

# Algorytmy sortowania

---

## Wstęp

Projekt opiera się na trzech algorytmach sortowania:

- a) Algorytm sortowania Quicksort
- b) Algorytm sortowania Shell-Sort
- c) Algorytm sortowania Merge Sort

## Algorytm sortowania Quicksort

Algorytm sortowania quicksort opiera się na zasadzie „dziel i zwyciężaj”, zatem jest to algorytm rekurencyjny. Jest uznawany za najszybszy sposób sortowania danych losowych. Idea metody „dziel i zwyciężaj” a zarazem całego algorytmu quicksort opiera się na rozbiciu głównego problemu na mniejsze podproblemy, rozwiązanie ich, a następnie połączeniu tych mniejszych rozwiązań w rozwiązanie problemu głównego.

W schemacie jego implementacji na początku wybierany jest jeden z elementów tablicy, którą chcemy posortować, tzw. pivot. Jego wybór może być zupełnie losowy (w moim projekcie przedstawiłem różnice w działaniu algorytmu quicksort dla różnych wyborów położenia pivota). Następnie po lewej stronie pivota ustawiamy wszystkie elementy naszej tablicy mniejsze (lub równe) od jego wartości a po prawej stronie wszystkie elementy większe (lub równe). Całą tę procedurę nazywamy partycjonowaniem. W następnej kolejności bierzemy dwie, nowo powstałe tablice, od 0 do indeksu pivota-1 i od indeksu pivota do końca tablicy i na nich powtarzamy tę procedurę (jasno widać tu rekurencję).

Złożoność czasowa algorytmu quicksort jest rzędu  $O(n \log n)$ . Pamiętać jednak należy, że w pewnych sytuacjach, zależnych od niekorzystnego ułożenia danych wejściowych lub wyboru pivota, złożoność czasowa tego algorytmu może być rzędu nawet  $O(n^2)$ . Quicksort nie potrzebuje dodatkowej tablicy, tak jak Merge Sort oraz jest łatwy w implementacji.

## Algorytm sortowania Shell-sort

Sortowanie Shella można uznać za pewnego rodzaju rozszerzenie sortowania przez wstawianie. Podstawowym usprawnieniem jest możliwość przestawiania ze sobą odległych od siebie elementów. Na początku zakłada on posortowanie elementów tablicy położonych daleko od siebie, a następnie stopniowo zmniejsza odstęp między sortowanymi elementami.

Zgodnie z algorytmem, sortowany zbiór dzielimy na podzbiory , których elementy odległe są od siebie w początkowym zbiorze o odległość  $h$ . Każdy z utworzonych podzbiorów sortujemy przez wstawianie, a po wykonaniu tego zmniejszamy odstęp. Sortowanie powtarzamy tak długo, aż odstęp osiągnie wartość 1. Na koniec sortujemy zgodnie z algorytmem sortowania przez wstawianie. Ma on jednak ułatwione zadanie, ponieważ zbiór został już w znacznym stopniu posortowany (mniejsze elementy zostały przeniesione w kierunku początku ciągu, a większe w kierunku końca).

Kluczowym elementem wpływającym na efektywność algorytmu Shell-sort jest dobór ciągu odstępów (w swoim projekcie zastosowałem różne metody jego doboru). Ciąg doboru optymalnego ciągu odstępów wciąż nie został rozwiązany matematycznie. Pierwotnie proponowano pierwszy odstęp jako połowę elementów ciągu, a tworzenie kolejnych wartości odstępów jako połowa długości poprzedniego. Okazało się jednak, że nie jest to wybór optymalny (wiele elementów jest branych pod uwagę więcej niż jeden raz), a wręcz jest jednym z najgorszych.

Zakładając najlepszy dotychczas sposób wybierania odstępu, zaproponowany przez Donalda Knutha, złożoność czasowa Shell sorta wynosi  $O(n^{1.15})$ , co czyni Shell sorta bezkonkurencyjnym w klasie sortowań  $O(n^2)$ .

## Algorytm sortowania Merge Sort (sortowanie przez scalanie)

Sortowanie przez scalanie to rekurencyjny algorytm sortowania danych, stosujący metodą „dziel i zwyciężaj”. Ma większą złożoność pamięciową niż podobny do niego, wspomniany wcześniej, algorytm Quicksort – potrzebuje do swego działania dodatkowej pomocniczej struktury danych. Jest szczególnie użyteczny przy danych dostępnych po kolej, jeden element naraz, na przykład w postaci listy jednokierunkowej.

Podstawową operacją algorytmu jest scalanie dwóch zbiorów uporządkowanych w jeden zbiór również uporządkowany. Początkowy zbiór dzielimy na dwa równe (lub prawie równe) podzbiory i stosujemy rekurencyjnie dla każdego z nich sortowanie przez scalanie, chyba że pozostał w nich już tylko jeden element. Posortowane podciągi łączymy w jeden ciąg posortowany. Scalanie jest procesem bardziej złożonym niż dzielenie, które zachodzi bez żadnych warunków. W ramach scalania, dopóki żaden ze scalanych zbiorów nie jest pusty, porównujemy ze sobą pierwsze elementy każdego z nich i w zbiorze tymczasowym umieszczamy mniejszy z elementów usuwając go jednocześnie ze scalanego zbioru. Jeśli jeden ze scalanych podzbiorów będzie pusty, to przekopiujemy wszystkie elementy z drugiego podzbioru (możemy tak zrobić, ponieważ są one już posortowane). Na koniec zawartość zbioru tymczasowego przepisujemy do zbiory wynikowego.

Algorytm ten jest klasy czasowej złożoności obliczeniowej równej  $O(n \log n)$ .

## Część badawcza

Długość generowanych ciągów jest zależna od struktury switch-case, w której kluczowym parametrem jest wartość podawana przez użytkownika (Scanner). Długość ciągów w projekcie, możliwa do wyboru przez użytkownika to 100 tys., 500 tys., 1 mln. lub 2 mln. elementów. Po utworzeniu tablicy o określonej długości przechodzimy do kolejnych operacji, związanych również z podawanymi przez użytkownika parametrami, a zatem ze stopniem początkowego posortowania ciągu oraz algorytmem sortowania jaki chcemy użyć do całkowitego posortowania zadanego ciągu. Tablica jest wypełniana w osobnej metodzie (random\_ciag) liczbami (int) losowymi z zakresu od 0 do 100000. Zakres jest taki żeby zagwarantować powtarzanie się pewnych elementów w trakcie sortowań i obserwacje jak algorytm zachowuje się w takiej sytuacji.

By uzyskać ciąg o określonym początkowym stopniu posortowania bezpośrednio przed wykonaniem faktycznego pomiarowego sortowania, np. dla posortowania pierwszy 50% ciągów wykonywano wpierw niepomiarowego posortowania tej części za pomocą zaimplementowanego już quicksorta.

W trakcie badań istotne jest obciążenie systemu, które wpływa na szybkość działania programu. By tego uniknąć, wszystkie badania zostały wykonane po wcześniejszym wyłączeniu wszystkich innych programów, a tym samym na prawie że identycznym obciążeniu systemu.

W trakcie badań wynikała często (lub prawie zawsze) sytuacja, że pierwszy z wyników znaczco różnił się od pozostałych. Prawdopodobnie wynika to z opóźnień wirtualnej maszyny Javy. Wyniki te pomijane były przy tabelach oraz wykresach związanych z badaniami. Wszystkie tabele oraz wykresy znajdują się w załączonych trzech arkuszach, a wyniki dla poszczególnych sortowań umieszczone są też na końcu tego dokumentu.

W trakcie badań, dla każdej kombinacji długość- stopień posortowania wygenerowano 100 różnych ciągów. Przykładowo weźmy jeden ciąg długości 100 tys. przy początkowym zupełnie losowym układzie elementów. Taki ciąg sortujemy po kolej wszystkimi trzema algorytmami sortowania (w moim projekcie quicksort, shell-sort, merge sort), co daje nam 3 różne wyniki.

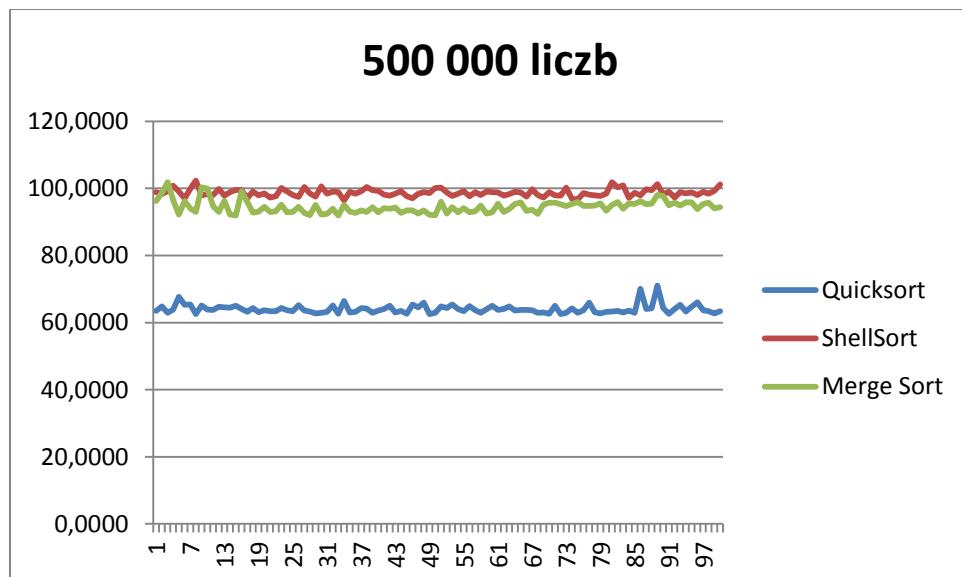
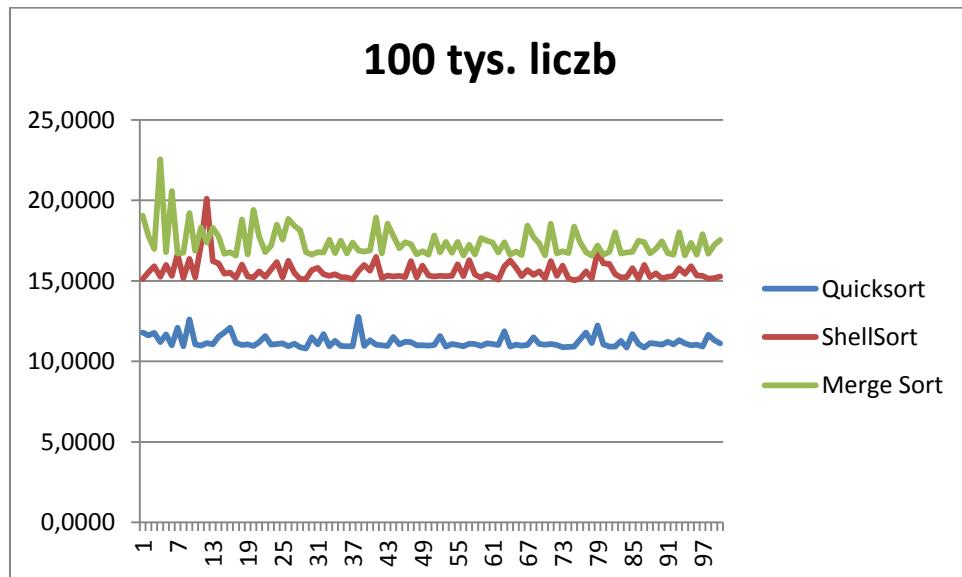
Czas sortowania mierzony jest za pomocą funkcji bibliotecznej System.nanoTime() co zwraca nam czas w milisekundach. Czas pobierano bezpośrednio przed wykonaniem sortowania i bezpośrednio po jego zakończenia. Różnica tych czasów to okres w jakim czasie wykonane zostało sortowanie.

