# Algoritmuselmélet 1. pótZH

| | |
|---|---|
| A rendelkezésre álló munkaidő 90 perc. Minden megoldást indokoljon! | 2024. április 30. |
| Minden feladat egységesen 10 pontot ér. Az aláírás megszerzéséhez minimum 24 pontot kell elérni. | |
| Kérjük, minden résztvevő írja fel a **nevét, NEPTUN kódját és a gyakorlatvezető nevét** az összetűzött lapok jobb felső sarkába. | |

**Általános alapelvek.** A pontozási útmutató célja, hogy a javítók a dolgozatokat egységesen értékeljék. Ezért az útmutató minden feladat (legalább egy lehetséges) megoldásának főbb gondolatait és az ezekhez rendelt részpontszámokat közli. Az útmutatónak nem célja a feladatok teljes értékű megoldásának részletes leírása; a leírt lépések egy maximális pontszámot érő megoldás vázlatának tekinthetők. Az útmutatóban feltüntetett részpontszámok csak akkor járnak a megoldónak, ha a kapcsolódó gondolat egy áttekinthető, világosan leírt és megindokolt megoldás egy lépéseként szerepel a dolgozatban. Így például az anyagban szereplő ismeretek, definíciók, tételek puszta leírása azok alkalmazása nélkül nem ér pontot (még akkor sem, ha egyébként valamelyik leírt tény a megoldásban valóban szerephez jut). Annak mérlegelése, hogy az útmutatóban feltüntetett pontszám a fentiek figyelembevételével a megoldónak (részben vagy egészében) jár-e, teljes mértékben a javító hatásköre. Részpontszám jár minden olyan ötletért, részmegoldásért, amelyből a dolgozatban leírt gondolatmenet alkalmas kiegészítésével a feladat hibátlan megoldása volna kapható. Ha egy megoldó egy feladatra több, egymástól lényegesen különböző megoldást is elkezd, akkor legföljebb az egyikre adható pontszám. Ha mindegyik leírt megoldás vagy megoldásrészlet helyes vagy helyessé kiegészíthető, akkor a legtöbb részpontot érő megoldáskezdeményt értékeljük. Ha azonban több megoldási kísérlet között van helyes és (lényeges) hibát tartalmazó is, továbbá a dolgozatból nem derül ki, hogy a megoldó melyiket tartotta helyesnek, akkor a kevesebb pontot érő megoldáskezdeményt értékeljük (akkor is, ha ez a pontszám 0). Az útmutatóban szereplő részpontszámok szükség esetén tovább is oszthatók. Az útmutatóban leírttól eltérő jó megoldás természetesen maximális pontot ér, de bizonyítás nélkül csak az előadáson szereplő tételekre és állításokra lehet hivatkozni.

1. Bizonyítsa be, hogy $(3n^3 + 4\log n)(2n^2 - 5\sqrt{n}) \in \Omega(n^5)$.

   *Megoldás:* Be kell látni, hogy van olyan $c$ és $n_0$, amire $(3n^3 + 4\log n)(2n^2 - 5\sqrt{n}) \geq cn^5$ teljesül minden $n \geq n_0$-ra. **2 pont**
   $(3n^3 + 4\log n) \geq 3n^3$ minden $n > 0$ esetén **1 pont**
   $n^2 = n \cdot n \geq 5\sqrt{n}$ minden $n \geq 5$ esetén (már $n > 1{,}8$-ra is igaz) **2 pont**
   $2n^2 - 5\sqrt{n} \geq n^2$ minden $n \geq 5$ esetén **2 pont**
   $(3n^3 + 4\log n)(2n^2 - 5\sqrt{n}) \geq 3n^3 \cdot n^2 = 3n^5$ minden $n \geq 5$ esetén **1 pont**
   Tehát például a $c = 3, n_0 = 5$ megfelelő konstansok **2 pont**
   *Ha valaki (tévedésből) $O(n^5)$-t bizonyít max.* **5 pont**

2. Adjon $O(n^2 \log n)$ összehasonlítást használó algoritmust, ami eldönti, hogy az adott $x_1, x_2, \ldots, x_n$ és $S$ racionális számok esetén van-e olyan $x_i, x_j, x_k$ számhármas ($1 \leq i, j, k \leq n$), hogy $x_i + x_j + x_k = S$. *(A három indexnek nem kell feltétlen különbözőnek lennie, akár mindhárom is lehet ugyanaz.)*

