Algoritmusok és gráfok
MINTAZH
2019. ősz

1. Az alábbi futási idők közül pontosan egyikre igaz, hogy $O(\log n)$.
   (a) $10^{10} \log_2 n + \log_2(\frac{n}{2}) - 10$ (b) $(\log_2 n)^3$ (c) $\frac{1}{10^{100}} 2^n$
   Válassza ki, hogy melyik az és erre bizonyítsa is ezt be megfelelő c konstans és n_0 küszöb megadásával.

2. Egy három-szintű bináris keresőfában a $3, 5, 7, 8, 9, 10, 11$ csúcsokat tároljuk.
   (a) Rajzolja fel a fát és indokolja meg, hogy miért csak így nézhet ki.
   (b) Az órán tanult módszerrel törölje a fából a 8-at, majd a 10-et, végül szúrja be az így kapott fába a 8-at. Mindegyik művelet után rajzolja fel a kapott fát.

3. A 0 és 70 közötti páratlan számokat szúrjuk be valamilyen sorrendben, nyílt címzéssel, lineáris próbával egy kezdetben üres 47 méretű hash táblába, a $h(K) = K$ maradéka 47-tel osztva hash függvényt használva. Mutassa meg, hogy mindegy, hogy a számok milyen sorrendben érkeznek, az ütközések száma minden sorrend esetén ugyanannyi.

4. Szomszédossági mátrixával adott egy $n \geq 1$ csúcsú irányított gráf. Adjon algoritmust, ami $O(n^2)$ lépésben eldönti, hogy van-e olyan csúcs, melynek ki-foka és be-foka megegyezik.

5. Inputként adott egy $n \geq 1$ méretű tömb, mely csupa különböző egész számot tartalmaz. Adjon $O(n \log n)$ lépésszámú algoritmust, ami eldönti, hogy van-e két olyan szám ebben a tömbben, melyek különbsége 42.

6. Adott egy $n \geq 1$ csúcsú, e élű irányított G gráf és benne egy olyan s csúcs, ahonnan nem érhető el irányított úton a gráf összes csúcsa.
   Adjon $O(e(n^2))$ lépésszámú algoritmust annak eldöntésére, hogy egyetlen él megfordításával kaphatunk-e G-ből olyan gráfot, ahol s-ből minden más csúcs elérhető irányított úton.

Algoritmusok és gráfok - Zárthelyi
2019. november 21.

A válaszokat indokolni kell. Hivatkozni csak az előadáson tanultakra lehet.

1. Az alábbi pszeudokód inputja egy $n \geq 2$ szám, a pszeudokódban lépésnek egy darab * kiírása és az értékadás számít. Mutassa meg, hogy a pszeudokód által megadott algoritmus lépésszáma $O(n^2)$.

```
ciklus j = 0-tól (n-1)-ig:
    k  := 0
    ciklus amíg k < j:
        kiírunk egy  darab *-ot
        k := k + 1
    ciklus vége
ciklus vége
```

2. A $12, x, 2, 9, 10, 5$ tömböt kiválasztásos rendezéssel és buborékrendezéssel is rendeztük. Adja meg x összes lehetséges értékét, ha tudjuk, hogy x a tömbben máshol elő nem forduló egész szám és a kiválasztásos rendezés során volt $2, 5, 12, 9, 10, x$ átmeneti állapot, a buborékrendezés során pedig $2, x, 9, 5, 10, 12$ állapot.

3. Egy kezdetben üres, 11 méretű hash táblába nyílt címzéssel, lineáris próbával szúrtunk be kilenc egész számot. A használt hash függvény értéke a K kulcs esetén K maradéka 11-gyel osztva. A beszúrások után egyetlen számot kitöröltünk a táblából, de a törölt cellát véletlenül nem állítottuk töröltre, üresnek látszik. Az alábbi táblát kaptuk, amiben tehát a törlés helye nem látszik. Melyik cellában történt a törlés?

0	1	2	3	4	5	6	7	8	9	10
	13	2	28	15		7	18		21	10

4. Egy öt csúcsú AVL-fát preorder bejárással bejárva a csúcsokat $9, 5, x, 8, 12$ sorrendben látogatjuk meg. Adja meg, hogy hogyan néz ki ez a fa és adja meg x összes lehetséges értékét, ha tudjuk, hogy x egy olyan pozitív egész szám, ami máshol nem fordul elő a fában.

