Popa Anda-Ioana

Grupa 132

SD-tema 1

Timpi teste (secunde)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Test | Nr. in test | Valoare maxima | Bubblesort | Countsort | Mergesort | Quicksort  Mediana din 3 | Radixsort8 | Radixsort256 | Nativsort |
| 1 | 1000 | 1000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1000 | 1000000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 10000 | 1000 | 0.062 | 0.18 | 0.002 | 0.01 | 0 | 0 | 0 |
| 4 | 10000 | 1000000 | 0.061 | 0.152 | 0 | 0.01 | 0.008 | 0 | 0 |
| 5 | 100000 | 1000 | 12.55 | 14.844 | 0.013 | 0.608 | 0.01 | 0 | 0 |
| 6 | 100000 | 1000000 | 12.608 | 15.385 | 0.02 | 0.639 | 0.011 | 0 | 0 |
| 7 | 1000000 | 1000 | Timp depasit | Timp depasit | 0.215 | 9.46 | 0.059 | 0.022 | 0.011 |
| 8 | 1000000 | 1000000 | Timp depasit | Timp depasit | 0.241 | 7.556 | 0.061 | 0.023 | 0.01 |

* Fiecare algoritm a fost testat pentru 8 seturi de date (teste.in). Timpii obtinuti se gasesc in tabelul de mai sus. Lungimea vectorilor de sortat a variat intre 10^3 si 10^6, iar valoarea maxima a numerelor intre 10^3 si 10^6.
* Vectorul de sortat a fost alocat dinamic. Elementele vectorului au fost generate aleator, tinand cont de valorile maxime citite din fisierul teste.in.
* Timpul de rulare al unui algoritm l-am obtinut prin scaderea valorilor de timp final si initial. In cadrul metodelor am folosit si un timp curent, care a permis intreruperea algoritmului in cazul depasirii a 30 de secunde. Datele de iesire, cat si mesajul de timp depasit sunt afisate in fisierul timpi.out.
* Pentru metoda quicksort in loc de alegerea pivotului ca prim element din vector, am folosit alegerea ca fiind mediana din 3, crescand astfel sansele ca lungimile celor 2 vectori cu elemente mai mici, respectiv mai mari decat pivotul sa fie mai apropiate. Indicii celor 3 elemente dintre care s-a ales pivotul au fost generati aleator.
* Din analiza comparativa se observa ca marimea timpilor de executie este influentata de lungimea vectorului de sortat si mai putin de ordinul de marime al numerelor din vector.
* Metoda radixsort nu se bazeaza pe compararea valorilor din vector. Am testat algoritmul pentru doua baze (8 si 256). Am observat ca pentru baze mai mari timpii sunt mai buni.
* Algoritmul nativ a obtinut cei mai buni timpi.
* Se observa ca metodele quicksort, mergesort si radixsort sunt mai rapide pentru aceeasi lungime a vectorului de sortat.





