Algoritmuselmélet pótzárthelyi
2012. április 25.

A rendelkezésre álló munkaidő 100 perc.
Kérjük, minden résztvevő **nevét, NEPTUN kódját**, a dolgozat *minden* lapjának jobb felső sarkában *olvashatóan* és *helyesen* tüntesse fel. Ezen kívül a legfelső lapra írja rá **gyakorlatvezetője nevét** is (akihez a NEPTUN szerint jár).
Minden egyes feladat helyes megoldása 10 pontot ér. A dolgozatok értékelése: 0-31 pont: 1, 32-43 pont: 2, 44-55 pont: 3, 56-67 pont: 4, 68-80 pont: 5. A puszta (indoklás nélküli) eredményközlést nem értékeljük. A megindokolt részeredményért arányos pontszám jár. Az évvégi jegy kiszámításakor a (legalább elégséges) zh pontszámát vesszük figyelembe.
Írószeren és papírokon kívül semmilyen segédeszköz használata sem megengedett, így tilos az írott vagy nyomtatott jegyzet, a számoló- és számítógép ill. mobiltelefon használata, továbbá a dolgozatírás közbeni együttműködés.
Az eredményeket a gyakorlatvezetőtől lehet majd megtudni.

1. Tudjuk, hogy az $f(n), g(n) : \mathbb{N} \to \mathbb{N}$ függvényekre igaz, hogy $f(n) = O(g(n))$, de $f(n) \neq \Theta(g(n))$. Lehetséges-e, hogy $f(n) + 2012 \cdot n^{\log n} = \Theta(n \cdot \log n + g(n))$?

Egy lehetséges megoldás:
„lehetséges-e": vagy belátjuk, hogy lehetséges (adunk példát), vagy bizonyítjuk, hogy sosem igaz
Az $f(n) = n \cdot \log n$ és $g(n) = 2012 \cdot n^{\log n}$ választás a feladatnak megfelelő állítás, mert:

(a) $f(n) = O(g(n))$, hiszen $n \cdot \log n = O(n^2)$, és - ha $n \geq 4$ - akkor már $n^2 = O(2012 \cdot n^{\log n})$ is teljesül

(b) $f(n) \neq \Theta(g(n))$, hiszen $n \cdot \log n \neq \Omega(n^2)$, de $2012 \cdot n^{\log n} = \Omega(n^2)$

(c) $f(n) + 2012 \cdot n^{\log n} = \Theta(n \cdot \log n + g(n))$, hiszen a két oldal teljesen megegyezik, és bármely $a(n)$-re $a(n) = \Theta(a(n))$

Tehát lehetséges.

---

2. Egy n-szer n-es táblázat minden kockájába egy (nem feltétlenül pozitív) egész szám van írva vagy rózsaszínnel vagy sárgával. Sétát szeretnénk tenni a táblázatban a következő feltételekkel: a séta bárhol indulhat és bárhol végződhet, egy lépésben vagy balra vagy felfelé léphetünk egy mezőt, de csak akkor ha közben színt is váltunk. Egy ilyen séta értéke a közben érintett mezőkön levő számok összege (a végponti mezők is számítanak). A séta állhat egy mezőből is, azaz végződhet ott, ahol kezdődik. Adjon algoritmust, ami $O(n^2)$ lépésben meghatározza a táblázatban található legértékesebb séta hosszát.

Egy lehetséges megoldás:
Legyen $c_{i,j}$ az eredeti táblázat $i.$ sorának $j.$ eleme.
Töltsünk ki egy $n \times n$-es $\mathcal{A}$ táblázatot, ahol $\mathcal{A}_{i,j}$ jelöli az $i.$ sor $j.$ eleméhez érve elérhető maximális értéket.
$\mathcal{A}_{n,n} := c_{n,n}$, hiszen a jobb alsó mezőben befejezve nem játhattunk máshol
Alulról felfelé, jobbról balra haladva:
$\mathcal{A}_{i,j} := \max\{\mathcal{A}_{i,j+1}, \mathcal{A}_{i+1,j}, 0\} + c_{i,j}$, a maximumképzésben pedig csak azokat vesszük figyelembe, ahol az adott $\mathcal{A}$-beli mező létezik, és a színe a (i, j) mezőétől különböző.
A megoldás az $\mathcal{A}$ táblázat maximuma lesz.
Az algoritmus lépésszáma $O(n^2)$, mert egy n^2 méretű táblázatot töltünk ki úgy, hogy bármely mező kiszámítása konstans lépést igényel, tehát $O(n^2)$ - emellett ugyanebben a táblázatban maximumkeresést csinálunk, ami szintén $O(n^2)$. Tehát összességében is jó a lépésszám.
Az algoritmus jó, mert kisebb eredményt nem tud adni, hiszen bármely jó séta értékét összegezzük ily módon. Emellett nagyobb eredményt sem kaphatunk, mert amit számláláskor megkapunk, mint eredmény, azon az algoritmust visszakövetve megkapjuk a hozzá tartozó sétát (tehát van olyan séta).

