Konrad Czarny
235778
Gr. Piątek 9^{15}

Prowadzący:
Mgr inż. Marta Emirsajłow

Projektowanie algorytmów i metody sztucznej inteligencji

Projekt 1

1. Wprowadzenie

1.1 Opis projektu

Celem projektu było przeanalizowanie trzech algorytmów sortowania pod względem wydajności, czyli czasu, jaki jest potrzebny na posortowanie określonych ilości danych. W projekcie analizowane były następujące algorytmy:

- Sortowanie szybkie (Quick sort)
- Sortowanie przez scalanie (Merge sort)
- Sortowanie introspektywne (Intro sort)

Algorytmy były testowane kolejno na poszczególnych porcjach danych, gdzie każdy element był typu całkowitego:

- 10 000 elementów
- 50 000 elementów
- 100 000 elementów
- 500 000 elementów
- 1 000 000 elementów

Każde porcje danych były kolejno wstępnie posortowane:

- 0%
- 25%
- 50%
- 75%
- 95%
- 99%
- 99.7%
- Posortowane w odwrotnej kolejności

1.2 Opis algorytmów

Sortowanie przez scalanie

Algorytm sortowania przez scalanie, ang. Merge sort stosuje metodę dziel i zwyciężaj. Działanie algorytmu polega na dwóch krokach:

- Dziel rekurencyjnie zestaw danych, aż do momentu uzyskania jednoelementowych zestawów danych
- Scalaj posortowane zestawy danych, aż do uzyskania pełnego zestawu danych

Złożoność obliczeniowa:

- Najlepszy przypadek : O(n log n)
- Typowy przypadek : O(n log n)
- Najgorszy przypadek : O(n log n)

Sortowanie Szybkie

Sortowanie szybkie, ang. Quick sort jest algorytmem opartym na zasadzie dziel i zwyciężaj. Polega na wybraniu elementu osiowego(ang. Pivot), następnie zestaw danych jest dzielony na dwie części. Pierwsza z nich zawiera elementy mniejsze od elementu osiowego, druga zaś większe lub równe, element osiowy znajduje się pomiędzy nimi. Proces dzielenia jest rekurencyjny, aż do momentu uzyskania jednoelementowych zestawów danych. Wybór elementu osiowego wpływa na równomierność podziału na podzestawy.

Złożoność obliczeniowa:

- Najlepszy przypadek: O(n log n)
- Typowy przypadek: O(n log n)
- Najgorszy przypadek: $O(n^2)$