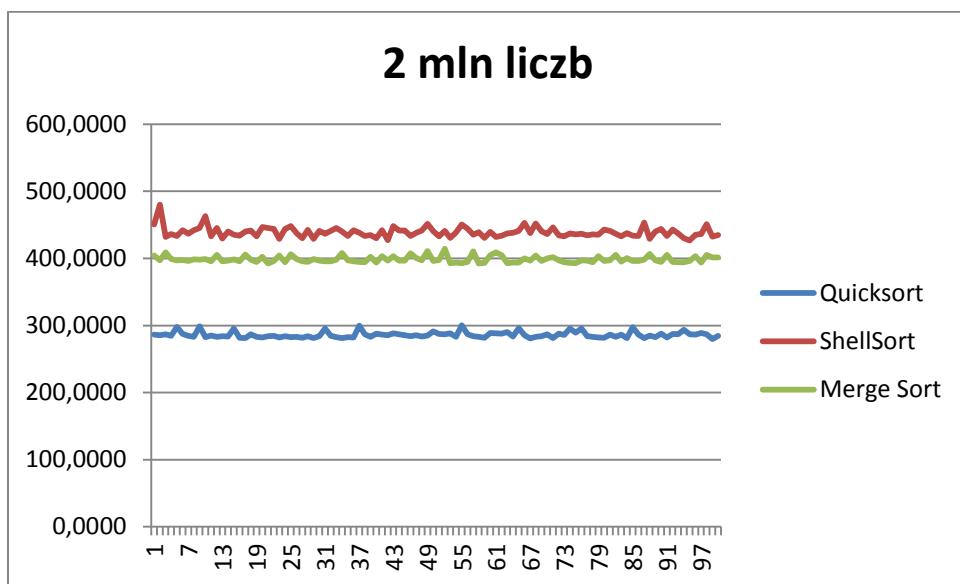
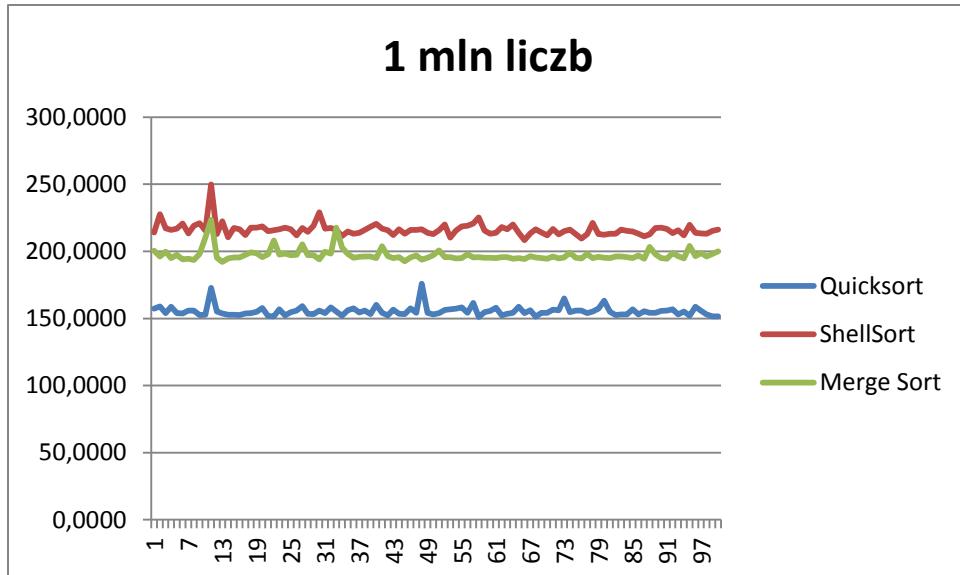
Jako funkcje testową możliwe jest po prostu wypisanie całego, uporządkowanego już ciągu, i obserwacja że ciąg jest posortowany. Można też uruchomić funkcję typu boolean dla danego ciągu sprawdzającą czy każdy wyraz począwszy od drugiego jest większy lub równy od swojego poprzednika.

## Analiza

### Elementy początkowo ułożone losowo

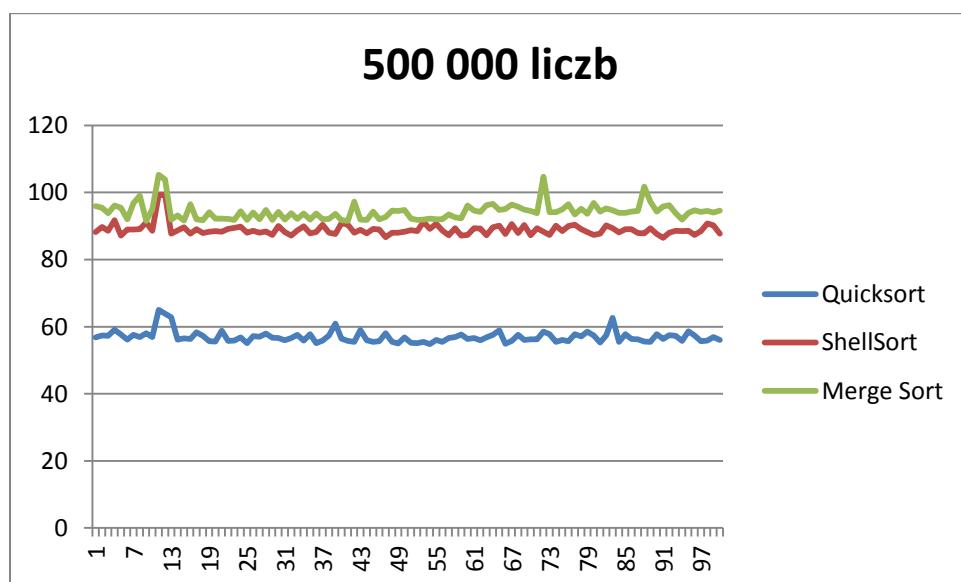
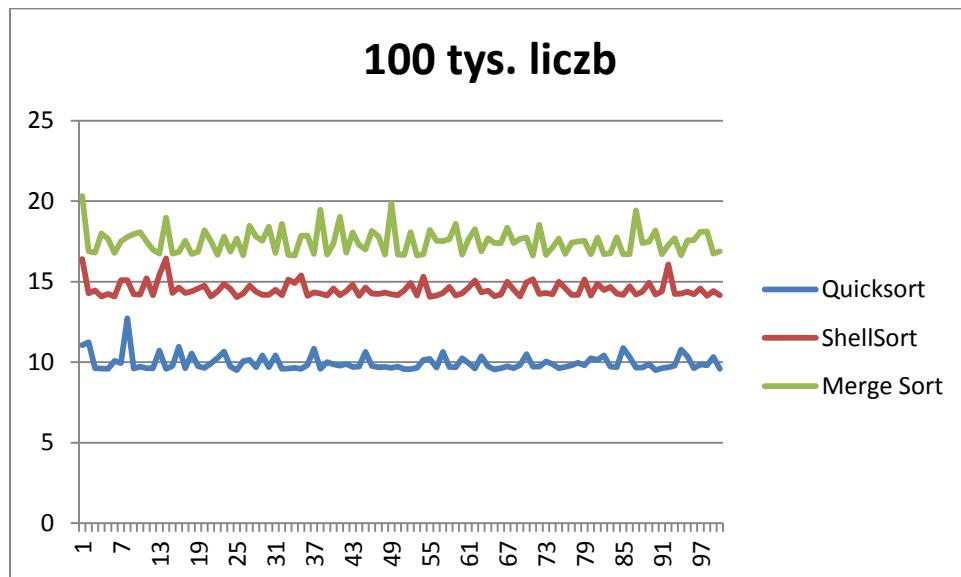
Dla elementów początkowo ułożonych losowo można zauważyc, że czas wykonania algorytmu quicksort jest najszybszy. Długość wykonania Shell-Sorta i Merge Sorta jest dosyć podobna. Wraz ze zwiększającą się liczbą elementów, które chcemy posortować można zauważyc zwiększające się odchylenie standardowe dla wszystkich rodzajów sortowań. Pozostaje ono jednak najmniejsze (wyniki są najbardziej zbliżone do siebie) dla quicksorta, a dla pozostałych dwóch jest podobne. Dla większej liczby elementów czas wykonania Shell-Sorta przekracza czas wykonania MergeSorta, zdecydowanie szybciej rośnie też jego odchylenie standardowe.

