Algorytmy i Struktury Danych

13 marca 2021

Ćwiczenia 3: QuickSort i HeapSort

Zadania obowiązkowe

Zadanie 1. Proszę zaimplementować algorytm QuickSort do sortowania n elementowej tablicy tak, żeby zawsze używał najwyżej $O(\log n)$ dodatkowej pamięci na stosie, niezależnie od jakości podziałów w funkcji `partition`.

Zadanie 2. Proszę zaimplementować funkcję wstawiającą dowolny element do kopca binarnego.

Zadania standardowe

Zadanie 3. Proszę zaimplementować algorytm QuickSort bez użycia rekurencji (ale można wykorzystać własny stos).

Zadanie 4. Proszę zaproponować algorytm scalający k posortowanych list.

Zadanie 5. Proszę przedstawić W jaki sposób zrealizować strukturę danych, która pozwala wykonywać operacje:

1. Insert
2. RemoveMin
3. RemoveMax

tak, żeby wszystkie operacje działały w czasie $O(\log n)$.

Zadanie 6. Proszę przedstawić W jaki sposób zrealizować strukturę danych, która pozwala wykonywać operacje:

1. Insert
2. RemoveMedian (wyciągnięcie mediany)

BIT ALGO START 1

tak, żeby wszystkie operacje działały w czasie $O(\log n)$.

Zadanie 7. (Partition Hoare'a) Proszę zaimplementować funkcję `partition` z algorytmu QuickSort według pomysłu Hoare'a (tj. mamy dwa indeksy, i oraz j, wędrujące z obu końców tablicy w stronę środka i zamieniamy elementy tablicy pod nimi jeśli mniejszy indeks wskazuje na wartość większą od piwota, a większy na mniejszą.

Zadania dodatkowe

Zadanie 9. (Select) Proszę zaimplementować algorytm znajdowania k-go co do wielkości elementu w tablicy n elementowej w "spodziewanym" czasie $O(n)$ na podstawie randomizowanego Partition z QuickSort'a