Sortowanie przez scalanie

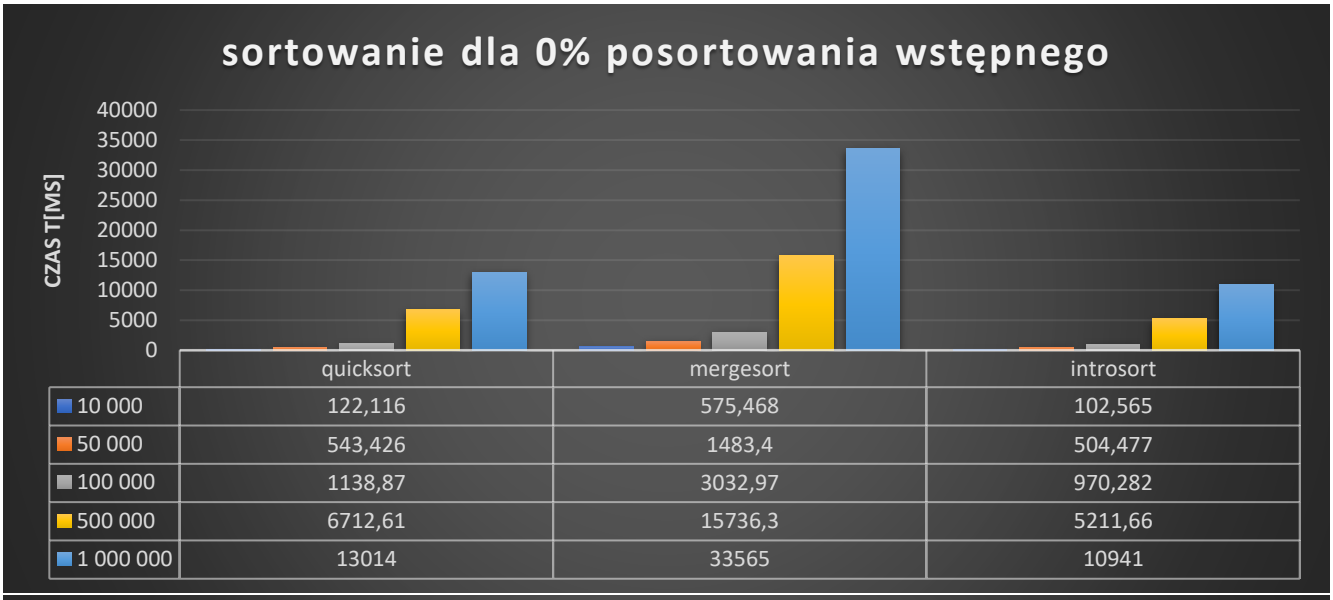
Jest to metoda hybrydowa, będąca połączeniem sortowania szybkiego oraz sortowania przez kopcowanie. Sortowanie introspektywne pozwala uniknąć najgorszego przypadku sortowania szybkiego, czyli nierównomierny podział tablicy w przypadku, gdy jako element osiowy zostanie wybrany element najmniejszy lub największy.

Złożoność obliczeniowa:

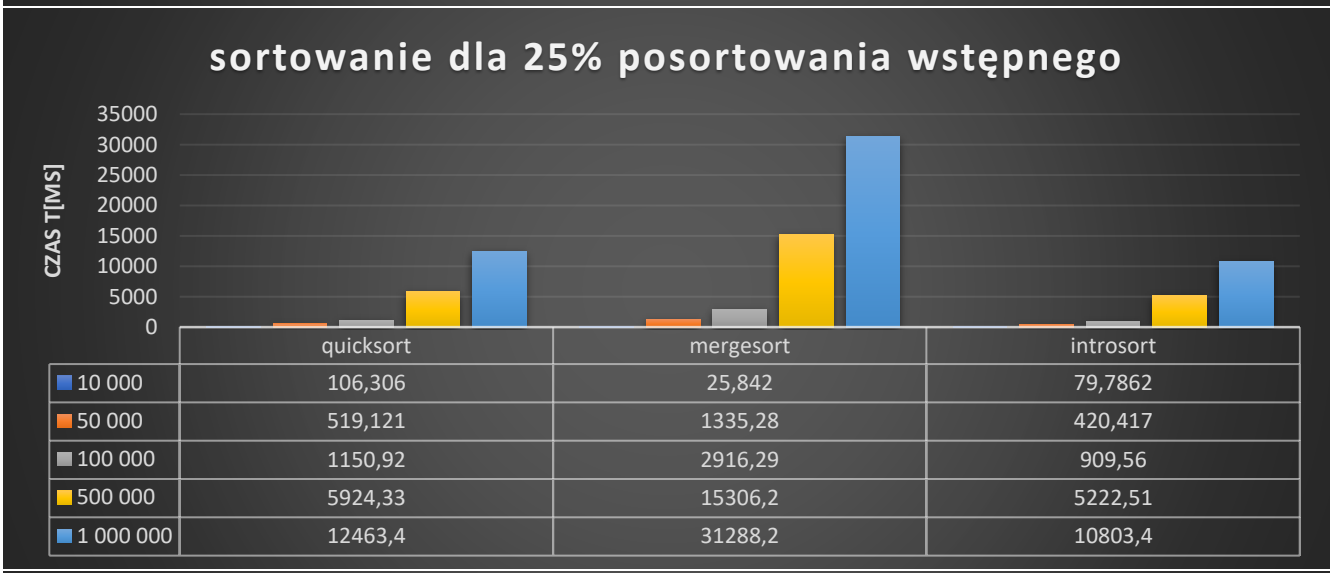
- Najlepszy przypadek: O(n log n)
- Typowy przypadek: O(n log n)
- Najgorszy przypadek: O(n log n)