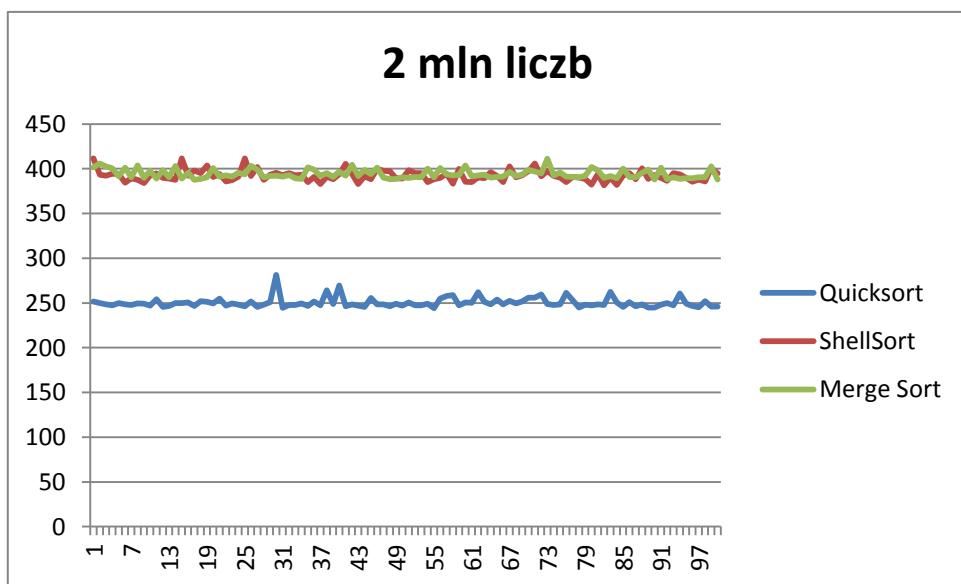
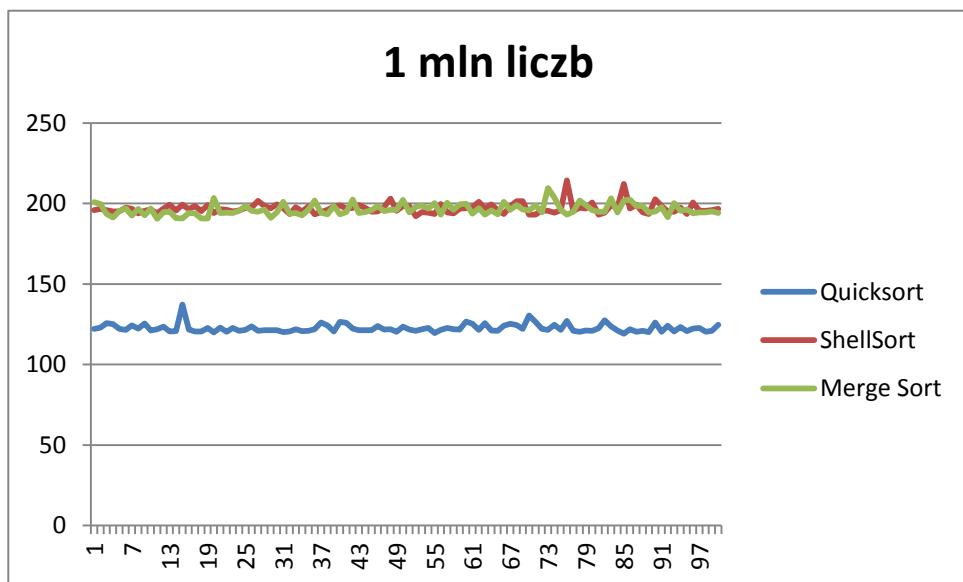




## Elementy początkowo posortowane w 50%

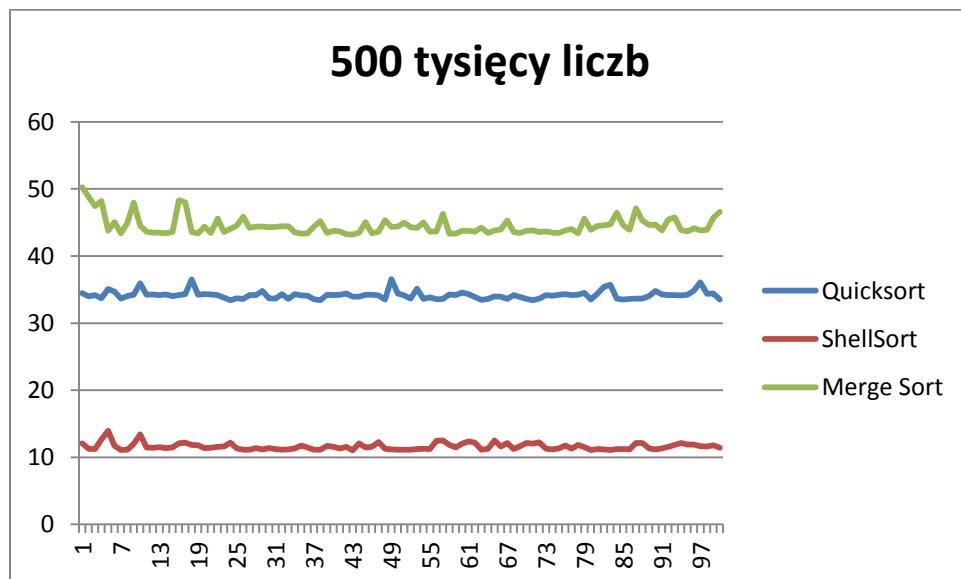
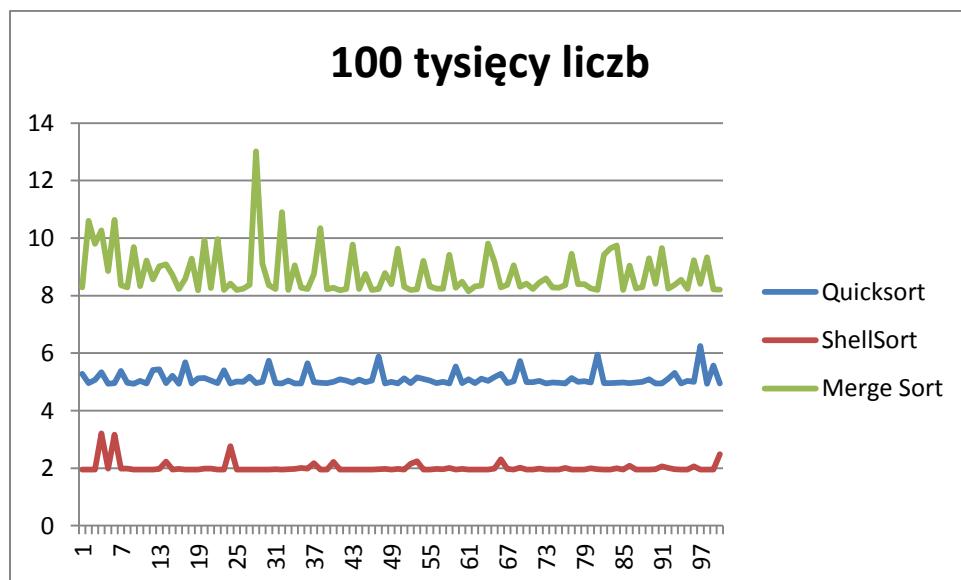
W tym przypadku widzimy zmianę w zachowaniach algorytmów. Widać, że dla quicksorta i Shell Sorta ten typ posortowania danych nieznacznie przyspieszył czas wykonywania algorytmów. Można zauważyć, że czasy potrzebne na wykonanie Merge Sorta są bardzo zbliżone do jego czasów na posortowanie ciągów początkowo zupełnie nieposortowanych. Warto również zwrócić uwagę na odchylenie standardowe. Dla wszystkich trzech algorytmów jest ono zdecydowanie porównywalne, a przy dużej liczbie sortowanych elementów odchylenie standardowe sortowania szybkiego jest nawet większe od odchylenia standardowego sortowania przez scalanie.

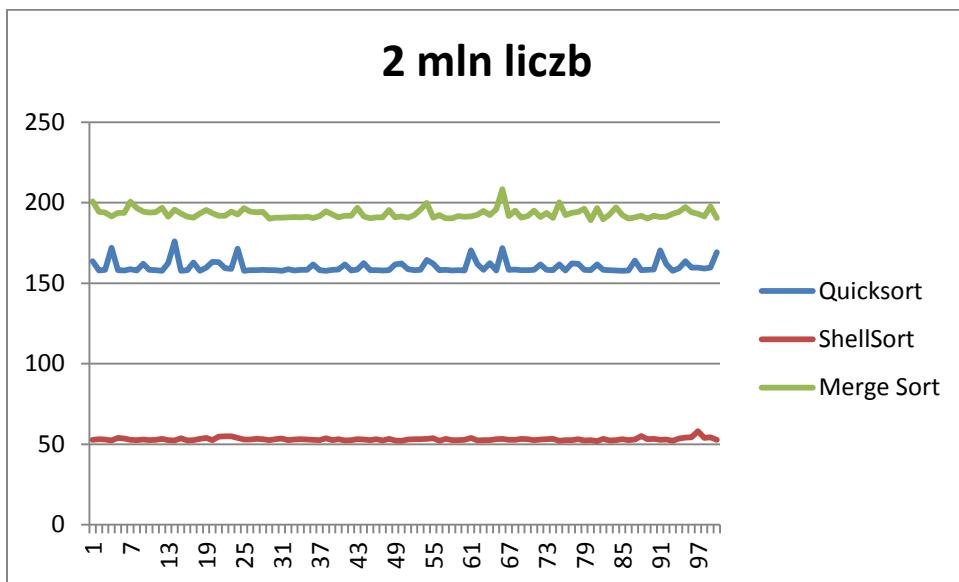
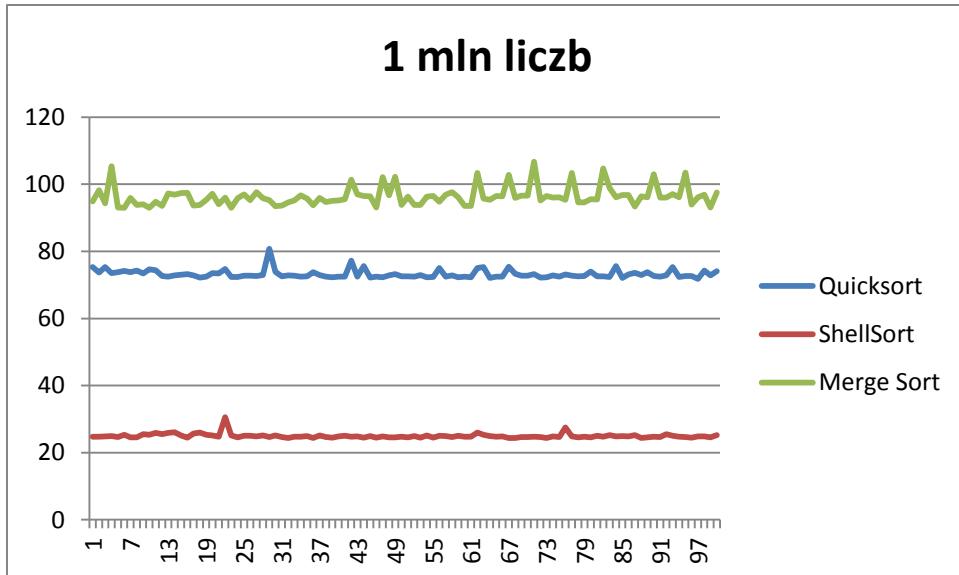




