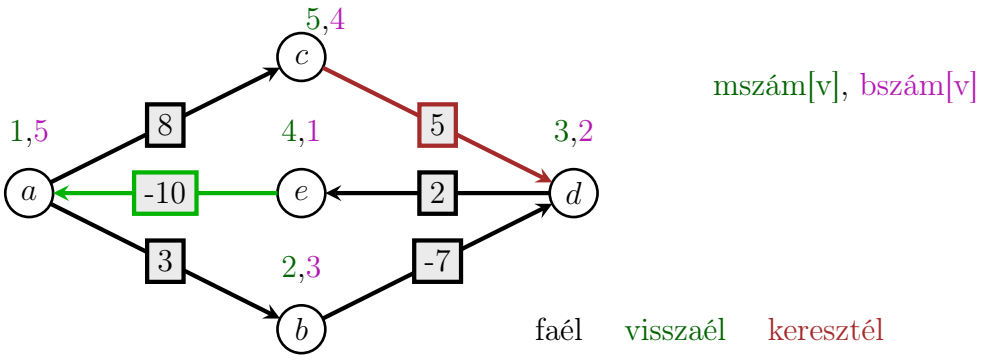
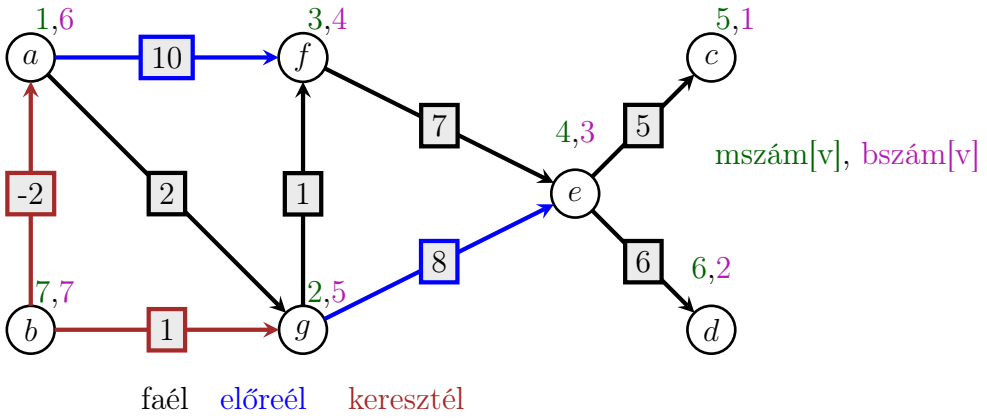
1. Éllistájukkal adottak az alábbi G_1 és G_2 irányított gráfok.
   G_1: **a**:b(3),c(8); **b**:d(-7); **c**:d(5); **d**:e(2); **e**:a(-10);
   G_2: **a**:g(2),f(10); **b**:a(-2),g(1); **c**:-; **d**:-; **e**:c(5),d(6); **f**:e(7); **g**:f(1), e(8);
   (a) Döntsük el mélységi bejárás segítségével erről a két gráfról, hogy DAG-e.
   (b) Amelyik gráf DAG, abban adjunk meg egy topologikus sorrendet és határozzuk meg az a jelű csúcsból a többi csúcsba vezető legrövidebb utak hosszát.

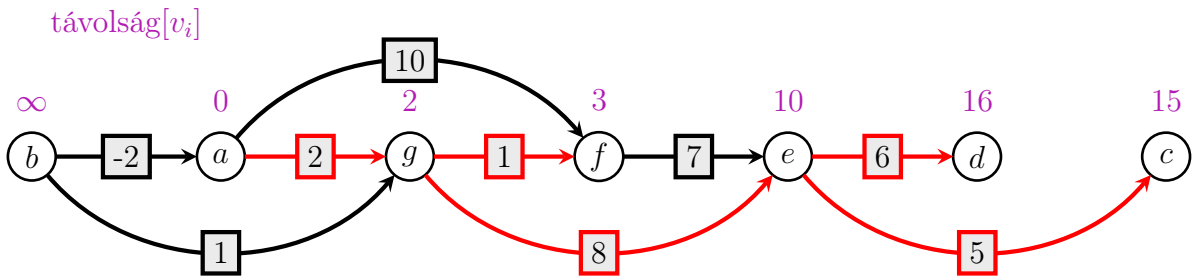
   Megoldás: (a) Nem DAG, mert a DFS során találunk visszaélet.