5. Adott két tömb, mindegyikben n darab különböző egész számot tárolunk. Adjon $O(n \log n)$ lépésszámú algoritmust a két tömb legkisebb közös elemének megtalálására, vagy ha nincs ilyen elem, akkor ennek jelzésére.

6. Egy szomszédossági mátrixával adott n csúcsú, irányított G gráfban kettő csúcs, u_1 és u_2 kékre van színezve, a többi csúcs színtelen. Adott továbbá két kijelölt színtelen csúcs, s és t és s-ből szeretnénk t-be eljutni a gráf éleit használva úgy, hogy közben mindkettő kék csúcson áthaladunk. Adjon $O(n^2)$ lépésszámú algoritmust, ami meghatározza, hogy ehhez legkevesebb hány élen kell áthaladnunk (vagy ha az eljutás nem lehetséges, akkor ez derüljön ki).

Algoritmusok és gráfok - Pótzárthelyi
2019. december 5.

A válaszokat indokolni kell. Hivatkozni csak az előadáson tanultakra lehet.

1. Igaz-e, hogy ha egy algoritmus lépésszáma $42 \cdot n^2 \log n + 2020 \cdot (n-1)^3 - 7 \cdot \sqrt{n} + 30$, akkor az algoritmus lépésszáma $O(n^3)$? Ha úgy véli, hogy ez igaz, akkor megfelelő c konstans és n_0 küszöbérték megadásával lássa ezt be, ha pedig úgy véli, hogy hamis, akkor bizonyítsa be ezt.

2. A $2, 10, 1, 3, 7, 8, 4$ tömb rendezése során előállhat-e az $1, 2, 3, 10, 7, 8, 4$ helyzet, ha
   (a) buborékrendezést használunk?
   (b) beszúrásos rendezést használunk?

3. Egy kezdetben üres, 11 méretű hash táblába nyílt címzéssel, lineáris próbával szúrtunk be nyolc egész számot, a beszúrások sorrendje nem ismert. A használt hash függvény értéke a K kulcs esetén K maradéka 11-gyel osztva. Csak beszúrások történtek, törlés nem volt és az alábbi táblát kaptuk.

0	1	2	3	4	5	6	7	8	9	10
	13	2	3	5	17	7	18			21

   Milyen sorrendben történhetett a $3, 5, 17, 7, 18$ elemek beszúrása, ha ezt a táblát kaptuk? Az összes lehetséges sorrendjét adja meg ennek az öt elemnek.

4. Egy hét csúcsú bináris keresőfát posztorder bejárással bejárva a csúcsokat $1, 10, 11, 8, 20, 13, 3$ sorrendben látogatjuk meg. Adja meg, hogy hogyan néz ki ez a fa.

5. Adott egy n hosszú tömb, mely csupa különböző egész számot tartalmaz. Adjon $O(n \log n)$ lépésszámú algoritmust, ami vagy talál a tömbben egy olyan k egész számot, melyre sem $k-1$, sem $k+1$ nincs a tömbben vagy jelzi, ha nincs ilyen k szám.

6. Egy szomszédossági mátrixával adott n csúcsú, irányított G gráfban néhány csúcs kékre van színezve, a többi csúcs színtelen. A kék csúcsok egy, a csúcsokkal indexelt K tömbben adottak úgy, hogy $K[v]$ értéke 1, ha a v csúcs kék, egyébként pedig $K[v] = 0$. Adott továbbá két kijelölt színtelen csúcs, s és t is. Adjon $O(n^2)$ lépésszámú algoritmust, ami eldönti, hogy el lehet-e jutni s-ből t-be kék csúcs érintése nélkül.

Algoritmusok és gráfok - Zárthelyi
2019. november 21.
Megoldások

A válaszokat indokolni kell. Hivatkozni az előadáson tanultakra lehet.