## Elementy początkowo całkowicie posortowane

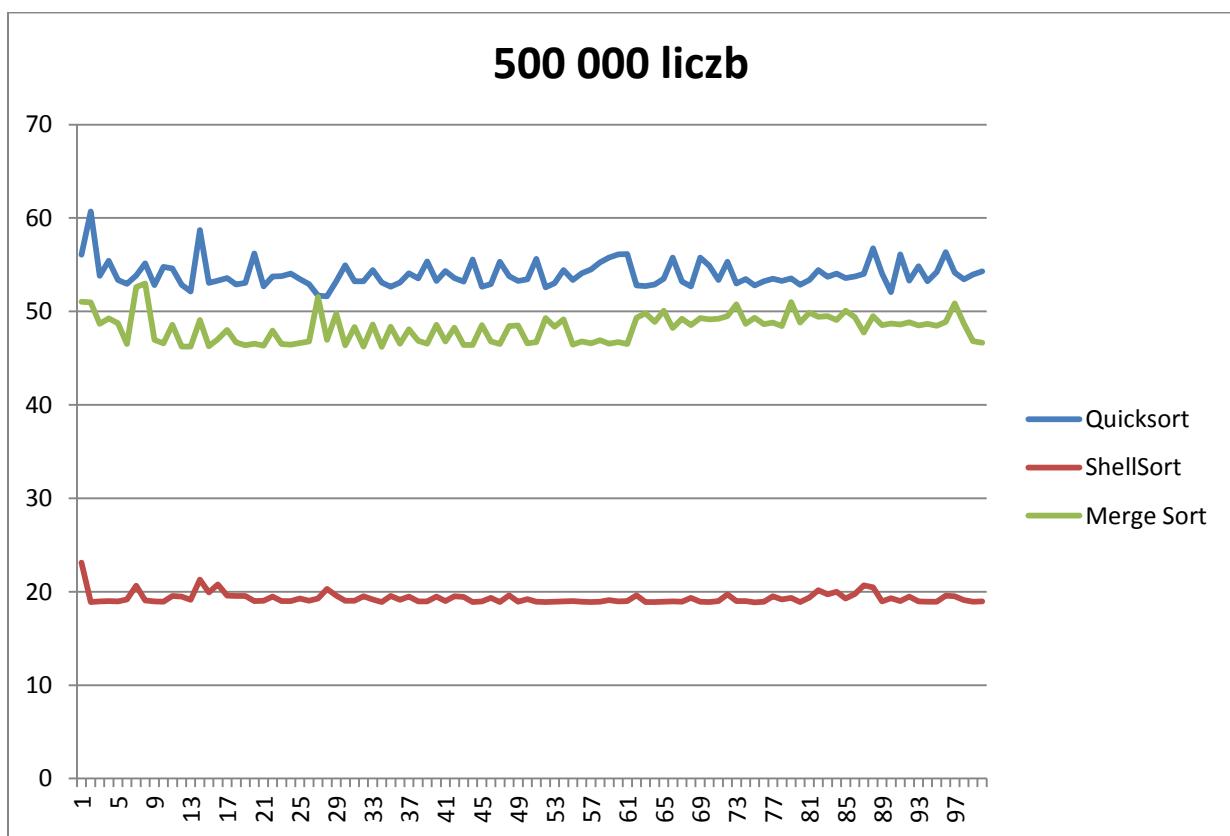
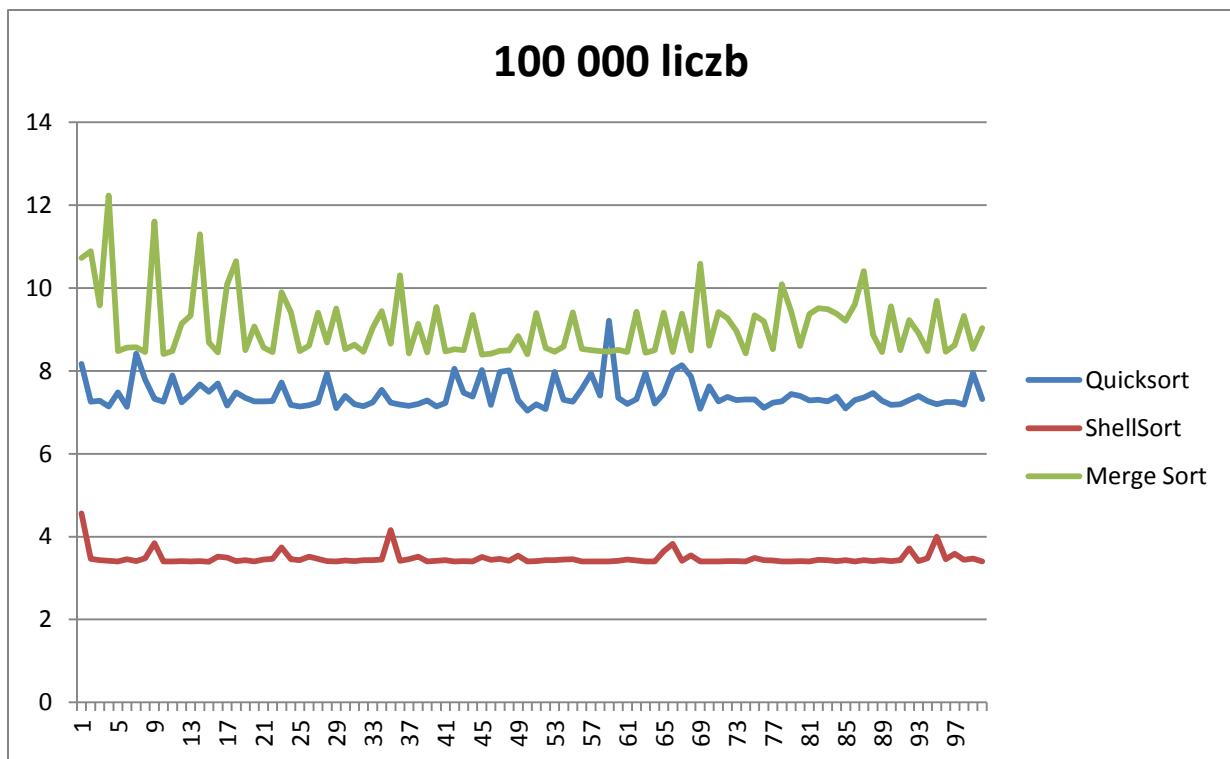
Patrząc po tabelach i wykresach od razu można zauważyc Shell-sort wykorzystuje początkowe posortowanie elementów, w związku z czym cały algorytm jest przeprowadzany szybciej. Dla większej liczby elementów można zauważyc, iż Shell-sort uzyskuje zdecydowaną przewagę czasową nad pozostałymi algorytmami. Odchylenie standardowe dla wszystkich trzech typów sortowań jest bardzo podobne – nieznaczne. Czasy sortowań dla tak uporządkowanych ciągów, nawet dwumilionowych, są rekordowo małe. Nawet większy wśród tych trzech, czas sortowania Merge Sort jest znacznie mniejszy niż jego odpowiednik dla sortowań ciągów początkowo ułożonych losowo lub posortowanych w 50%. Można więc dojść do wniosku, że również dla Merge Sorta początkowe całkowite posortowanie elementów jest wspomaganiem, jednakże nie w tak wielkim stopniu jak dla Shellsorta. Ciekawą obserwacją jest, iż pomimo zwiększającej się liczby elementów odchylenie standardowe wciąż pozostaje w zasadzie minimalne.