   (b) A gráf DAG, mert nincs visszaél.



   Topologikus sorrend: b,a,g,f,e,d,c, amit a DFS futtatásával kapunk, ha a befejezési számok szerint csökkenő sorrendben írjuk fel a csúcsokat.



   Az e csúcsnál a bejövő minimális él lehetne fe is ge helyett.

2. Egy éllistával adott irányított G gráfban néhány csúcs piros, néhány csúcs kék, a többi csúcs pedig színtelen. (A színezés egy, a csúcsokkal indexelt C tömbben adott). Melyik tanult algoritmus egyszeri futásával lehet $O(n+m)$ lépésben megoldani az alábbi feladatokat és hogyan?
   (a) El akarjuk dönteni, hogy az egyik adott p_1 piros csúcsból mik az elérhető kék csúcsok.
   (b) Egy adott piros p_1 csúcshoz meg akarjuk keresni a legközelebbi kék csúcso(ka)t.

Megoldás: (a) Futtassunk BFS-t vagy DFS-t p_1-ből újrakezdés nélkül. Azok a kék csúcsok alkotják a keresett halmazt, amelyek ezen futás során bejárttá válnak.

Ez azért igaz, mert mindkét algoritmus pontosan azokat a csúcsokat járja be (újrakezdés nélkül) a kezdőcsúcsból, amikbe van irányított út a gráfban a kezdőscsúcsból.

Az eljárás $O(n+m)$, mert mindkét algoritmus futásideje ennyi és ezután már csak végig kell menni a bejárva tömbön és ki kell olvasni a bejárt kék csúcsokat, ennek lépésszáma pedig $O(n)$.

(b) Futtassunk BFS-t p_1-ből újrakezdés nélkül, ennek végére minden csúcsra megkapjuk a p_1-től való távolságot. Végigmenve ezután a távolságok tömbjén minimumot keresünk a kék csúcsokhoz tartozó értékek között, majd kiírjuk azokat a kék csúcsokat, ahol ez a megtalált legkisebb távolság felvevődött.

Ez azért jó, mert a BFS helyesen határozza meg a távolságokat akkor, amikor a távolság alatt az utat alkotó élek számát értjük.

Az eljárás $O(n+m)$, mert a BFS futásideje ennyi és ezután már csak végig kell menni a távolságok tömbjén egyszer a minimum megkereséséhez, majd még egyszer azon kékek kiírásához, akiknek ez a távolsága. Ez utóbbi két lépés $O(n)$-s, ezért az egész futás $O(n+m)$.

---

3. Legyen G egy irányítatlan összefüggő gráf. Igaz-e, hogy
(a) G minden f éléhez van G-nek olyan éllistás megadása, melynek $\begin{matrix}\text{(a1) mélységi}\\ \text{(a2) szélességi}\end{matrix}$ bejárásában f egy faél?
(b) G minden F feszítőfájához van G-nek olyan éllistás megadása, melynek $\begin{matrix}\text{(b1) mélységi}\\ \text{(b2) szélességi}\end{matrix}$ bejárásában F minden éle faél?

Megoldás: (a1-2) Mindkét kérdésre igen a válasz. Tegyük fel, hogy f az u és v csúcsokat köti össze. Az éllista pontokat tartalmazó tömbjének első eleme legyen u, és u szomszédainak felsorolásában legyen v az első elem. Mind a BFS, mind a DFS u-ból fog indulni és elsőnek v-t fedezi fel.
(b1) Ez nem igaz például ha $G = K_n$ egy $n > 3$ pontú teljes gráf és F egy csillag (minden él középső csúcshoz csatlakozik). Ekkor akármilyen éllista esetén a DFS mindig tovább tud lépni és a DFS fa egy út lesz, nem pedig csillag. (b2) Ez sem igaz például ha $G = K_n$ ismét egy teljes gráf, de F most egy út (a maximális fokszáma 2). A BFS ekkor bármely éllista esetén a kiindulópontból rögtön felfedez minden más pontot, így a BFS fa egy csillag lesz, nem pedig út.

4. Éllistájával adott egy G irányított gráf, amiben nincsen irányított kör. Adott továbbá a gráf egy s csúcsa és szeretnénk meghatározni a gráf összes v csúcsa esetén az s-ből v-be vezető utak számát. Adjon erre a feladatra $O(n+m)$ lépésszámú algoritmust.