1. Az alábbi pszeudokód inputja egy $n \geq 2$ szám, a pszeudokódban lépésnek egy darab * kiírása és az értékadás számít. Mutassa meg, hogy a pszeudokód által megadott algoritmus lépésszáma $O(n^2)$.

```
ciklus j = 0-tól (n-1)-ig:
    k  := 0
    ciklus amíg k < j:
        kiírunk egy  darab *-ot
        k := k + 1
    ciklus vége
ciklus vége
```

Megoldás

A belső ciklus (az amíg-ciklus) magja j-szer fut le, mert minden lefutáskor nő eggyel a k értéke, ami 0-tól indul és $j-1$-re fut le utoljára. Mivel $j \leq n$, ezért a belső ciklus magja legfeljebb n-szer fut le.
A belső ciklus magja 2 lépés, azaz konstans, így a belső ciklus egésze $O(n)$-es.

A külső ciklus ciklusmagja egy utasításból, majd az előbb látott belső ciklusból áll, vagyis a külső ciklus ciklusmagja $1 + O(n)$, azaz $O(n)$ lépés.
A kód maga ez a külső ciklus, aminek $O(n)$-es magja n-szer fut le, így a teljes lépésszám $O(n^2)$.

2. A $12, x, 2, 9, 10, 5$ tömböt kiválasztásos rendezéssel és buborékrendezéssel is rendeztük. Adja meg x összes lehetséges értékét, ha tudjuk, hogy x a tömbben máshol elő nem forduló egész szám és a kiválasztásos rendezés során volt $2, 5, 12, 9, 10, x$ átmeneti állapot, a buborékrendezés során pedig $2, x, 9, 5, 10, 12$ állapot.

Megoldás

A kiválasztásos rendezés először a tömb legkisebb elemét keresi meg és egy cserével a tömb elejére mozgatja, így kerül a 2 az első helyre és a $2, x, 12, 9, 10, 5$ tömböt kapjuk (eközben kiderült, hogy $x > 2$, mert különben x került volna az első helyre és nem a 2). Ezután a maradék elemek közüli legkisebbet cserével a tömb második cellájába visszük, így kerül oda az 5, a kapott tömb $2, 5, 12, 9, 10, x$. Ekkor kiderül, hogy $x > 5$, mert különben az x került volna a második helyre és nem az 5. Több infót ebből a rendezésből nem tudunk meg, mert elértük a közbülső állapotot, tehát eddig annyit tudunk, hogy $x > 5$.

A buborékrendezés futása során először a 12 a tömb végére kerül, eközben a 12-t meg kellett cserélnünk minden őt követő elemmel, vagyis az első kör után a tömb állapota biztosan $x, 2, 9, 10, 5, 12$. Ezután a 10 a tömb hátulról második helyére

kerül, eközben a 2-t és az x-et megcseréltük (hiszen $x > 5$ a kiválasztásos miatt), de x már nem cserélődött fel 9-cel, vagyis $x < 9$. Mivel itt is elértük a közbülső állapotot több infó nem derül ki, csak az, hogy $x < 9$

A lehetséges értékek tehát $x = 6, 7, 8$.

3. Egy kezdetben üres, 11 méretű hash táblába nyílt címzéssel, lineáris próbával szúrtunk be kilenc egész számot. A használt hash függvény értéke a K kulcs esetén K maradéka 11-gyel osztva. A beszúrások után egyetlen számot kitöröltünk a táblából, de a törölt cellát véletlenül nem állítottuk töröltre, üresnek látszik. Az alábbi táblát kaptuk, amiben tehát a törlés helye nem látszik. Melyik cellában történt a törlés?

0	1	2	3	4	5	6	7	8	9	10
	13	2	28	15		7	18		21	10

Megoldás

A törlés az 5-ös indexű cellában történt, mert a 28-as számra a hash függvény értéke 6, azaz a 28 a 6-os indexű cellába került volna és csak úgy kerülhetett a 3-as indexűbe, ha a beszúráskor mind a 6-os, mind az 5-ös és a 4-es indexű cella is foglalt volt.

4. Egy öt csúcsú AVL-fát preorder bejárással bejárva a csúcsokat $9, 5, x, 8, 12$ sorrendben látogatjuk meg. Adja meg, hogy hogyan néz ki ez a fa és adja meg x összes lehetséges értékét, ha tudjuk, hogy x egy olyan pozitív egész szám, ami máshol nem fordul elő a fában.