2. Przebieg testów

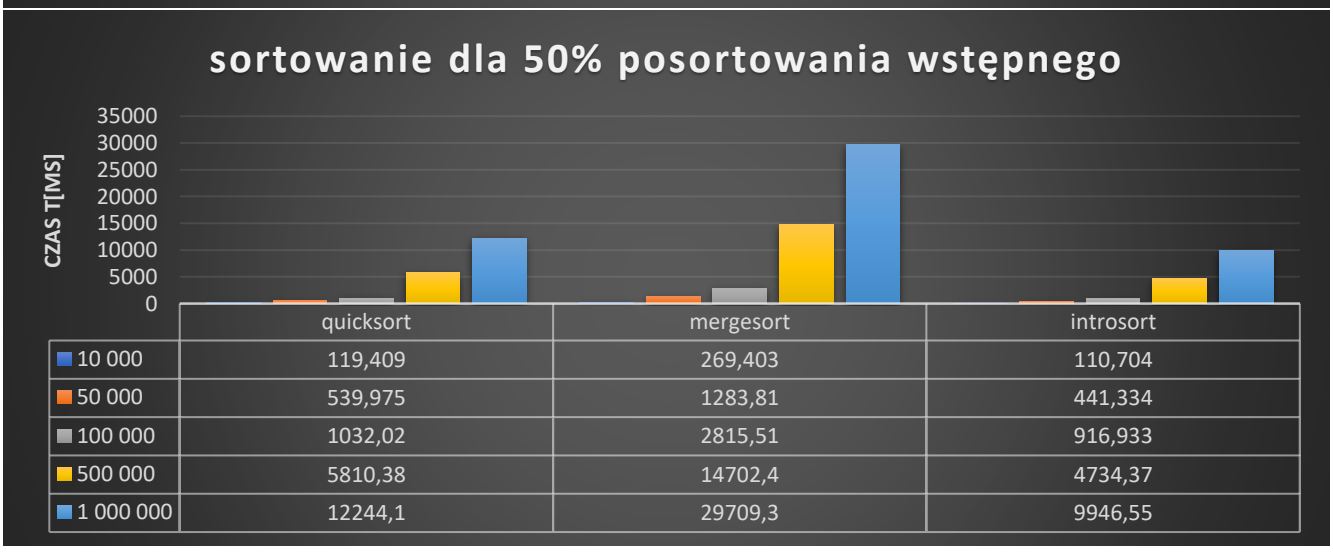
Testy zostały wykonane na dynamicznej tablicy dwuwymiarowej, która mieściła 100 tablic dynamicznych typu całkowitego. Czasy zostały zmierzone na sortowaniu 100 tablic, jedna po drugiej.