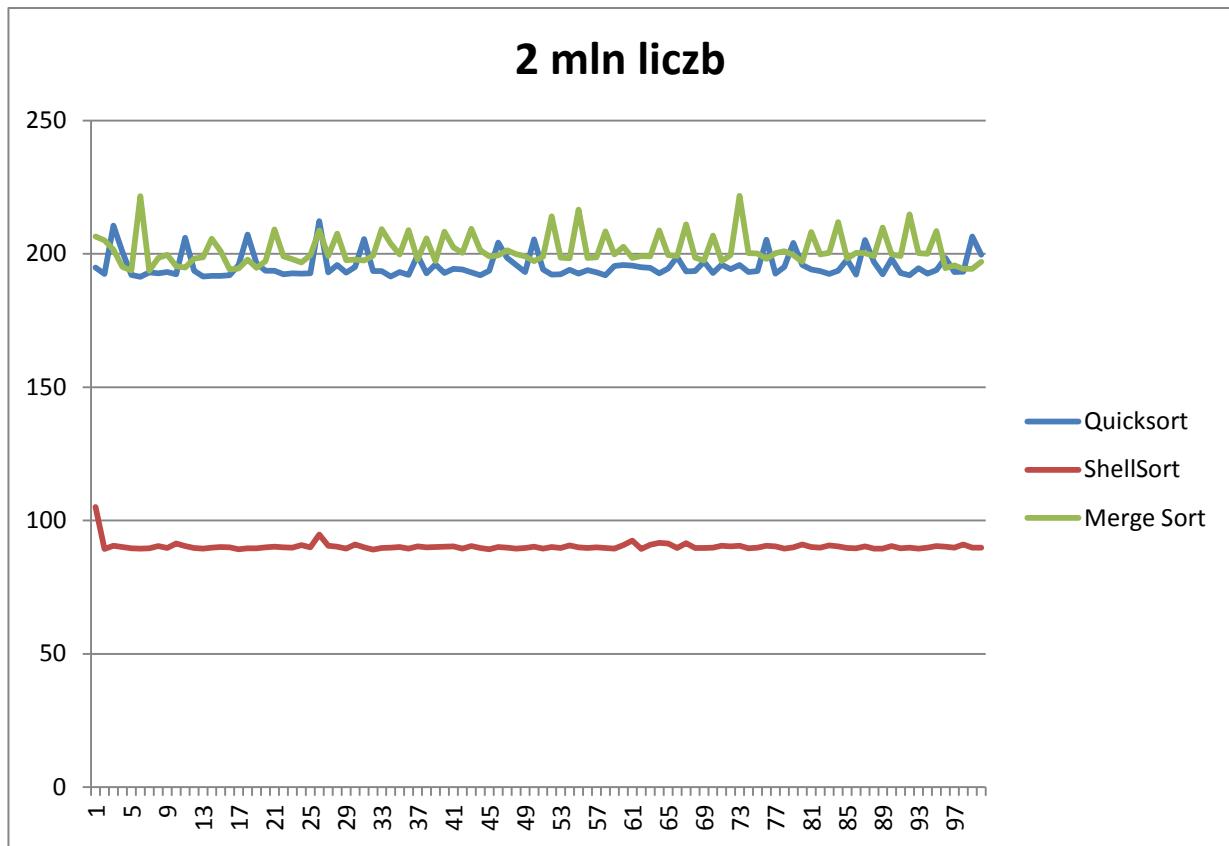
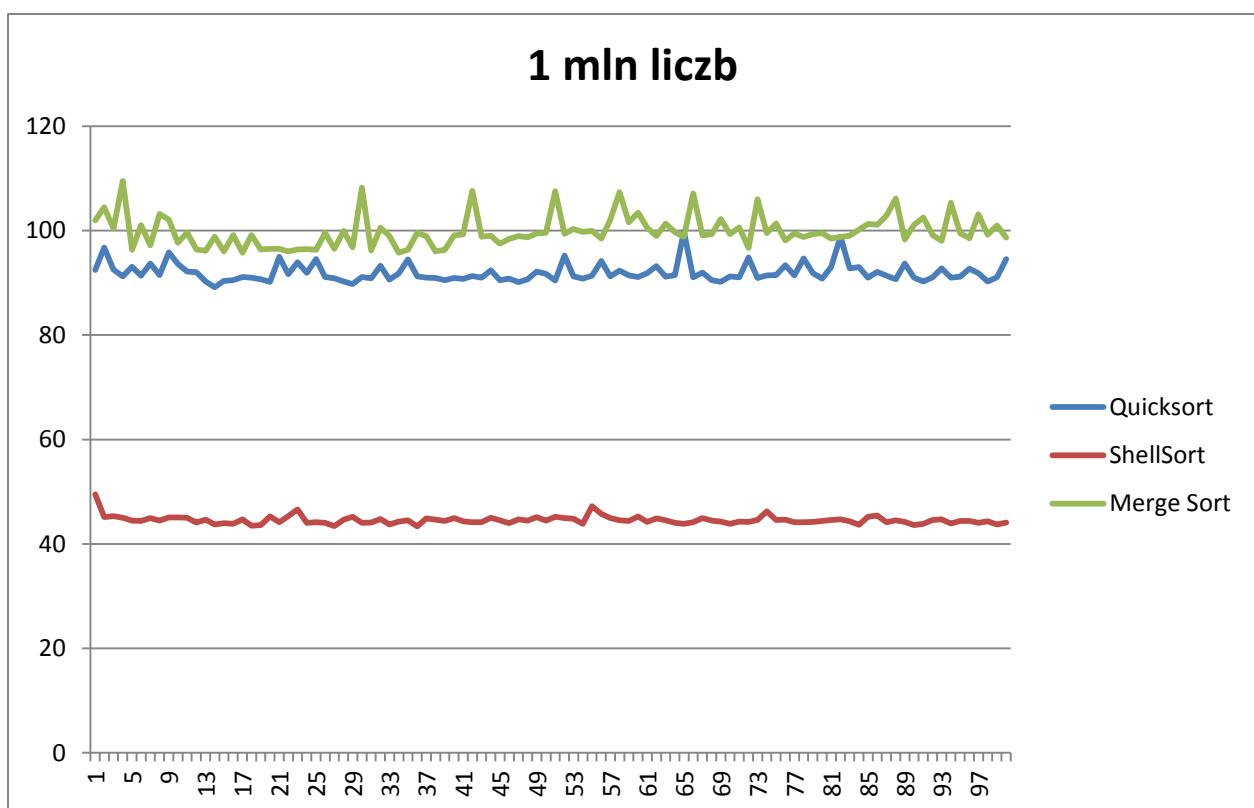




## Elementy początkowo posortowane odwrotnie

Dla elementów posortowanych odwrotnie shell-sort znów działa bardzo szybko. Merge Sort i Quicksort osiągają podobne, trochę gorsze wyniki. Dla większej liczby elementów sytuacja zachowuje się podobnie – Shell Sort wciąż jest najszybszy, a pozostałe algorytmy prezentują podobne rezultaty. Patrząc po wszystkich czasach w tabeli oraz wykresach, można ulec wrażeniu, że dane są bardzo podobne jak przy sortowaniu ciągu początkowo posortowanego całkowicie. Merge Sort uzyskuje wyniki prawie że takie same, a Quick Sort i Shell Sort są w tym wypadku odrobinę wolniejsze. Pomimo tego, czasy te są wciąż jednak zdecydowanie krótsze niż dla sortowania ciągu początkowo ułożonego losowo.





## Shell sort – różne odstępy

Ciekawą kwestią, którą można poruszyć jest dobieranie rozmiarów odstępów przy sortowaniu Shell-Sort, patrząc na wyniki dla trzech różnych odstępów widoczne w załączonym arkuszu, można dojść do wniosku że wielkość odstępu zaproponowana przez twórcę tego algorytmu zdecydowanie nie należy do najlepszych, a wręcz można by powiedzieć, że do najgorszych. Dla sortowania ciągu początkowo całkowicie posortowanego średnie czasy przy innych odstępach niż  $\frac{1}{2}$ , oscylują w okolicach 10 milisekund nawet dla 2 milionów elementów przy bardzo minimalnym odchyleniu standardowym. Odstępy różne niż  $\frac{1}{2}$ , które zastosowałem w projekcie to  $\frac{1}{4}$  i  $1/7$ . Wszystkie tabele dla różnych odstępów znajdują się w załączonym arkuszu Shell.xlsx na 4 kolejnych podarkuszach.

## Quicksort – różny wybór pivota

Patrząc po wynikach dla różnego sposobu wyboru pivota, za pomocą metody qs\_pivoty() można zauważać, że złożoność algorytmu zależy od wyboru właśnie elementu rozdzielającego. Jednym ze sposobów wyboru pivota w moim projekcie był losowy wybór tego elementu. Jeśli podziały są zrównoważone, to algorytm jest tak szybki jak Merge Sort  $O(n \log n)$ . W przeciwnym wypadku (dla złego ułożenia elementów) może działać nawet w czasie  $O(n^2)$ . Jednak na podstawie wyników załączonych w tabelach w arkuszu Qs.xlsx można zauważać, że średni czas wykonania sortowania przy losowym wyborze elementu rozdzielającego dorównuje przypadkowi optymistycznemu. Wszystkie tabele dla różnych pivotów znajdują się w tym pliku na kolejnych czterech podarkuszach.

## Wnioski

### Podsumowanie i identyfikacja możliwych błędów

Algorytm Shell-sort oraz Quicksort należą do algorytmów niestabilnych. Dla sortowania szybkiego średnia złożoność to  $O(n \log n)$ , jednakże przy niefortunnym ułożeniu elementu sięga ona nawet rzędu  $O(n^2)$ . Z kolei dla sortowania Shella złożoność jest nieznana, nie jesteśmy w stanie określić jaki odstęp pomiędzy kolejnymi wybieranymi elementami byłby optymalny. Sortowanie przez scalanie należy do stabilnych algorytmów sortowania, jest złożoności  $O(n \log n)$ , jednakże do jego wykonania potrzebuje  $O(n)$  dodatkowej pamięci,

czym różni się on w porównaniu do pozostałych algorytmów zamieszczonych w tym projekcie. Sortowanie Merge Sort oraz Quicksort opierają się na metodzie „dziel i zwyciężaj”.

W części badawczej przeprowadzane są sortowania ciągów liczbowych o pewnych własnościach. Badania przeprowadzono dla ciągów o długości 100 tys., 500 tys., 1mln i 2 mln elementów.

Bez trudu można zauważać, że pewne, nielosowe, początkowo posortowanie ciągu jest zdecydowanie wspomaganiem dla Shell-Sorta – korzysta on z takiego ułożenia elementów. Dla Quicksorta jest to nawet pewne utrudnienie. Z kolei Merge Sort osiąga praktycznie takie same wyniki dla początkowo ułożonych losowo elementów i elementów posortowanych w 50%, oraz takie same wyniki dla elementów posortowanych całkowicie i odwrotnie (są one mniej więcej połową czasów potrzebnych na wykonanie Merge Sorta dla elementów ułożonych losowo lub posortowanych w 50%). Przy początkowym posortowaniu elementów odchylenie standardowe Shell-Sorta jest zazwyczaj najwyższe. Jednakże dla losowego ułożenia elementów (z którym możemy chyba spotkać się najczęściej) Quicksort radzi sobie zdecydowanie najlepiej, przy bardzo małym odchyleniu standardowym. Prawdopodobnie większą różnicę w czasach bylibyśmy w stanie zauważać dla większej liczby elementów, które chcemy posortować. W praktycznie każdym przypadku sortowania pierwszy wynik podany przez algorytmy był odrzucany, ponieważ jego wartość znaczco przekraczała pozostałe wyniki. Wynika to najprawdopodobniej z opóźnień wynikających z inicjalizacji maszyny wirtualnej Javy. Wszystkie badania zostały przeprowadzone na jednakowym obciążeniu systemów (przy wyłączonych wszelkich innych programach). Dokładne wyniki wszystkich pomiarów wraz ze średnimi i odchyleniami standardowymi zamieszczone są na końcu dokumentu oraz są też do wglądu w załączonym arkuszu danych (na 4 kolejnych arkuszach zaprezentowane są pomiary dla 4 kolejnych typów początkowego posortowania ciągów).