---

3. Űrhajónkkal egy véges, egydimenziós univerzumban ragadtunk, amelynek pontjait 0 és r közötti egész koordinátákkal azonosítjuk. A közlekedést a $0 \leq x_1, x_2, \ldots, x_n \leq r$ koordinátákon elhelyezett *tükörteleportok* segítik: egy-egy tükörteleport aktiválásakor űrhajónk pozíciója tükröződik a teleport pozíciójára, azaz például az y koordinátáról a $2x_i - y$ koordinátára jutunk (vagy megsemmisülünk, ha ezzel kilépnénk az univerzumból). Az s koordinátáról indulva szeretnénk a t koordinátán található – menekülést jelentő – féregjárathoz jutni. Adjon $O(r^2)$ lépésszámú algoritmust, amely meghatározza, hogy ehhez legkevesebb hányszor kell teleportálnunk!

Egy lehetséges megoldás:
Egydimenziós, véges, 0 és r közötti egész koordinátákkal azonosítjuk. Tehát az univerzum (maximum) $r + 1$ darab pontból áll.
Vegyünk fel egy G gráfot, amely csúcsai az univerzum pontjai, élei pedig aszerint adódnak, hogy az adott csúcsból tükörteleport segítségével átléphetünk-e a másikba.
Indítsunk egy BFS-t a kapott gráf s csúcsából, és figyeljük meg, hogy el tudunk-e elakadás és új komponensből indulás nélkül jutni t-be. Ha el tudunk jutni, akkor a BFS ezen tulajdonságát (súlyozatlan gráfban legrövidebb út keresésére alkalmas) felhasználva tudhatjuk, hogy ahányadik szinten megtaláltuk a t csúcsot, az a legrövidebb utat adja meg.
Ez jó mert bármely útvonal a gráfban is egy útnak felel meg, és fordítva. Tehát a legrövidebb út keresése BFS-sel egy legrövidebb teleportálást ad meg.
Lépésszáma $O(r^2)$ az alábbiak miatt:
A gráfot éllistájával felvenni $O(r^2)$, hiszen csúcsokból $r + 1 = O(r)$ darab van, a tükörteleportokból pedig (mivel az univerzum $O(r)$ pontú) $O(r)$ van, és bármelyiktől vagy balra vagy jobbra $O(r/2)$ van, tehát legfeljebb $O(r^2)$ élről van szó.
A gráfban BFS-t futtatni $O(n + e)$ költségű, tehát $O(r^2)$.

---

4. Egy irányított gráf csúcshalmaza $\{A, B, C, D, E, F\}$, az élek és súlyaik pedig az alábbiak: $s(A, B) = 2$, $s(A, C) = 7$, $s(A, D) = 3$, $s(A, F) = 4$, $s(B, C) = 4$, $s(C, E) = 3$, $s(D, B) = x$, $s(D, E) = 2$, $s(E, F) = 4$, $s(F, C) = 2x$.
Az x nemnegatív valós paraméter függvényében határozza meg az A csúcsból a többi csúcsba vett legrövidebb utak hosszát a Dijkstra algoritmussal !