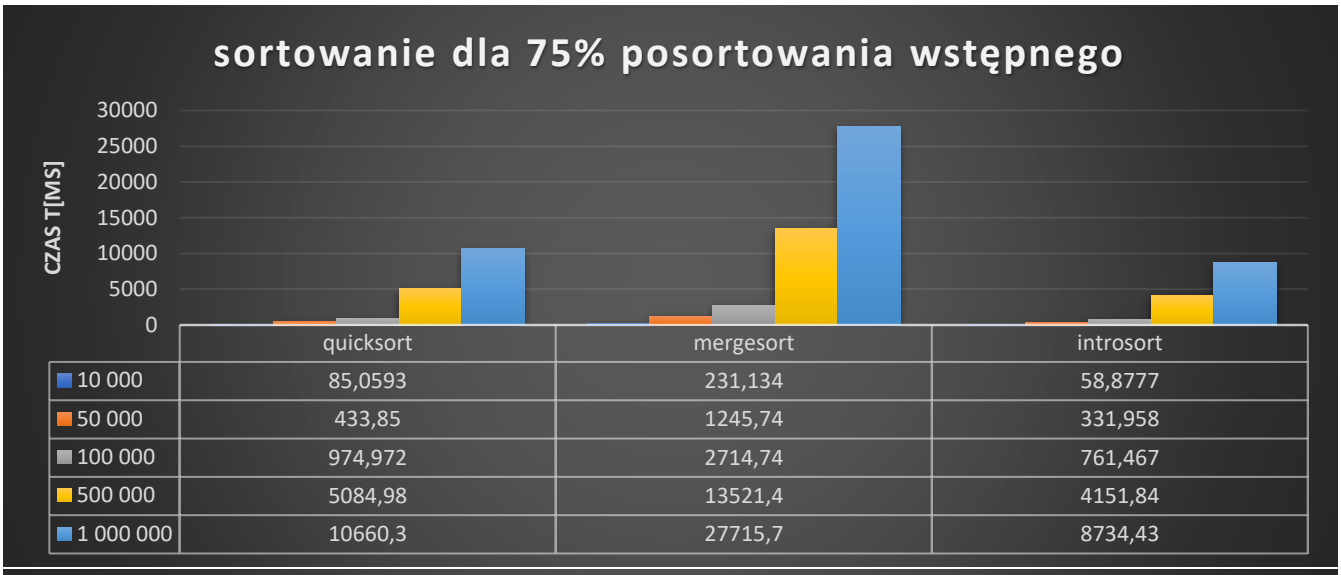
	quicksort	mergesort	introsort
10 000	122,116	575,468	102,565
50 000	543,426	1483,4	504,477
100 000	1138,87	3032,97	970,282
500 000	6712,61	15736,3	5211,66
1 000 000	13014	33565	10941



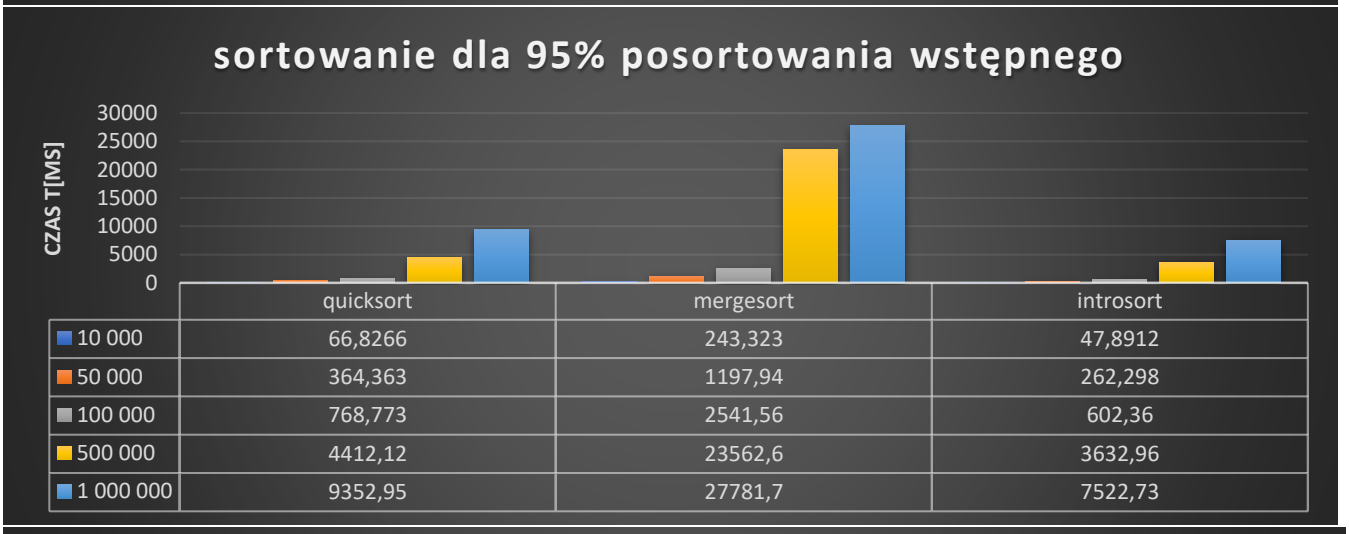
	quicksort	mergesort	introsort
10 000	106,306	25,842	79,7862
50 000	519,121	1335,28	420,417
100 000	1150,92	2916,29	909,56
500 000	5924,33	15306,2	5222,51
1 000 000	12463,4	31288,2	10803,4