Zaobserwowana złożoność obliczeniowa algorytmów z reguły pokrywa się z teoretyczną. W przypadku Quicksorta widzimy, że w większości przypadków złożoność wynosi  $O(n \log n)$ , a w pesymistycznym przypadku, dla niekomfortowego dla niego ułożenia danych, nawet  $O(n^2)$ . Zaobserwowana złożoność czasowa Shell-Sorta jest momentami bardzo różna, lecz wynika to z powodu, że w większości przypadków wykonujemy Shell-Sorta na elementach uporządkowanych, co znacznie go usprawnia. Co więcej, nie jesteśmy w stanie określić teoretycznej złożoności obliczeniowej sortowania Shella. Złożoność czasowa Merge Sorta pokrywa się z teoretyczną.

randomy	100tys.			500tys.			1mln.			2mln.				
	Quicksort	ShellSort	Merge Sort											
11,7980	15,1053	19,0522		63,4681	98,8842	96,1827		157,1864	214,0057	200,4752		286,6179	450,4843	404,0783
11,6184	15,5373	17,7981		64,8220	98,2623	98,6750		159,0337	227,7178	196,2148		285,4772	480,0458	397,1697
11,7780	15,9088	16,9822		62,9292	99,1687	101,8226		153,9796	217,1069	199,6771		286,6959	431,8247	408,8754
11,2072	15,2463	22,5421		63,8404	100,8321	96,5697		158,7903	215,8527	195,0852		284,7483	436,3971	398,9640
11,6761	15,9921	16,7762		67,7070	99,0297	92,1595		153,8573	216,8497	197,4619		297,9774	433,5564	397,0990
10,9927	15,3159	20,5713		65,2730	96,9600	96,1813		153,7675	220,6404	193,9872		287,5406	441,8204	397,7214
12,0945	16,6686	16,7060		65,3882	99,9160	93,9859		155,7469	213,3996	194,5710		284,4261	436,5262	396,1485
10,9407	15,1811	16,7408		62,5451	102,2765	93,0015		155,9618	219,0909	193,5202		283,1234	442,0833	398,4243
12,6107	16,3569	19,2048		65,1475	97,9305	100,2786		152,6213	220,9877	198,0228		298,9378	445,4939	398,2114
11,0586	15,2131	16,8731		63,8240	98,3409	99,8920		152,7656	215,4847	210,3478		282,6717	462,9296	398,9933
10,9847	17,2344	18,3066		63,7916	98,0100	94,6740		172,8119	249,7504	223,6978		285,2136	433,0505	395,8523
11,1314	20,1079	17,3714		64,7536	99,8589	92,9530		155,2419	212,8666	195,1271		283,0079	445,0961	405,3324
11,0572	16,2183	18,2999		64,4931	97,7901	96,3280		153,6038	222,5153	192,2270		284,0618	429,5338	395,4385
11,5309	16,0834	17,7540		64,4246	98,9281	92,1737		152,7964	210,4554	194,6527		283,3605	439,8409	396,4800
11,7894	15,4583	16,6820		65,0033	99,5939	91,9375		152,6797	217,3263	195,5405		295,3515	435,3841	398,6959
12,1001	15,5030	16,7774		64,0578	99,4133	99,1439		152,5396	216,3636	195,4085		281,5813	433,6933	395,8722
11,1633	15,1957	16,5747		63,2208	97,0816	95,9695		153,8096	212,0716	197,4667		281,3696	440,0282	405,8844
11,0135	16,0039	18,8062		64,3585	99,0927	92,7712		153,8626	217,7001	199,3686		286,9191	441,6794	397,9344
11,0636	15,2700	16,6369		63,1421	97,9637	93,0487		154,9692	217,5762	198,5533		283,2282	432,8839	394,8149
10,9667	15,2108	19,4033		63,6535	98,5225	94,4604		157,7584	218,6864	195,7270		282,3660	446,6823	402,5124
11,1961	15,5866	17,7425		63,3903	97,2242	92,9730		151,7359	214,8990	197,8014		284,1349	445,3740	392,4136
11,5845	15,2475	16,7840		63,3558	97,6246	93,1224		151,6414	215,7456	208,0221		284,6983	444,0319	396,0682
11,0322	15,7177	17,1826		64,3380	100,1201	95,1754		156,8999	216,3435	197,5440		282,1356	428,8453	404,2200
11,0833	16,1687	18,4915		63,6780	99,0728	92,8918		152,3905	217,7232	198,2577		284,2596	443,9760	394,0305
11,1155	15,2226	17,5505		63,4327	98,0227	92,9448		154,5546	216,4981	197,0028		282,7881	448,2657	406,1995
10,9377	16,2389	18,8590		65,2485	97,4217	94,4627		155,9156	212,0147	197,3458		283,2953	437,8054	399,1297
11,0988	15,4985	18,4175		63,5971	100,4192	92,7181		159,1515	217,4421	205,1219		281,7943	430,1490	395,5529
10,8764	15,1214	18,1653		63,3281	98,4862	91,9661		153,4624	214,4924	197,0546		283,8864	442,2321	394,9307
10,8085	15,1059	16,7846		62,7362	97,5029	95,0240		153,2243	218,9800	197,0032		281,4036	429,2667	398,8275
11,4913	15,6729	16,6209		62,9418	100,5961	92,1720		155,8218	229,1802	194,0699		284,0568	440,8329	396,6813
11,0585	15,8195	16,7863		63,1915	98,4065	92,3640		153,9673	217,0198	199,8146		295,7137	436,7639	395,6273
11,7005	15,4119	16,7710		65,0899	99,0702	93,9021		158,2861	217,3180	198,4079		284,5515	441,2040	395,5269
10,9389	15,3091	17,5498		62,6518	98,8691	91,8776		155,1233	215,9471	217,7188		282,7038	445,0433	397,8778
11,2717	15,4220	16,7168		66,3967	96,3879	95,0585		151,9516	211,4339	202,7430		281,3508	439,7948	408,3018
10,9622	15,2243	17,5027		63,0268	98,9300	93,0675		156,1011	214,7162	198,0793		282,8446	433,4930	397,0837
10,9459	15,2055	16,7134		63,1885	98,4172	92,6770		157,5091	212,9891	195,1016		282,2892	441,9555	395,6933
10,9434	15,1027	17,3815		64,3838	99,1406	93,4384		154,3933	213,7949	195,9356		299,7028	438,1361	394,5154
12,7710	15,6024	16,8897		64,1978	100,4445	92,9818		155,7373	215,9539	196,1452		286,6788	433,2661	394,0621
10,9606	15,9889	16,8273		62,9416	99,5009	94,4316		153,1813	218,3912	196,2808		282,8555	434,7116	402,5431
11,3232	15,6226	16,8773		63,5522	99,2504	92,8452		160,0563	220,4714	194,9152		287,6931	429,9922	393,5620
11,0346	16,4818	18,9336		64,0810	98,1002	94,0704		154,2268	217,0080	203,6764		286,6427	441,9704	403,0607
11,0035	15,1733	16,7070		65,0373	97,8046	93,9454		152,2639	215,7225	196,4865		285,3484	427,2803	396,5624
10,9633	15,3400	18,5590		63,0487	98,4020	94,2922		156,5911	212,1111	194,8679		288,4560	448,1526	403,4362
11,5255	15,2662	17,7985		63,5283	99,1629	92,6349		153,4730	216,4218	195,7477		287,1139	441,6249	396,4514
11,0532	15,3113	17,0182		62,6722	97,6108	93,4517		153,2948	213,1909	192,4800		285,6638	441,6785	396,5222
11,2109	15,2402	17,4078		65,3854	97,0360	93,4531		157,4842	216,0071	195,4308		283,9023	433,3062	407,3754
11,2036	16,2356	17,2778		64,5332	98,4137	92,4895		154,1829	216,0759	196,8642		285,6024	438,3460	400,5880
10,9973	15,2093	16,6374		65,9602	98,8516	93,4036		175,9222	216,3332	193,7974		283,7749	441,4281	397,1103
11,0096	15,9565	16,8339		62,5331	98,5628	92,1124		154,2306	213,7546	195,2754		284,9034	451,5047	411,0227
10,9832	15,3274	16,6227		62,9483	100,0787	92,0039		153,0408	212,9504	197,1064		291,2861	440,4006	396,2903
11,0108	15,2764	17,8117		64,7789	100,2066	95,9750		154,0406	215,7231	200,6281		287,2722	432,8438	397,6937