Megoldás: Jelöljük a gráfot $G = (V,E)$-vel. Felveszünk egy, a csúcsokkal indexelt tömböt, amelyet `darab`-bal jelölünk. A célunk az, hogy `darab`$[v]$ az $s \to v$ utak számát adja meg. Ennek egy lehetséges módja a következő. Meghatározzuk a gráf egy topologikus sorrendjét. Ha a v csúcs s előtt következik a topologikus sorrendben, akkor `darab`$[v] = 0$, mert ekkor nem lehet $s \to v$ út a topologikus sorrend definíciója szerint. Továbbá `darab`$[s] = 1$ (s-ből s-be pontosan egy út megy, az üres út, mivel a gráfban nincs irányított kör). Ezek ismeretében az s után következő v csúcsok esetén pedig a topologikus sorrendben előrefelé haladva a

$$\texttt{darab}[v] = \sum_{u \in V\,:\,(u,v)\in E} \texttt{darab}[u] \tag{1}$$

rekurzió adja meg az $s \to v$ utak számát, ahol a `darab`$[u]$-k már ismertek, hiszen ha van $u \to v$ él, akkor u biztos, hogy v előtt van a topologikus sorrendben. Ez a formula azért igaz, mert ha van $u \to v$ él, akkor s-ből u-n át v-be pontosan annyi út megy, mint s-ből u-ba, így az u-kra összegezve megkapjuk az $s \to v$ utak számát (itt megint használtuk, hogy élek csak a topologikus sorrend szerint előrefelé mehetnek).
Az algoritmus implementációja és lépésszáma: a topologikus sorrendről tudjuk, hogy $O(n+m)$ lépésben meghatározható. Az (1) lépéshez tudni kell, hogy melyik u-kból megy egy adott v-be él, amihez elkészítjük az eredeti éllistából a fordított éllistát, ami $O(n+m)$ lépés (lásd előző feladatsor 5. feladat).

Ezután minden v csúcsnál legfeljebb annyi u csúcsra kell megnézni a `darab`-értékeket, ahány bejövő éle van v-nek, és ezeket csúcsonként összeadni, vagyis ez és az egész algoritmus is $O(n+m)$ lépés.
Megjegyzés: az (1) rekurzióval definiált algoritmus tulajdonképpen dinamikus programozás, lásd később.

5. A 2. feladat folytatása:
(c) El akarjuk dönteni, hogy mindegyik kék csúcs elérhető-e legalább egy pirosból (azaz igaz-e, hogy minden kék csúcsra van olyan piros csúcs ahonnan ő elérhető).
(d) Meg akarjuk keresni a legrövidebb olyan utat a gráfban, ami piros csúcsból kék csúcsba vezet.

   Megoldás: (c) Adjunk a G gráfhoz egy új v csúcsot. Ebből vezessünk (irányított) élet a G minden piros csúcsához. A kapott gráf legyen G_1. A G_1 megkapható G-ből $O(n)$ lépésben. A csúcsok száma $n+1$ lesz, az éleké legfeljebb $n+m$. Ezután bejárást indítunk G_1-ben v-ből. Ez lehet DFS vagy BFS is. A v-ből elérhető kék csúcsok K halmazában pontosan azok a kék csúcsok lesznek, amelyek G-ben piros csúcsból elérhetők. A K-beli csúcsokat megjelöljük, és végül ellenőrizzük - ez $O(n)$ munka - hogy K az összes kék csúcs halmaza-e. A bejárás költsége $O(1+n+n+m) = O(n+m)$.

   (d) Itt az előzőekben felépített G_1 gráfban v-ből induló BFS-t hajtunk végre. Minden w csúcs mellé odaírjuk a v-től való G_1-beli távolságát, legyen ez a távolság d_w. Ez eddig $O(n+m)$ munka. Ezt követően meghatározzuk, a $d_w - 1$ mennyiségek minimumát, ahol w befutja a kék csúcsok halmazát. Ez további $O(n)$ munka. Az összköltség tehát $O(n+m)$.

6. Cirkuszi akrobaták egymás vállára állva minél nagyobb tornyot szeretnének létrehozni (a toronyban minden szinten csak egy akrobata lesz). Esztétikai és gyakorlati szempontok miatt egy ember vállára csak olyan állhat, aki nála alacsonyabb és könnyebb is. A cirkuszban n akrobata van, adott mindegyikük magassága és súlya.
(a) Adjon algoritmust, amely $O(n^2)$ lépésben megadja a lehetséges legtöbb emberből álló torony összeállítását.
(b) Adjon algoritmust, amely $O(n^2)$ lépésben megadja a lehetséges legmagasabb torony (hány centiméter magas) összeállítását.