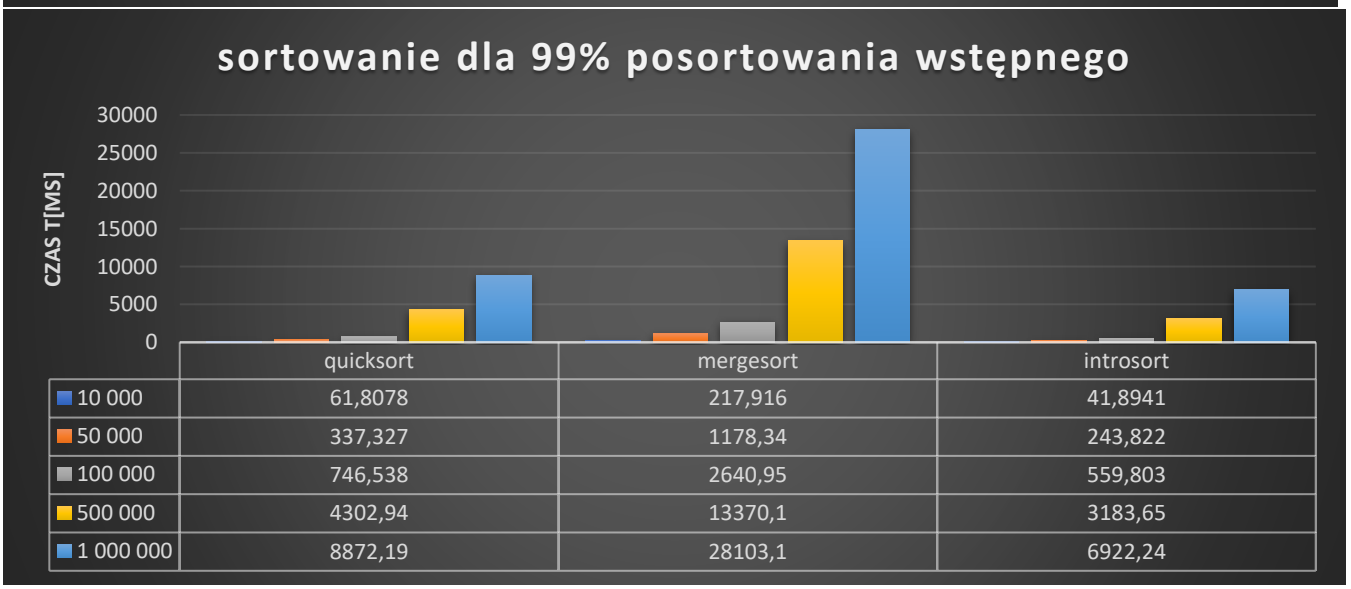
	quicksort	mergesort	introsort
10 000	119,409	269,403	110,704
50 000	539,975	1283,81	441,334
100 000	1032,02	2815,51	916,933
500 000	5810,38	14702,4	4734,37
1 000 000	12244,1	29709,3	9946,55



