Ion Melania-Victoria

Tema 2 laborator SDA

Problema Find-median-from-data-stream (40p)

Procesul gândirii:

Ideea Centrală:

Se utilizează două heap-uri, un max-heap și un min-heap, pentru a împărți setul de numere în două jumătăți aproximativ egale. Max-heap-ul conține jumătatea inferioară a numerelor, iar min-heap-ul conține jumătatea superioară.

Adăugarea unui Număr:

Întotdeauna se adaugă noul număr în max-heap (înmulțit cu -1 pentru a transforma min-heap-ul implicit oferit de Python în max-heap).
Se asigură că cel mai mare număr din max-heap este mai mic sau egal cu cel mai mic număr din min-heap.
Se echilibrează dimensiunile heap-urilor astfel încât să difere cu cel mult 1.

Găsirea Medianei:

Dacă unul dintre heap-uri este mai mare, mediana este elementul de top din acel heap. Dacă sunt de aceeași mărime, mediana este media celor două valori din varful heap-urilor.

Complexitate:

1) Complexitatea Spațiului: O(n) - se păstrează toate elementele în cele două heap-uri.
2) Complexitatea Timpului:
   - adăugare (addNum): O(log n) - Adăugarea într-un heap este O(log n)
   - găsirea Medianei (findMedian): O(1) - Accesul la elementul de varf din heap se face în timp constant.

Motivația Alegerii Structurii de Date - Heap (Min-Heap si Max-Heap:

Două Heap-uri pentru Echilibru: Utilizarea a două heap-uri diferite (max și min) permite menținerea unui echilibru între jumătatea inferioară și superioară a setului de numere. Acest lucru este esențial pentru calculul rapid al medianei.

Heap-uri pentru performanță: Heap-urile sunt ideale pentru operații frecvente de inserare și obținere a maximului/minimului, așa cum este necesar pentru a rezolva eficient această problemă.