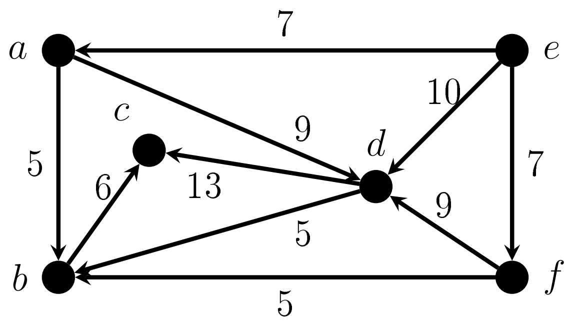
   *Megoldás:* Rendezzük a számokat nagyság szerint például összefésüléses rendezéssel **2 pont**

Minden $1 \leq i,j \leq n$ párra számítsuk ki az $S - x_i - x_j$ értéket **2 pont**
Bináris kereséssel döntsük el, hogy a kapott szám szerepel-e a rendezett sorozatban. Ha igen, akkor találtunk megfelelő számokat. **2 pont**
A rendezés lépésszáma $O(n \log n)$ **1 pont**
Egy párra a keresés $O(\log n)$ **1 pont**
Mivel $O(n^2)$ pár van, az összes lépésszám $O(n^2 \log n) + O(n \log n) \subseteq O(n^2 \log n)$ **2 pont**

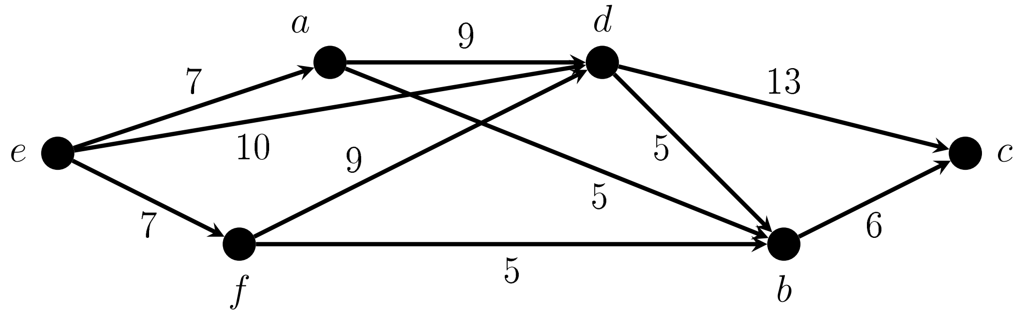
3. Egy $2k \geq 4$ csúcsú egyszerű, irányítatlan gráf mélységi bejárása során azt tapasztaltuk, hogy minden csúcsra a befejezési és a mélységi szám különbsége kisebb, mint $k$. Bizonyítsa be, hogy a gráf nem összefüggő és minden komponensben legfeljebb $k$ csúcs van.

*Megoldás:* Irányítatlan gráfban az 1-es mélységi számú csúcsból indított bejárás bejárja az összes vele egy komponensben lévő csúcsot **2 pont**
Ha az 1-es mélységi számú csúcs befejezési száma $b$, akkor ebben a komponensben pontosan $b$ csúcs van **2 pont**
Mivel most bszám $-$ mszám $= b - 1 \leq k - 1 < k$, ebben a komponensben legfeljebb $k$ csúcs van **2 pont**
Ilyenkor a DFS egy még be nem járt csúcsból újra indítja a keresést, (ennek mélységi száma $b + 1$ lesz) **2 pont**
Az előzőekhez hasonlóan látszik, hogy ebben a komponensben is legfeljebb $k$ csúcs van **2 pont**

4. a) Az előadáson tanult, DFS-t használó módszerrel adja meg a jobboldali gráf egy topologikus rendezését.
b) Határozza meg a leghosszabb út hosszát a topologikus rendezés első és utolsó pontja között az órán tanult módszerrel.