   Megoldás: Vegyünk egy olyan G gráfot, melyben a csúcsok az akrobatáknak felelnek meg és két tetszőleges (v_i és v_j) csúcs között pontosan akkor fut irányított él, ha a j. akrobata állhat az i. akrobata vállán. Vegyünk fel továbbá egy f (mint föld) csúcsot, húzzunk belőle élt minden másik csúcsba (a föld vállán bárki állhat). Vegyük észre, hogy G gráf DAG (az egymásra állás feltételei miatt) és a benne lévő, f-ből induló utak megfeleltethetőek tornyoknak. Úgy kell ügyeskednünk az élek súlyozásával, hogy az ilyen utak hossza pont a tornyok (kérdezett) magassága legyen. Ezután mindkét probléma megoldható G-ben lefuttatott f-ből leghosszabb út kereséssel.

   a) Legyen minden él súlya 1. Az a torony, aminek a tetején az i. akrobata áll, legfeljebb tavolsag[v_i] ember magas lehet (mivel f-ből indulva nem létezik tavolsag[v_i] -1 -nél több csúcsból (akrobatából) álló út v_i-be). Keressük meg a legnagyobb tavolsag[v_i]-t, ehhez fog tartozni a legmagasabb torony. Maga a torony úgy kapható meg, hogy feljegyezzük a leghosszabb út keresés során minden csúcsnál azt is, hogy honnan érkezett a leghosszabb út és ennek mentén vissza tudjuk követni, hogy emlyik akrobaták állnak a toronyban.

   b) Legyen minden $v_i \to v_j$ él súlya a j. akrobata magassága, és hasonlóan az $f \to v_j$ él súlya is legyen a j. akrobata magassága. Ekkor egy út hossza éppen meg fog egyezni az úton található akrobaták magasságának összegével, így az a torony, aminek a tetején az i. akrobata áll, legfeljebb tavolsag[v_i] magas lesz (mivel f-ből indulva nem létezik hosszabb (összesen magasabb akrobatából) álló út v_i-be). Keressük meg a legnagyobb tavolsag[v_i]-t, majd magát a tornyot ugyanúgy kajuk, mint az (a) részben.

   A gráf szomszédossági mátrixának (vagy éllistájának) megadása $O(n^2)$ lépés, mert minden akrobatapárt meg kell vizsgálnunk, a leghosszabb út keresés $O(n^2)$ vagy $O(n+m)$, a megadás módjától függően, a maximum megtalálása pedig $O(n)$ lépés. Mivel a gráf egyszerű, $m \in O(n^2)$, így a teljes algoritmusok is belefférnek $O(n^2)$-be.

7. Éllistával adott a súlyozott élű G irányítatlan gráf, ahol az élek súlyai az 1,2,3 számok közül valók. Javasoljunk $O(n+m)$ költségű algoritmust egy adott s pontból az összes további v pontokba vivő legrövidebb utak hosszának meghatározására!

   Megoldás: Húzzuk szét a 2 súlyú éleket 2 éllé (egy-egy új pont felvételével), a 3 súlyúakat pedig 3 éllé. Az így kapott $G'=(V',E')$ gráfra teljesül, hogy $|V'| \leq |V|+2|E|$ és $|E'| \leq 3|E|$. A G-beli legrövidebb utak helyett elegendő G'-ben legkevesebb élből álló utakat keresni. Ez pedig a v-ből induló szélességi bejárással megtehető. Lépésszám $= O(|V'|+|E'|) = O(n+m)$.

8. Egy irányított G gráfban hagyjuk el a forrásokat ($d_{be}(v)=0$) és a nyelőket ($d_{ki}(v)=0$). A maradék gráfban ismét hagyjuk el a forrásokat és a nyelőket, ezt ismételjük, amig lehet. Bizonyítsuk be, hogy akkor és csak akkor kapunk üres gráfot, ha G-ben nincs irányított kör!