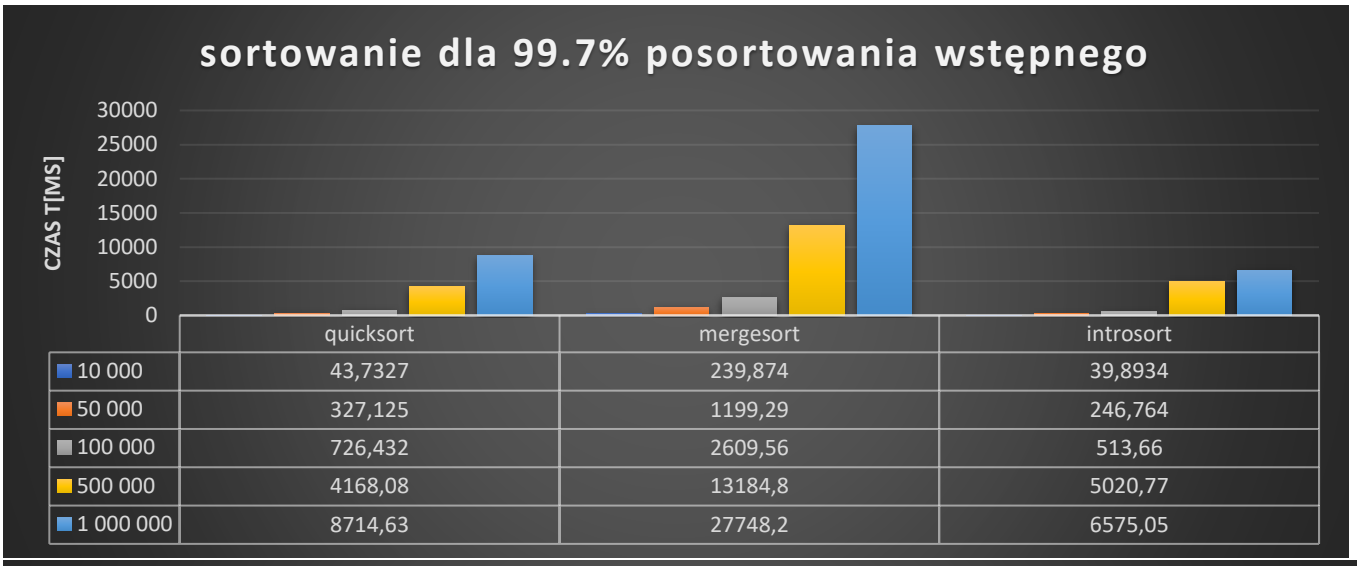
	quicksort	mergesort	introsort
10 000	85,0593	231,134	58,8777
50 000	433,85	1245,74	331,958
100 000	974,972	2714,74	761,467
500 000	5084,98	13521,4	4151,84
1 000 000	10660,3	27715,7	8734,43



