|  |  |
| --- | --- |
| Wydział Elektrotechniki, Automatyki, Informatyki i Inżynierii Biomedycznej | |
| Wstęp do Informatyki | |
| Zajęcia 8,9 – Sortowanie | |
| Imię i Nazwisko | Daniel Rubak |
| Data wykonania ćwiczenia | 10.12.2017 |

1. **Kod programu (wersja podstawowa)**

Kod programu z zaimplementowanymi wszystkimi wymaganymi algorytmami sortowania w formie pliku wykonywalnego został dołączony do sprawozdania.

1. **Tabela zależności czasu działania algorytmu od wartości zadanej.**

Na potrzeby niniejszego sprawozdania zdecydowano się przeprowadzić testy szybkości działania poszczególnych algorytmów sortowania w zależności od wielkości danych wejściowych. Przyjęto, że długość listy do posortowania będzie zmieniała się w następujący sposób:

Dla każdej z tych wartości listę uzupełniono losowymi liczbami z zakresu od 0 do 100. Ponadto, aby w jak największym stopniu usprawnić proces analizy otrzymanych danych, algorytmy były testowane dokładnie na tej samych danych. Oznacza to, że za każdym razem, gdy zwiększała się długość listy z liczbami do posortowania, wszystkie algorytmy musiały uporządkować dokładnie ten sam zestaw danych. Zebrane dane zostały przedstawione w tabeli poniżej:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Długość listy | Sortowanie przez wybieranie | Sortowanie przez wstawianie | Sortowanie bąbelkowe | Szybkie sortowanie | Sortowanie przez scalanie | Sortowanie przez kopcowanie |
| 100 | 0,000591999 | 0,001435485 | 0,001436306 | 0,000217436 | 0,000540717 | 0,000467281 |
| 1000 | 0,058190287 | 0,139302393 | 0,135420962 | 0,00354256 | 0,005353019 | 0,008099272 |
| 2000 | 0,229180846 | 0,552473166 | 0,563999716 | 0,00917004 | 0,012802446 | 0,017222543 |
| 3000 | 0,518359567 | 1,33956471 | 1,259770347 | 0,014950956 | 0,018015978 | 0,026438121 |
| 4000 | 0,904444425 | 2,213889378 | 2,25493384 | 0,022568998 | 0,027636889 | 0,038744978 |
| 5000 | 1,480524537 | 3,541293283 | 3,609504481 | 0,035457597 | 0,034598932 | 0,051924449 |
| 6000 | 2,184107261 | 5,169548509 | 5,050404861 | 0,037830928 | 0,038616568 | 0,060998899 |
| 7000 | 2,885975525 | 7,004947276 | 6,986891503 | 0,053036653 | 0,050565681 | 0,071465348 |
| 8000 | 3,675511374 | 8,952868571 | 8,990932524 | 0,065018587 | 0,057591724 | 0,083971999 |
| 9000 | 4,688193615 | 11,34395732 | 11,44121628 | 0,075447702 | 0,065446484 | 0,090264915 |

1. **Wykresy zależności czasu wykonania danego algorytmu sortowania od rozmiaru danych.**

Poniżej zamieszczono wykresy, które w lepszy sposób obrazują dane zamieszczone w tabeli powyżej.

1. **Wnioski**

Najwolniejszym algorytmem okazały się algorytmy bąbelkowy oraz przez wstawienie, czego się w zasadzie spodziewano. Najszybszy okazał się algorytm przez scalanie jednakże quicksort był niewiele wolniejszy. Z wykresów widać, iż przy małych porcjach danych można posłużyć się pierwszymi 3 algorytmami gdyż dają zadowalające wyniki. W przypadku dużych porcji danych do posortowania zaleca się stosowanie któregoś z algorytmów: quicksort, mergesort lub heapsort.