Egy lehetséges megoldás:

	B	C	D	E	F	KÉSZ
1.	**2**	7	3	∞	4	$\{A\}$
2.	**2**	6	**3**	∞	4	$\{A, B\}$
3.	**2**	6	**3**	5	**4**	$\{A, B, D\}$
4.	**2**	$\min\{6,4+2x\}$	**3**	5	**4**	$\{A, B, D, F\}$
ha $x \leq 1/2$: 5.	**2**	**4+2**x	**3**	5	**4**	$\{A, B, C, D, F\}$
ha $x \leq 1/2$: 6.	**2**	**4+2**x	**3**	**5**	**4**	$\{A, B, C, D, E, F\}$
ha $x \geq 1/2$: 5.	**2**	$\min\{6,4+2x\}$	**3**	**5**	**4**	$\{A, B, D, E, F\}$
ha $x \geq 1/2$: 6.	**2**	$\mathbf{\min\{6,4+2}x\mathbf{\}}$	**3**	**5**	**4**	$\{A, B, C, D, E, F\}$

Ha $x = 1/2$, akkor a választásunktól függ, merre is jutunk.

---

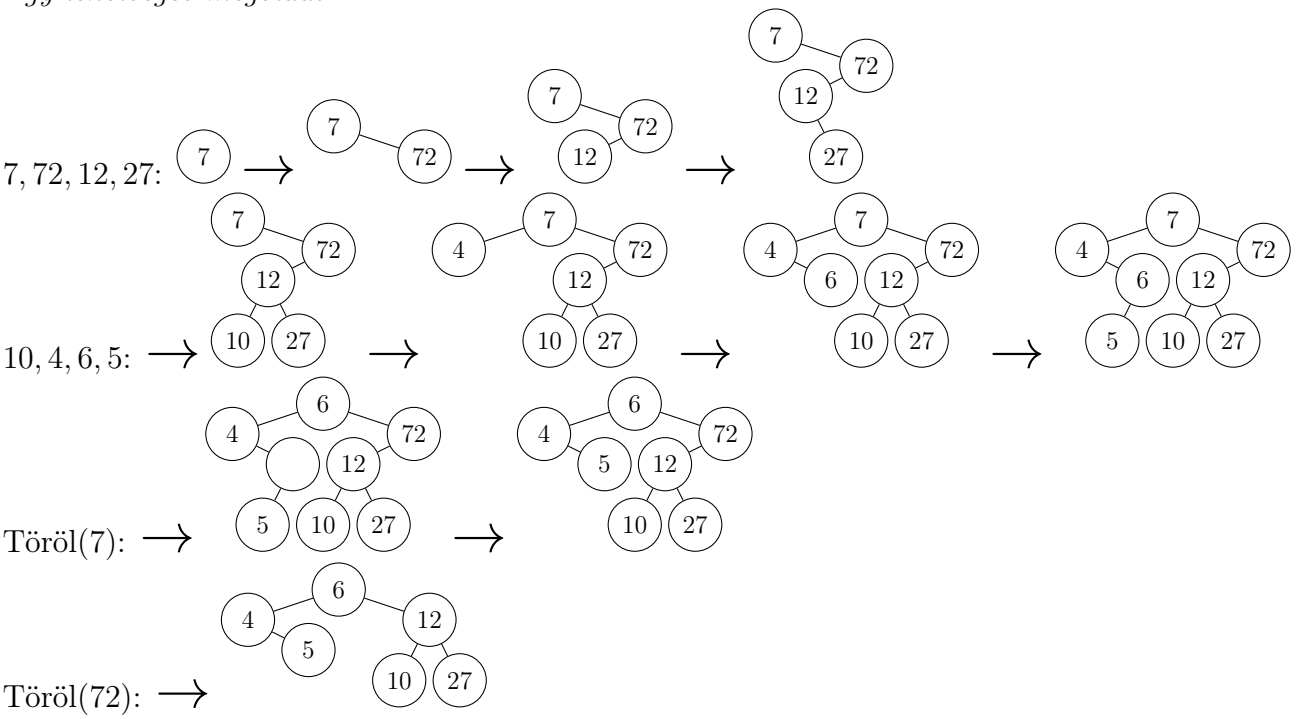
5. Egy n-szer n-es táblázatban az $1, 2, \ldots, n^2$ egész számok vannak valamilyen sorrendben (az n^2 szám mindegyike pontosan egyszer szerepel). Szeretnénk ezeket a számokat rendbe rakni, úgy, hogy az i. sorban $((i-1)n+1)$-től in-ig legyenek növekvően a számok, minden $1 \leq i \leq n$ esetén. A számokat csak egyféleképpen tudjuk mozgatni: két (nem feltétlenül szomszédos) elemet felcserélhetünk.
(a) Adjon $O(n^2)$ cserét használó algoritmust a feladat megoldására! (A cserén kívül bármi mást korlátlanul szabad csinálnunk.)
(b) Létezik-e olyan algoritmus, ami $O(n \log n)$ cserével megoldja a feladatot?