   Megoldás: Ha G-ben van egy irányított kör, akkor annak egyik csúcsa sem forrás és nem is nyelő, és ez akkor sem változik, ha a gráfból elhagyjuk a forrásokat és a nyelőket, tehát G-t a források és nyelők elhagyásának ismétlésével nem lehet kiüríteni.
   Ha viszont G-ben nincs irányított kör, vegyük G egy topologikus sorrendjét. A topologikus sorrend szerinti első csúcs mindig forrás, az utolsó mindig nyelő, így ezeket az eljárás első lépésében mindenképpen elhagyjuk (elképzelhető, hogy ezek egybeestek, de csak akkor, ha a gráf egyetlen izolált pontból állt). Ha a források és nyelők elhagyása után még van csúcsa a gráfnak, akkor az eredeti topologikus sorrend szerinti első meghagyott csúcs mindenképpen forrássá válik, mert aki előtte volt a topologikus sorrendben, azt elhagytuk, így belé már nem mehet él. Hasonlóan az utolsó meghagyott csúcs nyelővé válik, mert belőle kifelé már nem mehet él. Így ezeket a következő lépésben szintén el fogjuk hagyni (az összes többi, forrássá vagy nyelővé vált csúccsal együtt). Ezért minden lépésben el tudunk hagyni legalább két csúcsot, kivéve ha már csak egyetlen (izolált) csúcsa van a gráfnak, amit szintén el tudunk hagyni. Így a gráfot az eljárást legfeljebb $\lfloor n/2 \rfloor + 1$ lépésben ismételve ki lehet üríteni.

9. Legyen adott egy $n \times n$ pixelből álló fekete-fehér kép. Szeretnénk a képen a bal felső saroktól a jobb alsó sarokig egy jobbra-lefele haladó határvonalat húzni úgy, hogy a vonaltól jobbra-felfele eső fekete, valamint a vonaltól balra-lefele eső fehér pixelek számának az összege a lehető legkisebb legyen. Oldjuk meg ezt a feladatot $O(n^2)$ időben!

   Megoldás: Készítsünk egy gráfot, amiben csúcsok a pixelek (mint téglalapok) sarkai és élek a pixelek határai. A függőlegesek élsúlya legyen a tőle jobbra levő fekete és a tőle balra levő fehér pixelek számának összege. A vízszintes élek súlya viszont legyen 0. Minden függőleges élet irányítsunk lefelé, a vízszinteseket pedig jobbra. Könnyen látható, hogy a bal felső saroktól a jobb alsó sarokig jobbra-lefele haladó határvonal összsúlya éppen a minimalizálandó szám.

   Az élek irányításából következően a gráf DAG. Csúcsainak száma $(n+1)^2 = O(n^2)$, éleinek száma $m = 2n(n+1) = O(n^2)$. A függőleges élek súlyát meg tudjuk határozni $O(n^2)$ lépésben, ha soronként balról jobbra haladunk. Egy sorban először megszámoljuk a fekete pixelek számát ($O(n)$), ez lesz a bal szélső él súlya. Az i-edik pixel jobb oldalán lévő él súlya pedig 1-gyel kisebb a bal oldalán levőénél, ha a pixel fekete, és 1-gyel nagyobb, ha fehér. Így $O(n)$ lépésben megkapjuk egy sorban a súlyokat. A gráfban megkereshetjük a minimális súlyú utat $O(n+m) = O(n^2)$ lépésben. A kapott út épp megfelel a feltételeknek.

10. Egy városban 1000 ember lakik, mindenki minden nap elmondja az ismerőseinek az összes előző nap megtudott hírt. Tudjuk, hogy olyan az ismeretségek hálózata, hogy előbb-utóbb mindenki megtud mindent. Bizonyítsuk be, hogy van 90 olyan ember, hogy ha ők egyszerre megtudnak valamit, akkor legkésőbb 10 nap múlva mindenki megtudja!

    Megoldás: Tekintsük az ismeretségek által meghatározott (irányítatlan) gráfot. Ebben létezik egy leghosszabb út, $P=(x_0,x_1,\ldots,x_d)$. Futtassuk le a DFS(G,x_0) algoritmust. Mivel előbb-utóbb mindenki megtud mindent, ezért a gráf összefüggő, így a DFS egy fát eredményez. Fontos észrevétel, hogy P élei mind faélek lesznek, az út egyik vége x_0, a másik pedig x_d a DFS-fa egyik levele. (Ha a csúcsokat úgy rendezzük sorba, hogy először x_0,$x_1,\ldots,x_d$ jön, akkor a DFS először végigmegy ezen a P-n és a x_d-nek

