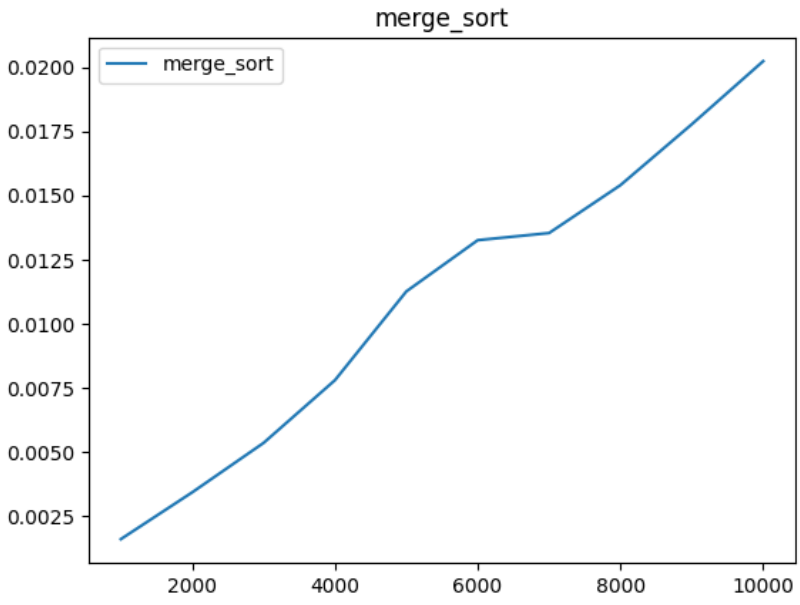


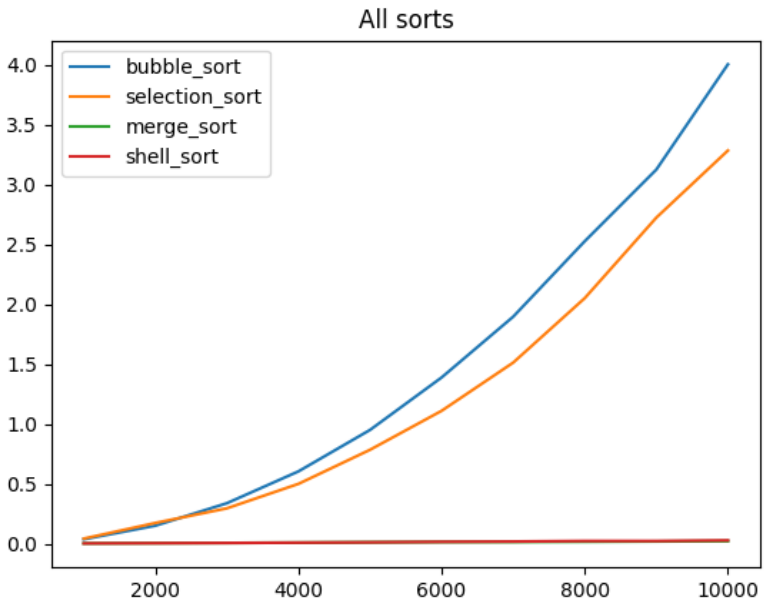
Powyżej prezentujemy wykres na którym to widać porównanie wyników dla poszczególnych wykonanych przez nas sortowań. Wyniki te pochodzą z tego samego komputera i były wykonywane w tym samym momencie. W celu jak największego zniwelowania potencjalnych zakłamań wyników procesor komputera na którym wykonywane było sortowanie był nieobciążony innymi działaniami. W celu sprawdzenia czy wyniki z maszyny testowej są takie same na innych komputerach wykonaliśmy też taki wykres:



Co ciekawe tutaj wszystkie sortowania są trochę wolniejsze niż na maszynie testowej, poza sortowaniem przez scalanie, które jako jedyne tutaj „przyspieszyło”.

Problematyczne okazały się metody pop() i append() dla list w Python (wersja 3.8.10), po zmianie implementacji na taką z iterowaniem, sortowanie przyspieszyło do poziomu prezentowanego przez sortowanie shella, co widać poniżej:





Podział pracy

Adam Kraś

- bubble_sort.py
- shell_sort.py
- time_count.py
- read_file.py

Krzysztof Król

- selection_sort.py
- merge_sort.py
- graph_maker.py
- main.py