*Megoldás:* a) A DFS futtatásánál meg kell határozni a mélységi és befejezési számokat **1 pont**
Ha pl. $e$-ből indítjuk: $e$(1,6), $a$(2,4), $b$(3,2), $c$(4,1), $d$(5,3), $f$(6,5) (az első a mélységi szám, a másodig a befejezési szám), a jó számok **3 pont**
Egy topologikus sorrend a befejezési számok szerinti csökkenő sorrend: $e, f, a, d, b, c$
(Ha más pontból indítjuk a DFS-t, pl. $f$-ből, vagy más sorrenben válaszjuk a következő csúcsot akkor kijöhet másik sorrend is: $e, a, f, d, b, c$) **1 pont**




b) A leghosszabb utak $e$-ből indulva ($t(v)$ jelöli az $e$-ből $v$-be vezető leghosszabb út hosszát). Annak felírása/indoklása, hogyan kell ezt csinálni. **2 pont**

$t(e) = 0; t(a) = 7; t(f) = 7, t(d) = \max(t(e) + 10, t(a) + 9, t(f) + 9) = 16;$
$t(b) = \max(t(a) + 5, t(d) + 5, t(f) + 5) = 21; t(c) = \max(t(d) + 13, t(b) + 6) = 29$

a jó számolás **3 pont**

Ha a számolásnál ki vannak írva a megfelelő maximumok, akkor azért is jár az indoklásért adandó pont.

5. A párizsi olimpiára kerékpárral szeretnénk eljutni. Az utat már megterveztük. Kiderül, hogy éppen minden km-nél van egy szállás. Az út hossza $n$ km, egy nap legfeljebb $k$ km-t tudunk haladni. Ha még nem értünk Párizsba, akkor az előző szállástól számítva legfeljebb $k$ km-en belül meg kell szállnunk. Minden szállás költsége adott, az $i$-edikben $A[i]$ euro. Adjon algoritmust, ami dinamikus programozást használva meghatározza az út során a szállások minimális összköltségét. Az algoritmus lépésszáma legyen $O(kn)$.

*Megoldás:* Legyen $C[i]$ a minimális költsége annak, hogy eljutottunk az $i$-edik km-ig. (Ha megszállunk az $i$-edik km-nél, azt még nem számoljuk bele. Az indulás városában való megszállást sem számoljuk, azaz $A[0] = 0$.) $C[0] = 0$ **2 pont**

$C[i] = \min\{C[j] + A[j] \mid i - k \leq j < i \text{ és } j \geq 0\}$ **3 pont**

Helyesség indoklása: Az előző $k$ km-en belül meg kellett szállni, ez legyen az $j$-edik km. Eddig már tudjuk, hogy mennyi a minimális költség ($C[j]$) és ki kell fizetni még $A[j]$-t.

**2 pont**

A keresett érték $C[n]$ **1 pont**

Lépésszám: minden $C[i]$ kiszámolásához $k$ szám minimumát kell meghatározni, ez $O(k)$ lépés, $n$ értéket kell kiszámolni, a lépésszám összesen $O(kn)$. **2 pont**

6. Jobboldalon látható egy irányított, súlyozott gráf szomszédossági mátrixa. Dijkstra algoritmusával határozza meg $a$-ból az összes többi csúcsba vezető legrövidebb út hosszát. *(Indokolni nem kell, de látszódjon, lépésenként hogyan változik a távolságokat tároló $D[\ ]$ tömb és a KÉSZ halmaz.)*

$$\begin{array}{c} \\ a \\ b \\ c \\ e \\ f \\ g \end{array} \begin{array}{c} \begin{array}{cccccc} a & b & c & e & f & g \end{array} \\ \begin{pmatrix} \infty & 6 & 5 & 8 & \infty & \infty \\ 6 & \infty & \infty & 1 & 4 & \infty \\ \infty & 2 & \infty & \infty & 6 & \infty \\ \infty & 6 & \infty & \infty & \infty & 2 \\ \infty & \infty & \infty & 1 & \infty & 1 \\ \infty & \infty & \infty & \infty & \infty & \infty \end{pmatrix} \end{array}$$

*Megoldás:*

| | $a$ | $b$ | $c$ | $e$ | $f$ | $g$ | KÉSZ |
|---|---|---|---|---|---|---|---|
| 1. | 0 | 6 | 5 | 8 | $\infty$ | $\infty$ | $\{a\}$ |
| 2. | 0 | 6 | 5 | 8 | 11 | $\infty$ | $\{a,c\}$ |
| 3. | 0 | 6 | 5 | 7 | 10 | $\infty$ | $\{a,c,b\}$ |
| 4. | 0 | 6 | 5 | 7 | 10 | 9 | $\{a,c,b,e\}$ |
| 5. | 0 | 6 | 5 | 7 | 10 | 9 | $\{a,c,b,e,g\}$ |
| 6. | 0 | 6 | 5 | 7 | 10 | 9 | $\{a,c,b,e,g,f\}$ |

Az $a$ oszlopa csupa 0, (vagy nincs is felírva) **1 pont**
Az esetek nagy részében jól választja ki, hogy melyik csúcs kerüljön át a KÉSZ halmazba. **2 pont**
Az esetek nagy részében jól végzi el a javítást. **2 pont**
*Ha sok számolási hiba van, akkor csak az eddigi 5 pont jár.*
Ha viszont minden számolás jó, beleértve, hogy jól választotta ki a minimálisat, akkor további **5 pont**
5-nél több elszámolásnál ezért a részért nem jár pont.