Megoldás

A preorder bejárás a fa gyökerét látogatja meg először, vagyis a gyökér a 9. A bal fába kerülnek a 9-nél kisebb értékek a bináris keresőfa tulajdonság miatt, vagyis 5 és 8 biztosan, de ha 8 balra kerül, akkor x is, mert a preorder bejárásnál a teljes bal fát előbb látjuk, mint a jobb fát. Vagyis a jobb fában csak a 12 van, a balban pedig $5, x, 8$.

A jobb részfa magassága tehát 1 és mivel ez egy AVL-fa, ezért a bal fa magassága legfeljebb 2 lehet, vagyis az $5, x, 8$ csúcsok úgy helyezkednek el, hogy egyikük a gyökér, a másik kettő meg ennek a jobb és bal gyereke.

Mivel a preorder bejárás az 5-öt látja meg az $5, x, 8$ közül elsőnek, ezért ez van középen és akkor x ennek a bal, 8 pedig a jobb gyereke.

A fából a bináris keresőfa tulajdonság miatt látjuk, hogy x olyan elem, ami 5-nél kisebb pozitív egész vagyis $x = 1, 2, 3, 4$ lehet.

5. Adott két n elemű tömb, mindegyikben n darab különböző egész számot tárolunk. Adjon $O(n \log n)$ lépésszámú algoritmust a két tömb legkisebb közös elemének megtalálására, vagy ha nincs ilyen elem, akkor ennek jelzésére.

Egy lehetséges megoldás

Algo:

(a) Rendezzük mindkét tömböt összefésüléses rendezéssel.

(b) Menjünk végig az első tömbön és a tömb minden elemére csináljunk egy bináris keresést a második tömbben (nézzük meg, hogy az első tömb aktuális eleme benne van-e a második tömbben). Amelyik értéknél ez először előfordul, az a legkisebb közös elem, ha egyszer sem fordul elő, akkor nincs közö elemük.

Jóság: A rendezés után a bináris keresés használható, hogy az első tömb elemeiről eldöntsük, hogy benne vannak-e a másodikban, mert a második tömb rendezett. Mivel minden elemet megnézek az első tömbből, ezért biztos észreveszem, ha van egyezés és mivel az első tömb elemeit nagyság szerinti sorban nézzük, ezért a legkisebb közöset találjuk meg először.

Lépésszám: A két rendezés $O(n \log n)$ és $O(n \log n)$, ez együtt is $O(n \log n)$, ezután pedig n darab bináris keresést csinálok egy n méretű tömbben, azaz n-szer futtatok egy $O(\log n)$-es eljárást, ez is $O(n \log n)$, együtt is $O(n \log n)$.

6. Egy szomszédossági mátrixával adott n csúcsú, irányított G gráfban kettő csúcs, u_1 és u_2 kékre van színezve, a többi csúcs színtelen. Adott továbbá két kijelölt színtelen csúcs, s és t és s-ből szeretnénk t-be eljutni a gráf éleit használva úgy, hogy közben mindkettő kék csúcson áthaladunk. Adjon $O(n^2)$ lépésszámú algoritmust, ami meghatározza, hogy ehhez legkevesebb hány élen kell áthaladnunk (vagy ha az eljutás nem lehetséges, akkor ez derüljön ki).

Megoldás

Ötlet: Két lehetőség van arra, hogy s-ből t-be eljussunk a két kék csúcsot érintve: s-ből elmegyek a u_1-be, onnan u_2-be, onnan t-be vagy s-ből u_2-be megyek, onnan u_1-be, onnan t-be (vagyis az a két lehetőség, hogy a két kék csúcsot milyen sorrendben látogatom meg). Az itt felmerülő $s-u_1, s-u_2, u_1-u_2, u_1-t, u_2-u_1, u_2-t$ távolságokat (a legkevesebb élből álló utak hosszát, azaz élszámát) kell tehát meghatározni és így megkapom a két lehetőséghez tartozó hosszakat, ebből a kisebb lesz a keresett érték.

Algo: Szélességi bejárást futtatok s-ből (így meglesz a távolság s-ből u_1-be és u_2-be), u_1-ből (így meglesz a távolság u_1-ből u_2-be és t-be) és u_2-ből (így meglesz a távolság u_2-ből u_1-be és t-be). Ebből a hat értékből kiszámolom a fenti két lehetőséghez tartozó legrövidebb séta hosszát és a kisebbik lesz a keresett szám.

