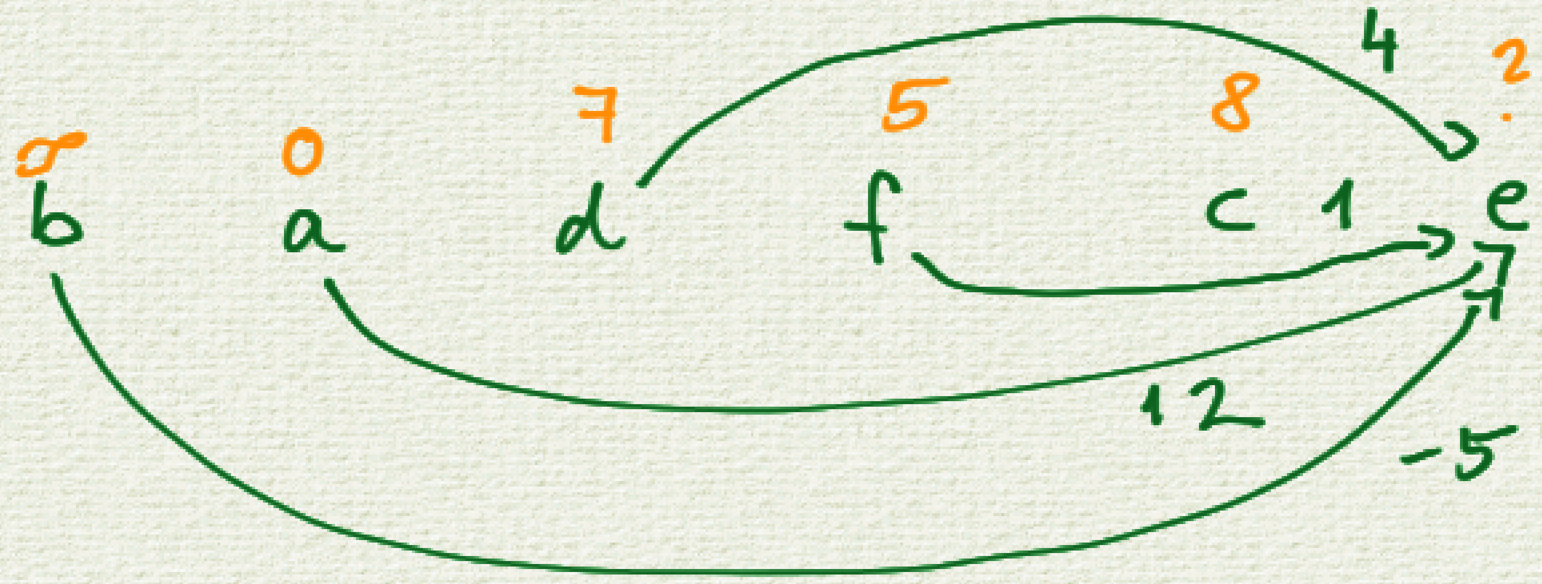
- Az alábbi irányított, élsúlyozott G gráf csúcsai a, b, c, d, e, f, élei pedig az alábbi éllistával adottak:
a :$c(8), d(7), f(6), e(12)$; **b** :$a(3), d(-8), e(-5)$; **c** :$-$; **d** :$c(2), e(4), f(-2), ;$ **e** :$-$; **f** :$c(5), e(1)$;
Ez a gráf egy DAG és topologikus sorrend benne a csúcsok b, a, d, f, c, e sorrendje (ezt nem kell leellenőrizni).

  Ezen topologikus sorrenddel alkalmazzuk a DAG-ban használható tanult eljárást (SzuperCsodás algoritmus) az a csúcsból induló legrövidebb utak meghatározására.
(a) Mennyi lesz a b és az a csúcsokra kiszámolt távolság és miért?
(b) Ha már tudjuk, hogy a d, f és c csúcsokra kiszámolt távolságok a következők: $távolság[d] = 7$, $távolság[f] = 5$, $távolság[c] = 8$, akkor mennyi lesz $távolság[e]$ és miért? (Azt nem kell leellenőrizni, hogy a d, f, c csúcsokra megadott távolságok helyesek.)