## Marcin Nahajowski 246711

Czwartek 15:15

	11,5699	15,3192	16,7605		64,3747	98,7740	92,4846		156,3738	220,1063	195,6451		286,7791	440,8247	414,1228
	10,9284	15,2841	17,4193		65,4256	97,7364	94,4405		156,8518	210,3463	195,5705		288,2832	430,4026	392,6833
	11,0766	15,3168	16,7474		64,0981	98,4394	92,8274		157,2097	215,5648	194,7537		283,1004	438,3689	393,5851
	11,0120	16,0208	17,4207		63,3928	99,1283	94,0735		158,1328	218,5991	194,9037		300,1635	450,4293	392,7951
	10,9466	15,2882	16,5814		64,9395	97,6384	92,9140		154,1355	219,1628	197,7229		287,0890	444,3871	394,5887
	11,0942	16,2905	17,2393		63,6972	98,9195	93,1192		161,4785	220,6960	195,4741		284,1540	435,2598	410,6881
	11,0786	15,3999	16,6356		62,8675	97,9838	94,8646		151,0556	225,3598	195,5994		282,8839	438,9227	392,3745
	10,9669	15,1856	17,6517		63,9867	99,1080	92,4422		154,7764	215,3856	195,2826		281,7943	430,4769	393,1893
	11,1130	15,4199	17,5094		65,0079	98,8702	92,8681		155,7043	213,1049	195,1459		288,8745	439,6355	404,4987
	11,0842	15,2447	17,3737		63,7353	98,7799	95,4534		157,9941	213,7788	194,9665		288,5392	431,7438	409,2429
	11,0151	15,0809	16,7650		64,0478	97,8041	92,9215		152,1645	218,0950	195,6240		287,7739	433,9934	404,9417
	11,8761	15,9132	17,4076		64,8406	98,3480	93,8105		153,3602	216,5081	195,6576		290,2773	436,9682	392,9136
	10,9248	16,2607	16,6285		63,5632	98,9750	95,3382		154,1055	220,0523	194,5506		283,4998	437,8923	393,8510
	11,0474	15,8202	16,8178		63,7727	98,7388	95,8238		158,6262	213,8322	195,0167		295,7368	441,0727	393,6382
	10,9810	15,2892	16,5954		63,7850	97,5031	93,2513		153,8408	208,3783	194,3259		285,4225	453,1285	399,9397
	11,0239	15,6676	18,4307		63,6299	99,7062	93,6505		156,0773	213,3728	196,5206		280,8196	437,5367	396,6570
	11,5037	15,3637	17,7113		62,8720	97,9988	92,4067		151,3568	216,4469	195,5283		283,1672	452,1148	404,1072
	11,0710	15,5899	17,3194		63,0077	97,2303	95,1508		154,2107	214,0097	195,0184		284,2029	440,2808	395,9283
	11,0342	15,1582	16,5944		62,6604	98,8438	95,7738		154,2893	211,8599	194,4086		287,0554	436,1991	400,1647
	11,0770	16,2182	18,5578		65,0007	97,9640	95,7142		156,5211	216,7637	196,1447		281,1163	446,0953	402,0974
	11,0275	15,3203	16,7135		62,5359	97,7885	95,2356		156,1965	212,7240	194,6593		288,0958	434,1859	396,9179
	10,8905	15,9504	16,8368		62,8667	100,1806	94,7059		164,9553	214,8927	195,4134		285,7522	432,9491	394,2639
	10,8957	15,1540	16,7211		64,2628	96,8460	95,3239		154,7438	216,1057	199,1081		295,2842	436,9595	393,4415
	10,9128	15,0335	18,3671		62,9000	96,6684	95,9428		155,9589	212,8570	195,1333		289,2771	435,8381	392,6808
	11,3676	15,1437	17,4220		63,7212	98,6424	94,7404		155,7440	209,4173	194,6907		295,5204	436,6257	397,0037
	11,7926	15,5835	16,7872		65,9660	98,1370	94,8065		154,0604	212,8581	198,1093		283,8946	434,2348	396,6015
	11,1379	15,1455	16,5606		63,0932	97,9713	94,8551		155,2434	221,2541	194,9930		283,1184	435,6479	394,2637
	12,2390	16,8192	17,1959		62,7053	97,7199	95,5625		157,2984	212,9682	195,8487		281,9776	435,4801	403,3767
	11,0593	16,0850	16,6308		63,1833	98,4462	93,3657		163,2546	212,4326	195,2212		281,8471	443,1145	396,3465
	10,9150	16,0543	16,8042		63,3267	101,8491	95,0366		154,9153	213,2040	195,0263		286,5685	440,8390	397,2445
	10,9237	15,4086	18,0093		63,4576	100,3213	95,8907		152,5512	213,0937	196,1348		283,2916	436,6907	405,0193
	11,2777	15,2235	16,7003		62,9686	100,8423	93,8995		152,9814	216,1626	196,1494		286,4374	432,7126	395,3363
	10,8654	15,2308	16,7599		63,6055	97,1699	95,4987		153,0846	215,1714	195,7433		281,1582	437,6306	400,2702
	11,6902	15,7898	16,8089		62,9393	98,6641	95,3564		156,7745	214,8330	194,8969		297,4575	433,7394	396,2176
	11,0787	15,1175	17,5034		70,0770	97,9253	96,1708		152,8910	213,0842	196,7812		286,2864	433,4177	395,9811
	10,8643	15,9814	17,4069		64,0471	99,7708	95,2996		155,3269	211,1692	194,5557		281,1822	453,5915	398,2620
	11,1331	15,2310	16,7007		64,2116	99,4435	95,4651		154,0897	212,5095	203,3126		285,0773	429,1493	406,6455
	11,0956	15,4750	16,9734		71,0124	101,2583	98,0130		154,2653	217,3970	197,9418		282,7001	440,1435	396,8600
	11,0374	15,1603	17,4604		64,3595	98,2326	97,7177		155,6209	217,7077	194,9940		287,6500	443,7041	394,6998
	11,2095	15,2545	16,7328		62,5889	99,0080	94,9665		155,8884	216,6765	194,5776		281,9245	433,2184	404,9992
	11,0525	15,2924	16,6181		64,0976	97,2241	95,7586		156,7088	213,5287	198,7148		287,2457	442,8219	394,7433
	11,3102	15,7818	18,0131		65,2662	98,8979	94,9008		152,8903	215,7882	196,3264		286,8519	437,2945	394,0537
	11,1113	15,4316	16,5950		63,2947	98,5473	95,8636		155,1210	211,9402	194,7329		293,7954	429,8596	394,4091
	11,0030	15,9001	17,3521		64,7186	98,8201	95,8732		152,0092	219,7817	204,0425		286,8076	426,8715	396,0014
	11,0337	15,3427	16,6186		66,0289	97,9916	93,8553		158,6130	213,9147	196,3484		286,3096	435,3646	403,2428
	10,9260	15,3132	17,8920		63,7241	98,9665	95,3437		155,5042	213,2207	198,7468		288,8783	436,3339	393,5244
	11,6597	15,1398	16,6897		63,3890	98,4159	95,7453		153,0055	212,9995	196,0803		287,0370	451,0791	405,4147
	11,3095	15,1745	17,2393		62,7599	99,2773	94,0271		151,5581	215,3619	198,0271		279,5700	432,4786	401,3461
	11,1103	15,2775	17,5377		63,3681	101,1764	94,3502		151,5909	216,1639	199,8812		284,4586	434,6128	401,4908
srednia	11,2103	15,5987	17,3790		64,0420	98,7169	94,4903		155,4475	216,1374	197,2791		286,0967	439,0503	398,6613
odchylenie standardowe	0,371742	0,633218	0,936995		1,360887	1,1276791	1,8889754		3,6894152	4,8409352	4,4923925		4,5133382	7,7575015	4,9009872

Marcin Nahajowski 246711  
Czwartek 15:15

50%	100tys.			500 tys.			1mln			2mln		
	Quicksort	ShellSort	Merge Sort									
11,06715	16,41043	20,31621		56,77256	88,19078	95,952301	122,09252	195,86892	200,77077	251,7853	411,57697	401,4582
11,23882	14,27266	16,87939		57,41049	89,74459	95,439149	122,97586	196,58427	199,56159	249,80434	393,51887	405,79834
9,632222	14,46329	16,81054		57,29769	88,57556	93,857399	125,76498	195,89373	193,50013	248,37878	392,10916	402,47073
9,60918	14,08622	17,98797		59,11291	91,72136	96,078833	125,14808	195,05833	191,36242	247,38136	394,54872	400,52655
9,588731	14,24121	17,66417		57,70504	87,10798	95,494274	122,19797	195,05386	195,32065	249,74342	394,11055	391,45092
10,08249	14,07954	16,77448		56,1677	88,98753	91,975226	121,50852	197,40493	196,98988	248,48375	384,47718	401,3523
9,942873	15,09188	17,50449		57,59777	88,92514	96,779954	124,3588	196,62805	192,46093	247,6176	389,41293	390,45192
12,72993	15,08869	17,78245		56,88833	89,04753	99,094151	122,25337	193,89319	196,58447	249,47751	387,76065	403,68092
9,601768	14,21925	17,96674		58,05046	91,07066	91,543283	125,47143	195,40609	192,69014	249,26774	384,20096	389,92379
9,724276	14,20351	18,06807		56,88389	88,5314	94,971566	121,21132	196,18024	196,91643	247,05315	392,63823	397,5125
9,626927	15,22251	17,49837		64,97813	99,21815	105,30856	121,90417	193,85372	190,58484	254,32712	394,60215	388,89495
9,625765	14,16586	16,97905		63,82089	99,27598	103,84685	123,44124	196,86193	194,75309	245,63672	389,75287	398,39291
10,71932	15,40354	16,75005		62,84929	87,71327	91,939264	120,47454	199,20138	194,85532	246,70006	389,18896	390,32655
9,586112	16,43889	18,97066		56,15665	88,68995	93,163712	120,80922	195,74989	190,89248	249,79743	387,65136	403,16227
9,766047	14,29719	16,74219		56,56716	89,58879	91,627065	137,13756	199,2903	190,62216	249,86253	411,59914	389,0879
10,96313	14,64004	16,8401		56,37456	87,73504	96,474634	121,6324	196,38134	194,15438	250,76905	392,25269	395,60552
9,622528	14,29349	17,5488		58,32557	89,01707	92,018629	120,50439	198,47929	193,90892	246,6729	398,16793	387,48258
10,5485	14,3927	16,71652		57,24123	87,92069	91,750831	120,61974	195,20305	190,74005	251,86812	395,21848	388,68288
9,739914	14,58171	16,86362		55,69024	88,25012	94,117284	122,68857	199,09381	190,64563	251,37606	403,62803	390,36141
9,658082	14,75669	18,19713		55,55003	88,46957	92,201334	119,92229	194,01278	203,34816	249,4226	390,74116	400,78401
9,932994	14,08938	17,52436		58,79065	88,24682	92,198929	122,88056	196,40163	193,98532	255,04163	394,03979	391,03205
10,27133	14,39312	16,65896		55,80258	89,13229	92,143213	120,33202	196,01756	194,22725	246,85805	385,91516	392,66473
10,66957	14,86413	17,79289		55,87922	89,45395	91,809751	122,64266	195,14448	193,98337	249,54428	387,14933	391,3209
9,72701	14,54989	16,86058		56,83593	89,8019	94,421884	120,91876	195,70183	195,54031	248,06576	391,23943	395,37788
9,499851	14,03648	17,67998		55,13051	87,99746	91,826068	121,62026	197,26857	198,42752	246,27972	411,56801	393,73333
10,06375	14,25202	16,64797		57,15415	88,54606	94,039899	123,80609	197,47417	195,47837	251,83866	391,8446	403,21756
10,15146	14,75656	18,47656		56,98652	87,97588	92,01741	120,95598	201,55384	194,94485	245,68541	401,80575	399,07526
9,694225	14,37015	17,80557		57,9548	88,36803	94,805256	121,36982	198,50936	196,18076	247,97276	387,64528	390,65328
10,42139	14,18553	17,53402		56,69546	87,36274	91,856899	121,3579	197,09063	191,1047	251,09531	393,49722	392,07848
9,683091	14,18622	18,41509		56,59039	90,03114	94,212284	121,31917	199,44405	194,54213	281,47962	395,39628	392,12412
10,41618	14,50022	16,78509		55,92376	88,18513	91,916305	120,21153	197,04316	201,02508	244,52208	392,60164	391,17389
9,591468	14,16107	18,60038		56,60222	87,16611	93,834328	120,57354	193,56338	193,7503	247,86912	395,09381	393,00207
9,603654	15,13263	16,66709		57,61234	88,70967	92,067762	121,96228	197,73989	194,17237	247,57853	392,56912	389,29563
9,641312	14,90719	16,62272		55,87761	89,90854	93,770145	120,71001	195,02449	192,65564	249,38649	393,37344	388,85934
9,598422	15,38646	17,85902		57,74264	87,82279	91,88604	121,01396	198,3503	196,33121	246,70612	384,94298	401,42743
9,833897	14,12718	17,86588		55,04339	88,21634	93,730291	121,96917	193,31579	201,86342	251,71936	390,81043	398,90605
10,8356	14,33198	16,73303		55,86738	90,38163	92,000376	126,04233	194,65231	194,26484	247,55152	383,10079	392,04173
9,591972	14,25755	19,46667		57,41187	87,99561	92,06231	124,33255	196,06685	193,24405	264,12992	391,0307	395,16105
10,01525	14,14441	16,68795		60,88155	87,58548	93,646617	120,48049	197,96682	198,52694	248,93647	388,33779	390,935
9,862596	14,57957	17,35229		56,44006	90,86116	91,697521	126,42514	198,85338	193,38072	269,4924	394,08775	396,84801
9,785332	14,16555	19,0241		55,79862	90,37282	91,528163	125,81008	196,84535	194,66102	246,42564	405,618	392,13634
9,881478	14,41605	16,81294		55,49106	87,98124	97,265262	122,37498	197,29625	202,37552	248,5167	393,92241	404,48351
9,714687	14,82249	18,06494		58,95577	88,86724	91,921509	121,35992	199,02113	194,04917	247,1956	383,03714	391,75936
9,73622	14,12304	17,28263		55,93727	87,85908	91,867696	121,29273	197,2329	194,594	245,48153	391,67332	398,55906
10,63308	14,63939	16,99795		55,485	89,13341	94,27558	121,43096	194,7751	195,90014	255,65227	388,16439	394,08816
9,762516	14,25379	18,1556		55,69738	88,96087	92,021738	123,9778	195,10081	198,48423	248,57709	400,2116	401,28816
9,689285	14,24521	17,78883		58,01467	86,69342	92,661194	121,8105	197,83981	195,36835	248,43275	397,55539	390,21681
9,714426	14,31478	16,68392		55,47163	87,96429	94,548141	121,93206	202,95897	196,05583	246,34231	397,40155	388,19464
9,643576	14,22441	19,84792		54,95828	87,98586	94,488942	120,31485	195,55866	196,10165	249,17542	389,50484	388,66346