Jóság: Csak ez a két lehetőség van, tehát elég ezt a hat rész-távolságot meghatározni, ezeket pedig a szélességi bejárás megadja (tanultuk).

Lépésszám: Egy bejárás $O(n^2)$, ezt kell lefuttatni háromszor, az $3 \cdot O(n^2)$, vagyis $O(n^2)$, utána már konstans sok munka a két összeg meghatározása.

Algoritmusok és gráfok - Pótzárthelyi
2019. december 5.

A válaszokat indokolni kell. Hivatkozni csak az előadáson tanultakra lehet.

1. Igaz-e, hogy ha egy algoritmus lépésszáma $42 \cdot n^2 \log n + 2020 \cdot (n-1)^3 - 7 \cdot \sqrt{n} + 30$, akkor az algoritmus lépésszáma $O(n^3)$? Ha úgy véli, hogy ez igaz, akkor megfelelő c konstans és n_0 küszöbérték megadásával lássa ezt be, ha pedig úgy véli, hogy hamis, akkor bizonyítsa be ezt.

   Megoldás
   Igaz az állítás, az $42 \cdot n^2 \log n + 2020 \cdot (n-1)^3 - 7 \cdot \sqrt{n} + 30$ függvény $O(n^3)$-ös. Ehhez egy olyan c pozitív konstanst és egy olyan n_0 küszöbindexet kell találnunk, melyre igaz, hogy

   $$42 \cdot n^2 \log n + 2020 \cdot (n-1)^3 - 7 \cdot \sqrt{n} + 30 \leq c \cdot n^3, \text{ha } n \geq n_0$$

   Ilyen c és n_0 értéket például az alábbi egyenlőtlenség-sorozattal tudunk találni:
   $42 \cdot n^2 \log n + 2020 \cdot (n-1)^3 - 7 \cdot \sqrt{n} + 30 \leq 42 \cdot n^2 \log n + 2020 \cdot (n-1)^3 + 30$
   mert a negatív tagot el lehet hagyni, tetszőleges $n \geq 1$ esetén
   $42 \cdot n^2 \log n + 2020 \cdot (n-1)^3 + 30 \leq 42n^3 + 2020 \cdot n^3 + 30n^3$
   mert $\log n \leq n$, $n - 1 \leq n$ és $30 \leq 30n^3$ tetszőleges $n \geq 1$ esetén.
   Azt kaptuk tehát, hogy
   $42 \cdot n^2 \log n + 2020 \cdot (n-1)^3 - 7 \cdot \sqrt{n} + 30 \leq 42n^3 + 2020 \cdot n^3 + 30n^3 = (42 + 2020 + 30)n^3$
   tetszőleges $n \geq 1$ esetén, vagyis $c = 42 + 2020 + 30 = 2092$ és $n_0 = 1$ jó választás.

2. A $2, 10, 1, 3, 7, 8, 4$ tömb rendezése során előállhat-e az $1, 2, 3, 10, 7, 8, 4$ helyzet, ha
   (a) buborékrendezést használunk?
   (b) beszúrásos rendezést használunk?

   Megoldás
   (a) A buborékrendezésnél ez nem állhat elő. A buborékrendezés során először végigmegyünk az egész tömbön és a szomszédos elemeket hasonlítjuk: ha egy nagyobb elem egy kisebb elem előtt áll közvetlenül, akkor megcseréljük őket. Így az első körben a 10-et megcseréljük az 1, majd a 3 értékkel, de eközben a tömb elején a többi elem sorrendje nem változik, vagyis mikorra a 10 elér a 4. pozícióba a tömb állapota $2, 1, 3, 10, 7, 8, 4$ lesz, azaz eddig nem állhatott elő a kívánt átmeneti helyzet, mert a 2 nem léphette még át az 1-et. Ha viszont tovább folytatjuk a rendezést, akkor a 10 továbbmegy jobbra és sose lesz már a 4. pozícióban vagyis a későbbiekben sem állhat elő ez a helyzet.
   (b) A beszúrásos rendezést futtatva az első két elem rendezetten áll, az első érdemi történés az, amikor a 3. pozícióban álló 1-et a helyére mozgatjuk (balra haladva, mindaddig cserélve, amíg nála nagyobb elem áll előtte), így az $1, 2, 10, 3, 7, 8, 4$ tömböt kapjuk. Ekkor a 3-at mozgatjuk balra, cserélve az előtte levővel, ha az nagyobb nála és az $1, 2, 3, 10, 7, 8, 4$ tömböt kapjuk, a kívánt átmeneti helyzetet, vagyis ennél az algoritmusnál lehetséges ez az átmeneti állapot.