1 lesz a befejezési száma.) A feszítőfában minden más pontba egyetlen út vezet, ennek éleinek száma az illető pont távolsága x_0-tól a fában. Legyen V_i a x_0-tól a fában i távolságra lévő pontok halmaza, ezek egymástól páronként diszjunkt halmazok. Ha a V_i halmazban lévő emberek megtudnak egy hírt, akkor 10 nap alatt megtudják a $V_{i+1}, \ldots, V_{i+10}$ halmazban levők is. Ráadásul azt is belátjuk, hogy ha $i \leq 10$, akkor a V_0,$V_1, \ldots, V_{i-1}$ halmazokban levők is. Tegyük fel, hogy van közük olyan aki nem, legyen ez y. Tudjuk, hogy $x_i \in V_i$. Ez azt jelenti, hogy a fában x_i és y távolsága nagyobb, mint 10. Így az $x_d, x_{d-1}, \ldots, x_i$ utat az y-ba vezető úttal folytatva egy hosszabb utat kapnánk, mint a leghosszabb út, ami ellentmondás.

Legyen $k = 0,1,2,\ldots,10$-re $W_k = V_k \cup V_{k+11} \cup V_{k+22} \cup \ldots$. Mivel a V_i halmazok páronként diszjunktak, a W_k halmazok is azok. Ha valamelyik napon W_k-ban mindenki megtudja a hírt, akkor 10 nap múlva mindenki meg fogja tudni. Tudjuk, hogy $\sum_{k=0}^{10} |W_k| = 1000$. Nem lehet, hogy minden $|W_k| \geq 91$, hiszen $91 \cdot 11 = 1001 > 1000$. Vagyis valamelyik $|W_k| \leq 90$. Nekik kell elmondani a hírt.

2. megoldás: A G gráfról feltehetjük, hogy fa. Vegyük egy tetszőleges r csúcsát és irányítsuk az éleit r-től távolodóan. Így DAG-ot kapunk. Ennek egy x csúcsa esetén az x gyökerű részfa azon csúcsok által feszített részgráf, amelyek elérhetők irányított úton x-ből.

Lemma. Ha $|G| \geq 11$, akkor van olyan x csúcs, amelynek a részfájában legalább 11 csúcs van és a részfa minden csúcsa elérhető x-ből legfeljebb 10 lépésben.

Bizonyítás: Ha a fában nincs 10-nél hosszabb r-ből induló irányított U út, akor $x = r$ jó lesz. Ha van ilyen út akkor azon a következő csúcs legyen r_1. Az r_1 részfája U létezése miatt legalább 11 csúcsot tartalmaz. Erre a részfára ismételjük az eddigieket: ha ebben a részfában nincs 10-nél hosszabb út, akkor megállunk, $x = r_2$ jó lesz; ha van hosszabb út, akkor azon lépünk egyet az r_3 csúcsba, stb. A DAG-tulajdonság miatt véges sok menetben véget ér az eljárás. □

Az eredeti gráfban válasszunk a Lemma szerinti $x_1 = x$-et, ezt töröljük a gráfból a részfájával együtt. A megmaradó gráf egy irányított fa, mint az eredeti, ebből válasszunk x_2-t a Lemma segítségével és töröljük a részfájával együtt. Tegyük ezt ismételten, amíg lehetséges. A kapott gyökerek $x_1, \ldots, x_k$. Ezekből elérhető a törölt részfák minden pontja legfeljebb 10 lépésben. A törölt részfák csúcsszáma legalább 11, ezért $k \leq 90$, hiszen $91 * 11 > 1000$. Ha már nem tudjuk folytatni a részfákat kiválasztó és törlő eljárást, akkor legfeljebb 10 csúcs marad, ezek x_k-val együtt egy legfeljebb 11 pontú fát alkotnak. Ennek a fának a csúcsai nyilván elérhetők x_k-ból legfeljebb 10 lépésben.

Az $\{x_1, \ldots, x_k\}$ halmaz tehát alkalmas lesz.

Megjegyzés: 89 embernek nem mindig elég elmondani a hírt. Vegyünk egy teljes gráfot 100 ponton, 90 pontjához $(v_1, \ldots, v_{90})$ csatoljunk egy-egy 10 hosszú utat (ez 10-10 új pont). Összesen 1000 pont lesz. Viszont, ha valamelyik v_i és a hozzá tartozó úton egyetlen ember sem tudja meg a hírt az első nap, akkor az út végén lévő ember biztos nem fogja megtudni 10 nap múlva. Mivel 90 ilyen út van (v_i-t az úthoz számolva), mindegyiken valakinek kell tudni a hírt.