SzA III. gyakorlat

Rendezettek vagyunk és jól keresünk

2012. szeptember 20.

1. Rendezzük a következő listát beszúrásos, buborék, kiválasztásos, összefésüléses és gyorsrendezés segítségével: $[4, 11, 9, 10, 5, 6, 8, 1, 2, 16]$.

2. Rendezzük a következő listát ládarendezéssel, ha tudjuk, hogy csak 0 és 10 közötti egész számok szerepelhetnek benne: $[6, 4, 3, 8, 6, 3, 3, 5, 2]$

3. Adottak a $p_0 = (0, 0), p_1 = (x_1, y_1), \ldots p_n = (x_n, y_n), p_{n+1} = (100, 0)$ pontok a síkban ($n \geq 1$) úgy, hogy $1 \leq i \leq n$ esetén x_i és y_i racionális számok, $0 < x_i < 100$, és semelyik három pont nem esik egy egyenesbe. Egyenes szakaszokkal akarjuk ezeket a pontokat valamilyen sorrendben összekötni úgy, hogy egy $n + 2$ csúcsú zárt töröttvonalat kapjunk, amiben a behúzott szakaszok nem metszik egymást. Adjunk egy legfeljebb $c \cdot n \log n$ lépést használó algoritmust annak meghatározására, melyik pontot melyikkel kössük össze!

4. Rendezzük a következő számokat kupacos rendezéssel: 10, 9, 6, 5, 2, 3

5. Szúrjuk be egy kezdetben üres bináris keresőfába a 6, 2, 1, 4, 5, 8, 9, 7, 10 elemeket, majd töröljük a 7, 8, 2 elemeket!

---

6. A $[6, 4, 8, 3, 7, 2, 5, 1]$ tömb rendezése során (a rendező algoritmus néhány lépése után) a következő közbülső állapot jött létre: $[4, 6, 3, 8, 7, 2, 5, 1]$. Az alább felsorolt módszerek közül mely(ek) alkalmazásakor fordulhatott elő?

   (a) beszúrásos rendezés
   (b) buborékrendezés
   (c) összefésüléses rendezés
   (d) gyorsrendezés

7. Szeretnénk n db SZA-hallgató ZH-ereményeit (csak az összpontszámot) növekvő sorrendben felsorolni. Adjunk erre $c \cdot n$ lépést felhasználó algoritmust!

8. Adjunk $c \cdot n$ lépésszámú algoritmust n olyan egész számból álló sorozat rendezésére, melynek elemei az $\{1, \ldots, 3n\}$ tartományba esnek!

9. **[pótpótZH, 2010. ősz]** A valós számokból álló $a_1, \ldots, a_n$ sorozat olyan, hogy az $a_1^3, a_2^3, \ldots, a_n^3$ sorozat egy darabig nő, utána csökken. Adjunk konstansszor n összehasonlítást használó algoritmust, ami rendezi az $a_1, \ldots, a_n$ sorozatot.

10. Az $A[1 \ldots n]$ tömbben egész számokat tárolunk, ugyanaz a szám többször is szerepelhet. Határozzuk meg $c \cdot n \log n$ lépésben az összes olyan számot, amelyik egynél többször fordul elő a tömbben!

11. Az $A[1 : n]$ tömbben levő elemekről tudjuk, hogy $A[1] \neq A[n]$. Adjunk $c \log n$ összehasonlítást használó algoritmust, amely talál egy olyan i indexet, hogy $A[i] \neq A[i + 1]$!

12. Adott a síkon n pont, melyek koordinátái $(a_1, b_1) \ldots (a_n, b_n)$. Olyan $P = (x, y)$ pontot keresünk a síkon, amire az alábbi összeg minimális.

$$\sum_{i=1}^{n} (|a_i - x| + |b_i - y|)$$

Adjunk algoritmust, ami $c \cdot n \log n$ lépésben meghatároz egy ilyen P pontot!

13. 🖉 Adottak a sík egész koordinátájú $P_1 = (x_1, y_1), \ldots, P_n = (x_n, y_n)$ koordinátájú pontjai. Javasoljunk egy legfeljebb $c \cdot n$ lépésszámú módszert olyan $P_i \neq P_j$ pontok kiválasztására, amelyeken átmenő egyenes által meghatározott félsíkok közül az egyik tartalmazza az összes pontot!