3. Egy kezdetben üres, 11 méretű hash táblába nyílt címzéssel, lineáris próbával szúrtunk be nyolc egész számot, a beszúrások sorrendje nem ismert. A használt hash függvény értéke a K kulcs esetén K maradéka 11-gyel osztva. Csak beszúrások történtek, törlés nem volt és az alábbi táblát kaptuk.

0	1	2	3	4	5	6	7	8	9	10
	13	2	3	5	17	7	18			21

Milyen sorrendben történhetett a $3, 5, 17, 7, 18$ elemek beszúrása, ha ezt a táblát kaptuk? Az összes lehetséges sorrendjét adja meg ennek az öt elemnek.

Megoldás

Az 5-öt biztosan később szúrtuk be, mint a 17-et, mert különben az 5 az 5-ös indexű cellába került volna, mivel nem ott van, ezért annak akkor már foglaltnak kellett lennie. Hasonló érveléssel látható, hogy a 17-et biztos később szúrtuk be, mint a 7-et, mert különben a 17 a 6-os indexű cellában lenne. Ugyanígy látható, hogy a 7–et a 18 után szúrtuk be, mert különben a 7 a 7-es indexű cellában lenne. Az derült tehát ki, hogy az 5 később jött, mint a 17, az később jött, mint a 7, az pedig később jött, mint a 18, vagyis ezen négy elem sorendje biztosan $18, 7, 17, 5$ volt.

A 3 beszúrása ezekhez képest bármikor történhetett, mert a 3 a hash függvény szerinti helyére került és ez a hely teljesen független a többi elem által bejárt celláktól, a 3 beszúrása nem befolyásolja a másik négy elem sorsát.

Vagyis a lehetséges sorrendek: $3, 18, 7, 17, 5$ vagy $18, 3, 7, 17, 5$ vagy $18, 7, 3, 17, 5$ vagy $18, 7, 17, 3, 5$ vagy $18, 7, 17, 5, 3$.

4. Egy hét csúcsú bináris keresőfát posztorder bejárással bejárva a csúcsokat $1, 10, 11, 8, 20, 13, 3$ sorrendben látogatjuk meg. Adja meg, hogy hogyan néz ki ez a fa.

Megoldás

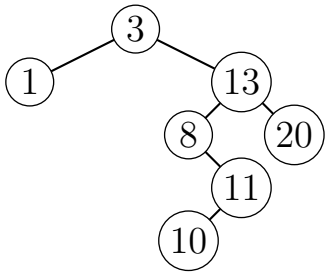
A posztorder bejárásban a gyökeret látogatjuk meg utoljára, ezért a fa gyökere a 3 és a bináris keresőfa tulajdonság miatt tőle balra csak az 1 van, a $8, 10, 11, 13, 20$ értékek a jobb fában vannak.

A jobb fa elemei közül a posztorder bejárás a 13-at járja be utoljára, vagyis ez a gyökér a jobb részfában és a bináris keresőfa tulajdonság miatt a jobb részfájában csak a 20 van, a balban pedig $8, 10, 11$.

Az most a kérdés, hogy milyen elrendezésben vannak a $8, 10, 11$ elemek a 13 bal részfájában. Ezt onnan tudjuk kideríteni, hogy a posztorder bejárás ezen értékek közül a 8-at látja utoljára, vagyis ez a gyökér és a bináris keresőfa tulajdonság miatt ennek bal gyereke nincs, a jobb fájában pedig 10 és 11 van.

Az most már csak a kérdés, hogy a 10 és 11 hogyan állnak ebben a jobb fában, de ezt megint csak a posztorder bejárással tudjuk kitalálni, mivel a posztorder bejárás a 11-et látja később, ezért ez a gyökér, ennek pedig bal gyereke lesz a 10.

