Spis treści

1 Wstep **2**

2 Wyniki **3**

3 Wnioski: **5**

Sprawozdanie 2

Sortowanie 2

Paweł Żurek 200404

17.03.2014

1 Wstep

Do posortowania zbioru losowo wygenerowanych elementów użyłem :

- Qucik Sort (Sortowanie Szybkie)
- Merge Sort (Sortowanie przez scalanie)
- Heap Sort (Sortowanie przez kopcowanie)

Sortowanie zostało przeprowadzone dla następującej ilości elementów :

- 10
- 100
- 1000
- 10000
- 100000
- 1000000

Sortowanie zostało przeprowadzone dla następująco wypełnionych tablic :

- Posortowanych rosnąco
- Posortowanych malejąco
- Wypełnionych losowo

2 Wyniki

Dane zebrane w formie tabeli (tylko wartości uśrednione) :

Dane uśrednione pomiarów czasu wykonywania sortowania dla następująco wypełnionych elementów

Malejąco			
	Quick Sort	Merge Sort	Heap Sort
10	0,0006	0,0022	0,0012
100	0,0089	0,0482	0,0258
1000	0,0409	0,2035	0,2344
10000	0,3897	10,2117	1,8548
100000	4,6819	970,3489	23,4913
1000000	50,3577	3102,604	287,4419

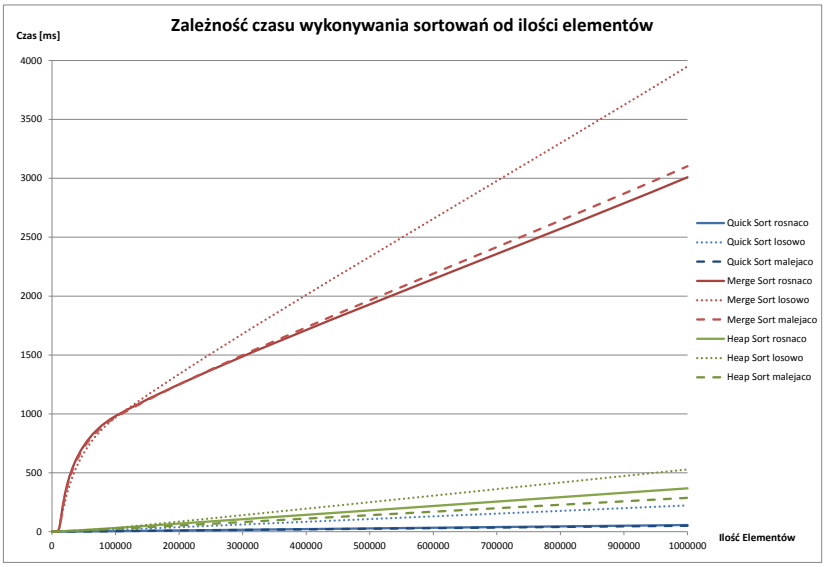
Losowo			
	Quick Sort	Merge Sort	Heap Sort
10	0,0004	0,0008	0,0005
100	0,0067	0,017	0,0116
1000	0,1812	0,3903	0,3296
10000	1,3755	10,9138	2,312
100000	15,561	969,0219	30,8183
1000000	222,9609	3949,789	528,4516

Rosnąco			
	Quick Sort	Merge Sort	Heap Sort
10	0,0006	0,0023	0,0013
100	0,0064	0,0266	0,0583
1000	0,0522	0,2689	0,392
10000	0,4191	10,772	2,3594
100000	4,9767	981,5392	31,477
1000000	56,7099	3008,892	368,0569

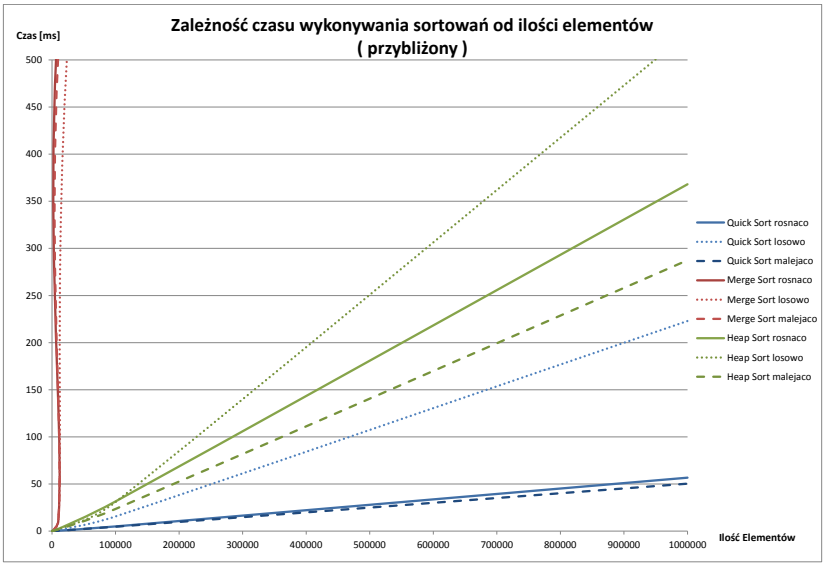
Czasy są w mili seknudach.

Te oraz pełne dane dostępne także w pliku dane.xlsx . Są tam dane odnośnie każdego wykonania dla każdej ilości elementów oraz każdego elementu.

Dane wyświetlone za pomoca wykresu (wykres dostępny osobno w pliku Wykres1.pdf)



Wykres przybliżony (wykres dostępny osobno w pliku Wykres2.pdf)



3 Wnioski:

- Dla każdego algorytmu, posortowanie elementów wypełnionych losowo trwało najdłużej
- Dla elementów posortowanych rosnąco i malejąco różnice w czasie są na tyle małe, iż można uznać, że zajmują podobną ilość czasu dla każdego algorytmu. Aczolwiek z tych trzech, Merge Sort jest jedyną metodą, której posortowanie elementów posortowanych malejąco zajęło więcej czasu niż elementów posortowanych rosnąco
- Posortowanie elementów wypełnionych losowo dla metody Quick Sort (mimo tego, że dla tej metody najwolniejsze) i tak okazało się szybsze niż inne metody (nawet z elementami posortowanymi)
- Posortowanie elementów wstępnie posortowanych okazało się szybsze. Dzieje się tak, ponieważ większość algorytmów zamienia elementy po porównaniu elementów. Jeśli zbiór już jest posortowany, jest po prostu mniej operacji zamiany wartości elementów. To znacznie przyśpiesza wykonywanie algorytmu. Dla algorytmu Quick Sort więcej niż 4 razy
- Na wykresie numer 2, przybliżonym, widać dziwne zachowanie lini Merge Sort. Jest to spowodowane małą ilością punktów pomiarowych. Program wypełnia sam miejsca gdzie nie ma tych punktów przwidując ich położenie, co często nie jest poprawnie robione
- Sortowanie zostało wykonane za każdym razem 10 razy. Za każdym razem czas wykonywania różni się, co widać w pliku z danymi. Prawdopobnie jest to spowodowane tym, iż za każdym razem procesor wykonuje inne obliczenia w tle podczas wykonywania programu

Dokumentacja program dostępna w pliku pdf o nazwie "Dokumentacja" (zapisana w LaTeXu) oraz w DoxyGen'ie (dostępna w pliku : dox/html/index.html)