---

14. A 10 elemű A tömb első 8 elemére legyen $A[i] = 2i (1 \leq i \leq 8)$, és tekintsük ezt, mint egy 8 elemű kupacot. Rajzoljuk le az ehhez tartozó fát! Hajtsuk végre rajta a BESZÚR(3), BESZÚR(1), MINTÖR műveletsort!

15. Egy orvosi rendelőben a regisztrációnál kell bejelentkezni, ahol az ott dolgozók eldöntik, hogy a beteg az épp rendelő két orvos közül A-hoz vagy B-hez kell kerüljön, vagy bármelyikükhöz kerülhet. Ezen kívül, a beutaló ismeretében, a beteghez egy, a sürgősséget kifejező, számot is rendelnek. Amikor valamelyik orvos végzett egy beteggel, akkor azon betegek közül, akiket nem csak a másik orvos láthat el behívja a legnagyobb sürgősségi számút. Tegyük fel, hogy a kiosztott sürgősségi számok egymástól különbözőek. Írjunk le egy olyan adatszerkezetet, ami abban az esetben ha n beteg várakozik, akkor a regisztráción az új beteg beillesztését, illetve az orvosoknak a következő beteg kiválasztását $c \log n$ lépésben lehetővé teszi!

16. Adott egy n elemet tartalmazó kupac és egy k kulcs. Keressük meg a kupac k-nál kisebb elemeit! Ha m ilyen elem van, akkor az algoritmus $c \cdot m$ elemi lépést használhat.

17. 🖉 Egy kupacba beraktunk egy új x elemet, majd végrehajtottunk egy MINTÖR műveletet. Mikor fordul elő, hogy végül az eredeti kupacot kapjuk vissza?

---

18. Adott egy n csúcsú és egy k csúcsú keresőfa. A két fában tárolt összes elemből $c(n + k)$ lépésben készítsünk rendezett tömböt!

19. Egy bináris keresőfában csupa különböző egész számot tárolunk. Lehetséges-e, hogy egy $KERES(x)$ hívás során a keresési út mentén a $20, 18, 3, 15, 5, 8, 9$ kulcsokat látjuk ebben a sorrendben? Ha lehetséges, határozzuk meg az összes olyan x egész számot, amire ez megtörténhet! Ha nem lehetséges, miért nem?

20. Egy bináris fa csúcsai 0 és 9 közötti egész számokkal vannak megcímkézve. Az inorder bejárás során a címkék sorrendje: 9, 3, 1, 0, 4, 2, 7, 6, 8, 5, a postorder bejárásnál pedig 9, 1, 4, 0, 3, x, 7, 5, y, 2. Mi lehet az x és mi az y?

21. Egy bináris keresőfa csúcsait egy, a gyökértől egy levélig menő út szerint három osztályba soroljuk: B az úttól balra levő, U az útra eső, J pedig az úttól jobbra levő csúcsok halmazát jelöli. Igaz-e mindig, hogy minden B-beli csúcs kulcsa kisebb tetszőleges U-beli csúcs kulcsánál, és minden U-belis csúcs kulcsa kisebb tetszőleges J-beli csúcs kulcsánál?

22. Az MSc-re jelentkezőknek a felvételit alkotó 3 témakör mindegyikéből lesz egy írásbeli pontszámuk (P_1, P_2, P_3), és keletkezik egy felvélteli pontszámuk is (FP). Tegyük fel, hogy a P_i-k 1 és 30 közötti egészek, míg az FP tetszőleges pozitív egész szám lehet. Adjunk meg egy olyan adatszerkezetet, amivel a következő műveletek az adott időben végrehajthatóak (n a jelentkezők számát jelöli)!
BESZÚR(P_1, P_2, P_3, FP): az adott pontszámok beillesztése – átlagosan $c \log n$
KERES(p): a pontosan p felvételi ponttal $(FP = p)$ rendelkező jelentkezők számát határozza meg – átlagosan $c \log n$
KORLÁT(i, q): az írásbelin az i-edik témakörből legalább q pontot elért jelentkezők számát határozza meg – konstans

23. 🖉 Adott $2^k - 1$ különböző szám, mindegyik az $\{1, 2, \ldots, n\}$ halmazból, ezekből kell egy k mélységű bináris keresőfát készíteni. Adjunk olyan algoritmust, amely ezt $c \cdot n$ lépésben megcsinálja!