# Sprawozdanie

## Projekt 2 – Algorytmy sortowania

## Wojciech Konury

## Wprowadzenie

Projekt polegał na zbadaniu czasu wykonywania różnych algorytmów sortowania w zależności od ilości danych. Algorytmy sortowania testowano na dwóch różnych danych wejściowych:

Tablica wypełniona losowymi elementami

* Tablica wypełniona posortowanymi elementami

Badano czasy wykonania trzech różnych algorytmów:

* Bubble Sort
* Merge Sort
* Heap Sort

## Przedstawienie czasu wykonania algorytmów w zależności od ilości danych dla tablicy wypełnionej losowymi danymi

|  |  |  |  |
| --- | --- | --- | --- |
|  | Średni czas wykonania [s] | | |
| Ilość elementów w tablicy | Merge Sort | Heap Sort | Bubble Sort |
| 10 | 7.08E-07 | 8.00E-07 | 6.04E-07 |
| 100 | 8.48E-06 | 1.06E-05 | 3.35E-05 |
| 1000 | 0.000122428 | 0.000154872 | 0.00267698 |
| 10000 | 0.00126835 | 0.00162845 | 0.328749 |
| 100000 | 0.0142742 | 0.0187957 | 35.8928 |
| 1000000 | 0.155645 | 0.239987 |  |

Na podstawie powyższych danych możemy zaobserwować, że najszybszymi algorytmami sortowania tablicy wypełnionej losowymi elementami jest algorytm Merge Sort lub Heap Sort, których czasy wykonania są podobne do siebie. Algorytm Bubble sort widocznie odstaje od pozostałych, jednakże dla tablicy z małą ilością elementów (10) jest on najszybszy.

## Przedstawienie czasu wykonania algorytmów w zależności od ilości danych dla tablicy wypełnionej posortowanymi danymi

|  |  |  |  |
| --- | --- | --- | --- |
|  | Średni czas wykonania [s] | | |
| Ilość elementów w tablicy | Merge Sort | Heap Sort | Bubble Sort |
| 10 | 5.12E-07 | 6.56E-07 | 3.25E-07 |
| 100 | 6.49E-06 | 8.99E-06 | 2.70E-05 |
| 1000 | 7.75E-05 | 0.000128441 | 0.0023648 |
| 10000 | 0.000913107 | 0.00147974 | 0.210238 |
| 100000 | 0.0094136 | 0.016447 | 20.6818 |
| 1000000 | 0.106864 | 0.185156 |  |

Na podstawie wyników uzyskanych dla tablicy wypełnionej posortowanymi elementami również możemy zauważyć, że algorytm Bubble Sort jest najszybszy dla tablic z małymi ilościami elementów (10 – 100). Dla tablic z większą ilością elementów dalej najoptymalniejszym wyborem jest algorytm Merge Sort lub Heap Sort, których czasy wykonywania są porównywalne.

Możemy również zauważyć, że czasy sortowania uległy skróceniu przy tych danych wejściowych.

## Różnice w czasie wykonywania algorytmów pomiędzy tablicą wypełnioną losowo a tablicą wypełnioną posortowanymi elementami

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Średni czas wykonania [s] | | | | | |
|  | Merge Sort | | Heap Sort | | Bubble Sort | |
| Ilość elementów w tablicy | Losowo | Posortowane | Losowo | Posortowane | Losowo | Posortowane |
| 10 | 7.08E-07 | 5.12E-07 | 8.00E-07 | 6.56E-07 | 6.04E-07 | 3.25E-07 |
| 100 | 8.48E-06 | 6.49E-06 | 1.06E-05 | 8.99E-06 | 3.35E-05 | 2.70E-05 |
| 1000 | 0.000122428 | 7.75E-05 | 0.000154872 | 0.000128441 | 0.00267698 | 0.0023648 |
| 10000 | 0.00126835 | 0.000913107 | 0.00162845 | 0.00147974 | 0.328749 | 0.210238 |
| 100000 | 0.0142742 | 0.0094136 | 0.0187957 | 0.016447 | 35.8928 | 20.6818 |
| 1000000 | 0.155645 | 0.106864 | 0.239987 | 0.185156 |  |  |

Źródła:

* <https://www.geeksforgeeks.org/>
* <https://www.youtube.com/user/kolboch/>