7. Adott egy városok közötti úthálózatot leíró irányított gráf éllistája, ahol csúcsok a városok, élek a közvetlen utak és az élek súlya a városok közötti közvetlen útszakaszok hosszát adja meg. Egyes városokban nevezetességek is vannak, ezek száma minden városra meg van adva egy tömbben (a nevezetességek száma mindegyik városnál egy nemnegatív egész szám). Szeretnénk eljutni az $A$ városból a $B$ városba, az elsődleges szempont, hogy az út a lehető legrövidebb legyen. De ha esetleg több egyforma hosszú legrövidebb út is van, akkor ezek közül azt akarjuk kiválasztani, ami során az érintett városokban a lehető legtöbb nevezetességet tudjuk megnézni. Adjon algoritmust egy ilyen út megkeresésére, az algoritmus lépésszáma legyen $O(n^2)$.

*Megoldás:* Módosítsuk a Dijkstra algoritmust, minden csúcsra az eddigi legrövidebb utat tároló $D[\,]$ tömbön kívül egy plusz információt is tároljunk. Mindegyik legrövidebb úton megszámoljuk hány nevezetesség látható. $N[\,]$ tömbben ezek maximumát. **2 pont**
Az algoritmust úgy módosítjuk, hogy minden körben azt a csúcsot tesszük át a KÉSZ halmazba, amire a $D[\,]$ érték minimális, de ha több minimális van, akkor azok közül azt választjuk, amire $N[\,]$ maximális. **2 pont**
Amikor áttettük a $v$ csúcsot, akkor a $D[\,]$ és $N[\,]$ értékeket értelemszerűen módosítjuk a $v$-ből kimenő éleken. **2 pont**
Az algoritmus helyességének bizonyítása nagyon hasonló a Dijkstra algoritmus helyességének bizonyításához. Csak a következőt kell meggondolni: A kezdőpontból induló bármely út elejére teljesül, hogy az út hossza nem lehet nagyobb, mint az egész út hossza (mivel az élsúlyok nemnegatívak), valamint az első részen nem lehet több látnivaló, mint az egész úton. **2 pont**
A lépésszám ugyanannyi, mint a Dijkstra lépésszáma, ami $O(n^2)$ **2 pont**

*2. megoldás: Bár a feladat szövegében nem szerepel, feltételezhetjük, hogy minden közvetlen útszakasz hossza pozitív egész. A súlyok átskálázásával ez elérhető. Ezt indoklás nélkül is elfogadjuk.* Legyen a összes város látnivalóinak számának összege $C$. Szorozzunk meg minden élsúlyt $2nC$-vel. Az így kapott $G'$ gráfban ugyanaz lesz a legrövidebb út bármely két pont között, mint az eredeti $G$ gráfban, csak a hossza lesz $2nC$-szerese. **2 pont**
Most minden $v$ csúcsot kettőzünk meg, legyen egy $v'$ csúcs is. Minden $v$-ből kimenő él menjen ki most $v'$-ből és legyen egy él $v$-ből $v'$-be. Ennek súlya legyen $C$-ből kivonva a $v$ város látnivalóinak száma. **2 pont**
Keressük meg $G'$-ben a legrövidebb utat $A$-ból $B$-be, ez adja meg a keresett utat. **2 pont**
Helyesség: A új élek súlyai minden csúcson legfeljebb $C$-vel növelik az út hosszát, így bármely úton legfeljebb $2nC$-vel nő a hossz. Mivel ez kisebb, mint $2nC$, ezek nem befolyásolják azt, hogy az algoritmus egy eredeti legrövidebb útnak megfelelőt fog megtalálni. Viszont ezek között a legtöbb látnivalót adót adja meg. **2 pont**
A lépésszám ugyanannyi, mint a Dijkstra lépésszáma, $O(n^2)$, ebbe belefér az $G'$ előállítása is. **2 pont**