Marcin Nahajowski 246711  
Czwartek 15:15



odwrotnie	100 tys.			500 tys.			1 mln.			2 mln.		
	Quicksort	ShellSort	Merge Sort									
8,165895	4,56079	10,71953		56,07603	23,12565	51,04505	92,470412	49,51807	101,98476	194,8466	105,06341	206,4773
7,2559	3,460161	10,88491		60,69653	18,90816	50,98643	96,723334	45,14332	104,46347	192,42545	89,355706	205,1168
7,274688	3,435071	9,577498		53,80348	18,94798	48,68155	92,53093	45,30673	100,36081	210,61824	90,498222	201,63152
7,14004	3,420535	12,22983		55,43743	18,99949	49,25311	91,207693	45,00716	109,47606	200,77849	90,041802	195,07991
7,478458	3,404275	8,478367		53,37668	18,98084	48,74342	93,066248	44,46304	96,299734	192,12726	89,603467	193,82245
7,132446	3,451829	8,561476		52,94997	19,16093	46,51763	91,340492	44,39407	101,04595	191,33182	89,4892	221,5922
8,410072	3,405949	8,569376		53,85377	20,62696	52,62496	93,710533	44,9645	97,165585	193,03385	89,597231	193,60391
7,786638	3,476649	8,449963		55,13364	19,05133	52,98214	91,469815	44,48981	103,18061	192,6978	90,424077	198,49605
7,328556	3,841021	11,59985		52,80948	18,96173	46,9747	95,830927	45,06846	102,07215	193,14479	89,746244	199,61259
7,256236	3,400429	8,403987		54,76259	18,92468	46,58572	93,618322	45,07569	97,652293	192,35928	91,373944	195,28362
7,890315	3,404839	8,472543		54,61684	19,53641	48,58787	92,128314	45,0102	99,602515	206,07902	90,387119	194,8122
7,242165	3,405931	9,141438		52,8507	19,47045	46,23615	92,033876	44,11157	96,361838	193,61597	89,742896	198,22217
7,432287	3,402879	9,331853		52,13942	19,12169	46,23508	90,247904	44,66541	96,123869	191,49021	89,491132	198,65649
7,668449	3,410947	11,29559		58,7181	21,30409	49,09236	89,123618	43,70703	98,820555	191,78047	89,815386	205,7168
7,49384	3,395788	8,681028		53,05322	19,92085	46,26399	90,368378	43,97541	95,999628	191,78434	90,115037	201,01701
7,697273	3,519859	8,447269		53,30583	20,79742	47,04257	90,478529	43,83952	99,131311	192,03797	90,006615	194,08555
7,159266	3,497946	10,09086		53,57653	19,57136	48,01106	91,100424	44,69308	95,761239	195,83786	89,296174	194,44553
7,479301	3,406373	10,64272		52,87523	19,53534	46,67325	90,956308	43,47188	99,12153	207,24168	89,598073	197,87937
7,345008	3,434638	8,502088		53,05159	19,54605	46,38455	90,709585	43,5917	96,383257	196,40155	89,54052	194,68756
7,266301	3,405058	9,069513		56,22676	18,98998	46,53553	90,188779	45,25366	96,502633	193,69966	89,976885	197,29441
7,266523	3,445541	8,55745		52,67501	19,02936	46,32802	95,000807	44,17726	96,521174	193,67149	90,145199	209,12012
7,269677	3,464784	8,450433		53,74635	19,46847	47,96375	91,687162	45,35257	96,013248	192,37228	89,90385	199,08058
7,718367	3,740198	9,892871		53,77743	18,99798	46,52501	93,940497	46,6325	96,355614	192,67698	89,783787	197,92537
7,175708	3,452371	9,420044		54,06738	19,00655	46,46052	91,89662	44,04555	96,427476	192,61529	90,738715	196,82497
7,139739	3,433127	8,473715		53,48631	19,28313	46,61981	94,517657	44,15924	96,335706	192,68921	89,947858	199,43134
7,167862	3,518193	8,607353		52,93311	19,04775	46,80476	91,084325	44,00723	99,553142	212,27691	94,795749	208,82265
7,241486	3,466956	9,404752		51,67779	19,25675	51,53728	90,864025	43,41206	96,474081	193,01575	90,567966	199,16233
7,931912	3,407583	8,68344		51,63458	20,29029	46,96474	90,245656	44,62898	99,909737	195,85791	90,166886	207,56955
7,104469	3,400873	9,505332		53,22925	19,57412	49,75994	89,76066	45,20076	96,832178	192,89937	89,498332	197,57994
7,393213	3,427949	8,523068		54,93001	19,03228	46,38645	91,114852	44,02134	108,23815	195,10388	90,992407	197,85239
7,195296	3,408347	8,628934		53,23728	19,01634	48,3181	90,885076	44,07098	96,187162	205,49434	89,903446	197,50451
7,145998	3,435997	8,463571		53,24236	19,5273	46,25213	93,228544	44,77309	100,52383	193,57569	89,109466	199,33313
7,236399	3,428728	9,036131		54,41798	19,17869	48,59842	90,627283	43,71626	98,931364	193,54474	89,698356	209,22726
7,542986	3,445045	9,443118		53,08605	18,88821	46,21906	91,783559	44,29659	95,795064	191,45126	89,844581	203,87227
7,228275	4,156455	8,655365		52,65987	19,55039	48,35698	94,456909	44,5542	96,31937	193,20277	90,097952	199,83035
7,182026	3,413203	10,30287		53,0794	19,12316	46,54146	91,252375	43,38063	99,538778	192,06126	89,457033	208,94626
7,156909	3,458845	8,42418		54,08646	19,49154	48,08585	90,986429	44,91101	98,960136	199,40043	90,325338	197,87138
7,20427	3,518566	9,130998		53,54212	18,95808	46,84701	90,94804	44,65362	95,999593	192,75171	89,940047	205,77157
7,287791	3,403087	8,441238		55,35297	18,97827	46,54233	90,494299	44,41557	96,311166	196,05627	90,115268	197,2169
7,140555	3,413973	9,542537		53,27549	19,49023	48,57276	90,935912	44,96452	99,072246	192,77219	90,163698	208,27267
7,22299	3,431747	8,46993		54,31955	19,00032	46,80074	90,73564	44,31779	99,234623	194,42983	90,315467	202,40718
8,052456	3,403299	8,524966		53,57744	19,49954	48,26518	91,316168	44,17803	107,57319	194,12016	89,519054	200,38475
7,47318	3,407512	8,495697		53,21325	19,45986	46,41295	90,973752	44,15878	98,858468	193,05346	90,400774	209,35803
7,374951	3,405005	9,346949		55,54836	18,88955	46,41891	92,419756	44,98902	98,986037	192,01155	89,727741	201,39277
8,020969	3,511362	8,389296		52,65869	18,96423	48,55115	90,522222	44,55501	97,454927	193,65326	89,267289	199,11014
7,176213	3,438093	8,41548		52,9113	19,34604	46,7781	90,815211	43,98033	98,416928	204,22458	90,067238	199,40431
7,975444	3,465837	8,48359		55,33836	18,89195	46,50353	90,141467	44,72084	98,984217	198,44835	89,787868	201,37998
8,012997	3,416306	8,487411		53,7892	19,62008	48,43389	90,697892	44,46684	98,726897	195,83369	89,455202	199,88465

