Algorytmy i struktury danych (Sortowanie i kopce)

Przyjmując, że $t1[] = 1, 2, 3, 4, 5, 6, 7$ oraz $t2[] = 7, 6, 5, 4, 3, 2, 1$ i stosując algorytmy sortujące ściśle wg procedur z pliku sorty2020.cc i wykonaj polecenia:

Zadanie 1 Ile dokładnie porównań (między elementami tablicy) wykona insertionSort(t2) a ile insertionSsort(t1)?

Posortowana: n-1 porównań. Nieposortowana: $\frac{n(n-1)}{2}$ porównań.
Dla t1 wykona 6 porównań. Dla t2 wykona 21 porównań.

Zadanie 2 Ile co najwyżej porównań (między elementami tablic) wykona procedura scalająca merge dwie tablice n-elementowe?

Zadanie 3 Jaka jest pesymistyczna złożoność czasowa procedury mergeSort? Odpowiedź uzasadnij.

Dla wersji rekurencyjnej i iteracyjnej: $O(nlogn)$ Porównujemy elementy i scalamy je w jedną tablicę. W każdym kroku wykonujemy n porównań. W każdym kroku dzielimy tablicę na dwie części o połowie długości. W sumie wykonujemy $logn$ kroków. Złożoność czasowa zawsze jest $O(nlogn)$.

Zadanie 4 Ile co najwyżej porównań (między elementami tablicy) wykona procedura partition?

Przy jednym wywołaniu partition wykona się maksymalnie $n - 1$ porównań. To przypadek, gdy nasze odwrócone elementy są najbliżej pivota lub musimy zamienić stronami każdy z elementó. W obu przypadkach musimy każdą wartość po stronie prawej i lewej porównać z pivotem.

Zadanie 5 Jak jest średnia a jaka pesymistyczna złożoność quickSort. Odpowiedź uzasadnij.

Średnia złożoność czasowa: $O(nlogn)$
Pesymistyczna złożoność czasowa: $O(n^2)$
Pesymistyczną złożoność osiągamy, gdy

Zadanie 6 Jaka jest złożoność funkcji buildheap? Przeprowadź dowód - uzasadnij swoją odpowiedź.

Zadanie 7

Zadanie 8

Zadanie 9