Egy lehetséges megoldás:
(a) keressük meg a legkisebb nem helyrerakott elemet és tegyük a helyére, majd folytassuk így tovább a maradékon. Ez $O(n^2)$ cserét igényel, hiszen minden elemet legfeljebb egyszer cserélünk (akkor a helyére), kikeresni pedig nem kerül cserébe.
(b) nem lehetséges, hiszen egy cserével legfeljebb két elemet kerülhet a helyére, emiatt ha egyik elem sincs a helyén kezdetben, akkor kell legalább $\left\lceil \frac{n^2}{2} \right\rceil$ csere, ami nem $O(n \log n)$.

---

6. Egy kezdetben üres bináris keresőfába szúrja be az alábbi számokat az alábbi sorrendben: $7, 72, 12, 27, 10, 4, 6, 5$, majd törölje ki a 7-t, aztán pedig a 72-t.
Minden lépés után adja meg az aktuális állapotot.

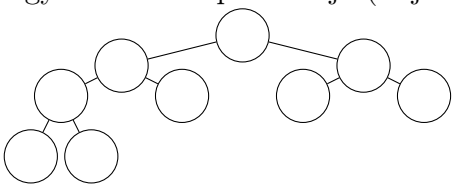
Egy lehetséges megoldás:



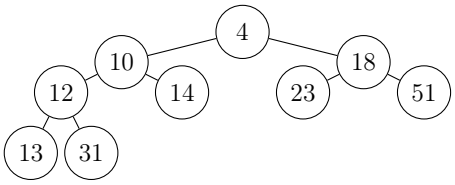
---

7. Egy kupac elemeit inorder bejárás szerint kiolvasva a $13, 12, 31, 10, 14, 4, 23, 18, 51$ sorozatot kapjuk. Rajzolja fel a kupacot!

Egy lehetséges megoldás:
Egy 9-elemű kupac alakja (teljes bináris fa):

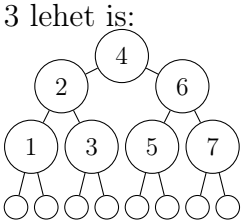


A kupacban a legkisebb elem a gyökér, tehát a 4.
Inorder: inorder(bal(x)), x, inorder(jobb(x))
Tehát a 4-estől balra lévő elemek vannak a bal részfában a jobbra lévők a jobb részfában.
A bal részfa minimum és gyökere tehát 10, attól balra lévők minimuma a 12, ez van tőle balra.
A 10-től jobbra csak a 14 lehet.
A 4-estől jobbra a 18 van, mert az a legkisebb abban a részfában.
Tehát a kupac:

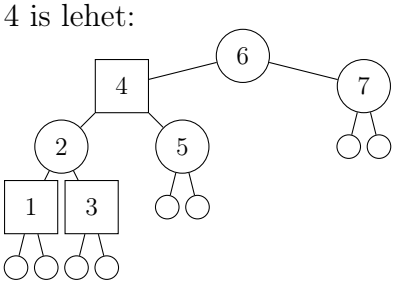


8. Mekkora lehet egy olyan piros-fekete fa magassága, amiben 7 elemet tárolunk?

Egy lehetséges megoldás:
A magasság a gyökértől a levélig vezető úton az élek száma.
A magasság legalább 3, hiszen 2 magasságú fában nem fér el, csak 3 elem.
3 lehet is:



4 is lehet:



4-nél több nem lehet, mert akkor a leghosszabb ágon legalább 5 él kellene legyen, emiatt pedig (mivel a levél és a gyökér is fekete, legfeljebb 2 piros csúcs lehet rajta (2. és 4. vagy 2. és 5. vagy 3. és 5.), tehát kell legyen legalább 3+1 (3 belső csúcs, plusz 1 levél) fekete csúcs. Azaz, mivel a feketemagasság minden irányba ugyanannyi, bármely ágon kell legyen 3+1 fekete csúcs, ez pedig azt jelenti, hogy legalább 7 fekete elem van a fa belsejében. Ez pedig ellentmondás, hiszen akkor nem lehet piros csúcs már, ha 7 elemet tárolunk, hisz mind fekete. Tehát 4-nél több nem lehet a magasság.