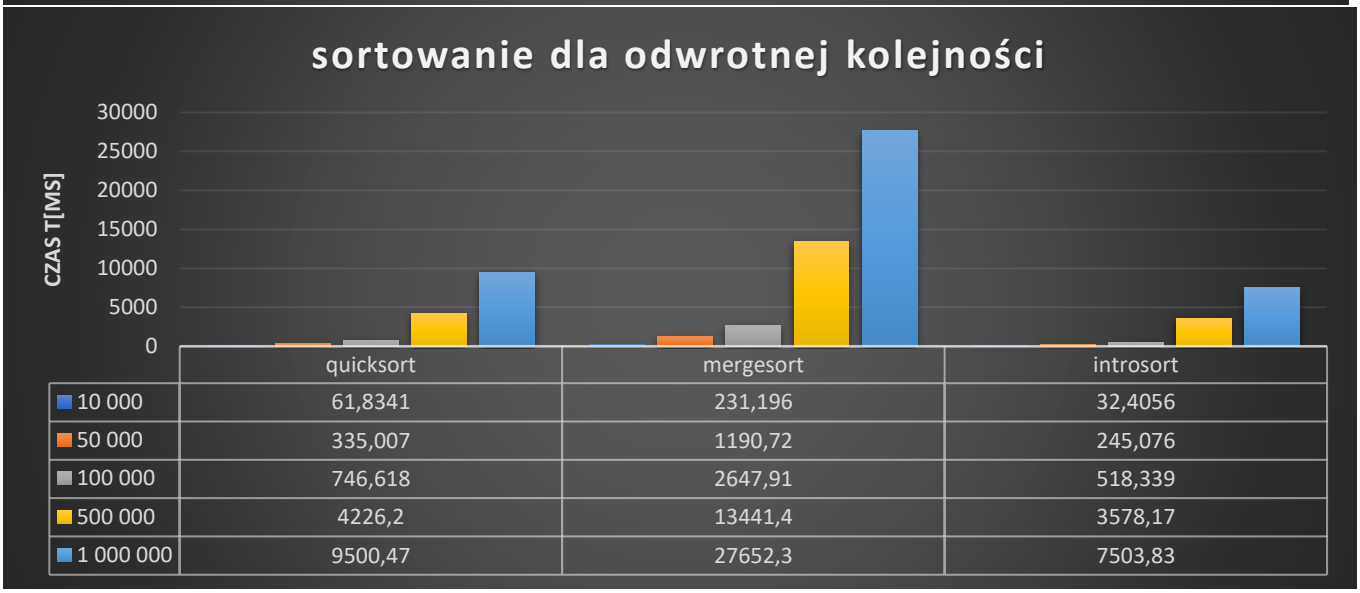
	quicksort	mergesort	introsort
10 000	66,8266	243,323	47,8912
50 000	364,363	1197,94	262,298
100 000	768,773	2541,56	602,36
500 000	4412,12	23562,6	3632,96
1 000 000	9352,95	27781,7	7522,73



	quicksort	mergesort	introsort
10 000	61,8078	217,916	41,8941
50 000	337,327	1178,34	243,822
100 000	746,538	2640,95	559,803
500 000	4302,94	13370,1	3183,65
1 000 000	8872,19	28103,1	6922,24



	quicksort	mergesort	introsort
10 000	43,7327	239,874	39,8934
50 000	327,125	1199,29	246,764
100 000	726,432	2609,56	513,66
500 000	4168,08	13184,8	5020,77
1 000 000	8714,63	27748,2	6575,05



	quicksort	mergesort	introsort
10 000	61,8341	231,196	32,4056
50 000	335,007	1190,72	245,076
100 000	746,618	2647,91	518,339
500 000	4226,2	13441,4	3578,17
1 000 000	9500,47	27652,3	7503,83

3. **Wnioski**

Ćwiczenie polegało na przetestowaniu algorytmów pod względem czasu wykonywania sortowania dla 100 tablic. Wyniki zaprezentowane zostały w postaci wykresów kolumnowych dla każdego stopnia posortowania wstępnego. Zgodnie z przewidywaniami, najgorsze wyniki odnotowano dla sortowania przez scalanie, zaś najlepsze dla sortowania introspektywnego. Sortowanie introspektywne przez naturę algorytmu hybrydowego, łączy w sobie zalety sortowania szybkiego, a zarazem niweluje niepożądane przypadki, w których algorytm sortowania szybkiego okazuje się nie lepszy, od pozostałych algorytmów.

Wstępne sortowanie zauważalnie skraca czas potrzebny do posortowania zestawu danych.

Powyższe wyniki potwierdzają teoretyczne założenia oraz poprawność implementacji algorytmów.

Literatura

[1] Wikipedia Merge sort https://en.wikipedia.org/wiki/Mergesort

[2] Wikipedia Quick sort https://en.wikipedia.org/wiki/Quicksort

[3] Wikipedia Intro sort https://en.wikipedia.org/wiki/Introsort

[4] Geeksforgeeks Merge sort https://www.geeksforgeeks.org/merge-sort/

[5] Geeksforgeeks Quick Sort https://www.geeksforgeeks.org/quick-sort/

[6] Geeksforgeeks Intro Sort https://wwww.geeksforgeeks.org/know-your-sorting-algorithm-set-2-introsort-cs-sorting-weapon