A kapott fa tehát ez:



5. Adott egy n hosszú tömb, mely csupa különböző egész számot tartalmaz. Adjon $O(n \log n)$ lépésszámú algoritmust, ami vagy talál a tömbben egy olyan k egész számot, melyre sem $k-1$, sem $k+1$ nincs a tömbben vagy jelzi, ha nincs ilyen k szám.

   Egy lehetséges megoldás

   Algo:

   (a) Rendezzük a tömböt összefésüléses rendezéssel.

   (b) Menjünk végig a rendezett tömbön és a tömb minden $A[i]$ elemére nézzük meg, hogy előtte $A[i]-1$, utána pedig $A[i]+1$ van-e. Ha találunk olyan i indexet, ahol $A[i]$ előtt nem $A[i]-1$, utána pedig nem $A[i]+1$ van, akkor $A[i]=k$ olyan szám, amit keresünk, egyébként pedig nincsen k szám.

   Jóság: A rendezés után az egymást követő számok egymás után lesznek a tömbben, vagyis ha a rendezett sorban egy szám előtt közvetlenül nem a nála eggyel kisebb szám áll, akkor az eggyel kisebb szám nincs a tömbben, hasonlóan ha egy szám után közvetlenül nem a nála eggyel nagyobb szám áll, akkor az eggyel nagyobb szám nincs a tömbben.

   Lépésszám: A rendezés $O(n \log n)$, utána pedig minden i értékre két ellenőrzést kell tennünk, ami $O(n)$, így az algoritmus $O(n \log n)$ lépésszámú.

6. Egy szomszédossági mátrixával adott n csúcsú, irányított G gráfban néhány csúcs kékre van színezve, a többi csúcs színtelen. A kék csúcsok egy, a csúcsokkal indexelt K tömbben adottak úgy, hogy $K[v]$ értéke 1, ha a v csúcs kék, egyébként pedig $K[v]=0$. Adott továbbá két kijelölt színtelen csúcs, s és t is. Adjon $O(n^2)$ lépésszámú algoritmust, ami eldönti, hogy el lehet-e jutni s-ből t-be kék csúcs érintése nélkül.

   Megoldás

   Ötlet: Pontosan akkor lehet eljutni s-ből t-be kék csúcs érintése nélkül, ha van s-ből t-be olyan út, melyen minden csúcs színtelen.

   Algo:

   1. Módosítsuk a gráf szomszédossági mátrixát úgy, hogy minden olyan élet kitörlünk belőle, aminek legalább egy végpontja kék. Ezt úgy lehet megtenni, hogy végigmegyünk a mátrixon soronként, soron belül oszloponként és ha $A[i,j]=1$ (azaz van él az i

csúcsból a j csúcsba) és vagy $K[i] = 1$ vagy $K[j] = 1$, akkor $A[i, j] = 0$-t állítunk be (azaz töröljük az élet a mátrixból).
Pszeudokódban ugyanez:

```
ciklus i =1-tól n-ig:
    ciklus j= 1-tól n-ig:
        ha A[i,j] == 1:
            ha (K[i]  == 1 vagy  K[j]  == 1):
                A[i,j] := 0
    ciklus vége
ciklus vége
```

2. BFS-t vagy DFS-t futtatunk a kapott mátrixon s-ből és ha t bejárt lesz a végén, akkor van jó út, különben meg nincsen.

Jóság: Az új gráfban pontosan azok az utak maradnak meg, amik nem használnak kék csúcsokat, mert a kék csúcsokba se bemenni, se kijönni belőlük nem lehet és a BFS/DFS el tudja dönteni, hogy az új gráfban van-e út s-ből t-be.

Lépésszám: A gráf módosítása $O(n^2)$, mert a belső ciklus magja konstans lépés és n-szer fut le, azaz a belső ciklus $O(n)$-es, a külső ciklus magja ez az $O(n)$-es belső ciklus és n-szer fut le, vagyis $O(n^2)$ összesen.
A BFS/DFS futtatása $O(n^2)$, mert az új gráfnak is n csúcsa van (csak éleket töröltünk.)