algowiki.miraheze.org

Növekvő részsorozatokra bontás – Algowiki

3-4 perc

A feladatszöveg nagyon tömören leírja a megválaszolandó kérdést: minimum hány darab szigorúan monoton növekvő részsorozatra lehet felbontani egy N egész számból álló sorozatot, amelyben minden elem 1 és 1 millió közötti értéket vesz fel.

Korlátok:

* $1 \leq N \leq 10^5$
* futásidő $\leq$ 0,5 másodperc
* memóriahasználat $\leq$ 32 MiB

megoldás[szerkesztés | forrásszöveg szerkesztése]

első ötlet[szerkesztés | forrásszöveg szerkesztése]

A feladat szempontjából fontos egy stratégiát/módszert találni, amellyel az elemeket a megfelelő részsorozatokba és, hogyan kerülnek az előállított sorozatok tárolásra. Először gondolhatunk mindenféle brute-force és branch-and-bound algoritmusra, azonban azok futásideje messze

nem elégséges a feladat korlátainak teljesítéséhez.

második ötlet[szerkesztés | forrásszöveg szerkesztése**]**

Olyan ötletünk támadhat, hogy esetleg megpróbálhatnánk egy végigmenetellel beosztani az elemeket a lehető legkevesebb számú részsorozatba. Ehhez azt csinálhatjuk, hogy amikor egy elemet feldolgozunk, akkor megnézzük, hogy az eddig elkészített részsorozatok valamelyikéhez hozzá tudjuk-e adni. Ha nem, akkor értelemszerűen létre kell hoznunk egy új részsorozatot. Ha igen, akkor felmerül a kérdés, hogyan döntsük el, hogy melyik részsorozathoz tegyük hozzá (ha esetleg több is lehetséges). Megnézve a lehetőségeket arra juthatunk, hogy azok közül, amelyekhez a jelenlegi szám hozzátehető, ahhoz kell hozzátenni, amelyiknek az utolsó eleme a legnagyobb. Ennek az az oka, hogy így lesz majd a további elemek feldolgozásakor a legtöbb lehetőségünk további elemeket is már létező részsorozatokhoz hozzáadni. Ha például a 6 a jelenlegi elem, és két részsorozatunk van már: az egyiknek 2 az utolsó eleme, a másiknak 5; akkor ha a kettő végű részsorozathoz adjuk hozzá, akkor ha a 6 után például 3 következik, akkor új részsorozatot kellene létrehozni. Ezután felmerül a kérdés, hogy hogyan is tudjuk a leggyorsabban kiválasztani, hogy melyik sorozathoz érdemes hozzáfűzni a jelenlegi elemet.

harmadik ötlet[szerkesztés | forrásszöveg szerkesztése**]**

Megtehetnénk azt, hogy az eddigi sorozatokat végignézve kiválasztjuk azt, amelyik megfelel a fentebb említett

feltételnek. Sajnos ennek a futásideje nem elégségesen gyors (legrosszabb esetben $O(n^2)$ a futásideje: ha csökkenő sorrendben vannak a számok a bemeneten). Egy fontos tulajdonság, amit megfigyelhetünk, hogy az algoritmus futása alatt minden időpillanatban az eddigi részsorozatok utolsó elemei nem akármilyen, hanem növekvő/csökkenő (nézőpont kérdése) sorrendben vannak. És mivel a sorozat kiválasztása szempontjából csak az utolsó elem számít, ezért a következő problémát kell már csak megoldani: rendezett sorozat legnagyobb olyan elemét kell megkeresni, amelyik kisebb egy megadott számnál. Erre pedig létezik egy remek algoritmus: a bináris keresés. Így logaritmikus aszimptotikus komplexitással eldönthető, hogy hova illesszük be a jelenlegi elemet; a teljes megoldás $O(n \cdot \log(n))$ futásidejű. Továbbá, mivel a feladat csak a szigorúan monoton növő részsorozatok minimális számát kéri, ezért természetesen elég csak a sorozatok jelenlegi utolsó elemét tárolni, így a memóriaigény $O(n)$.

fontosabb észrevételek[szerkesztés | forrásszöveg szerkesztése]

- a mohó szemléletű algoritmus észrevétele
- a létrehozott részsorozatok utolsó elemei rendezett sorrendűek