a-ból legrövidebb utak hossza



a) $távolság[b] = \infty$

$távolság[a] = 0$

mert top sorrend =>
- a-ból b-be nem lehet eljutni
- a-ból a-ba csak úgy hogy sehova se megyek => 0 hosszú út

b): e-be bemenő élekre

$$\min_{u \to v} \left\{ távolság[u] + c(u,v) \right\} = \min \begin{cases} \infty - 5 & (b) \\ 0 + 12 = 12 & (a) \\ 7 + 4 = 11 & (d) \\ 5 + 1 = 6 & (f) \end{cases} = 6$$

A $2, 10, 3, 5, 1, 4, 6$ tömb rendezése során előállhat-e az $1, 2, 3, 6, 5, 4, 10$ helyzet, ha
(a) buborékrendezést használunk? (b) kiválasztásos rendezést használunk?
A válaszait indokolja is meg!

2, 10, 3, 5, 1, 4, 6

a) buboréknál: amíg 10 a végére ér: 2, 3, 5, 1, 4, 6, 10 → [a 4-es soha meg a 6-os után]

közbenső fázisban : 2, 3, 1, 4, 5, | 6, 10 (közben 4 nem volt a 10 előtt)

innentől 6-os itt marad => soha lesz ez

b) 2, 10, 3, 5, 1, 4, 6 ⟶ 1, 10, 3, 5, 2, 4, 6 ⟶ 1, 2, 3, 5, 10, 4, 6 ⟶ 1, 2, 3, 4, 10, 5, 6

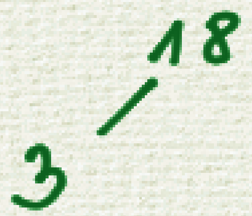
ez nem jó (még)

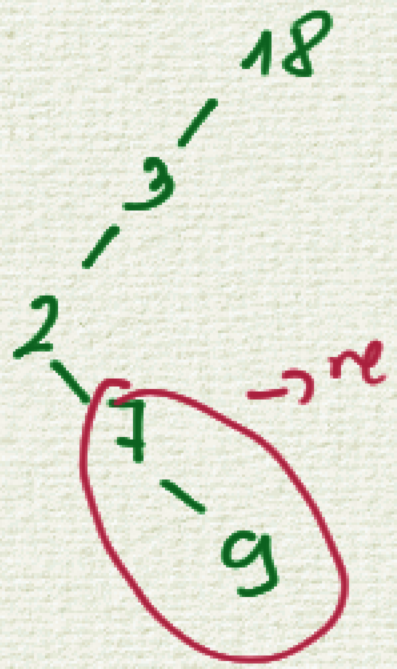
itt marad => soha lesz ez

Egy bináris keresőfában csupa különböző egész számot tárolunk és a 9-es szám keresésekor a keresési út mentén a $18, 3, x, 7, 9$ kulcsokat látjuk ebben a sorrendben.
A $2, 6, 8, 10, 20$ értékek közül melyekkel lehet egyenlő x és miért?
Figyelem, mind az 5 értékről nyilatkoznia kell!

mert először gyökérnél vagyunk és aztán bin.ker.fa tul. miatt 3 balra

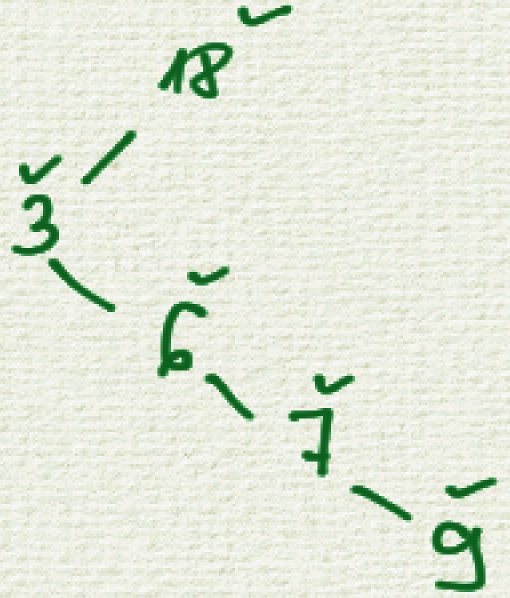
gyereket nézünk



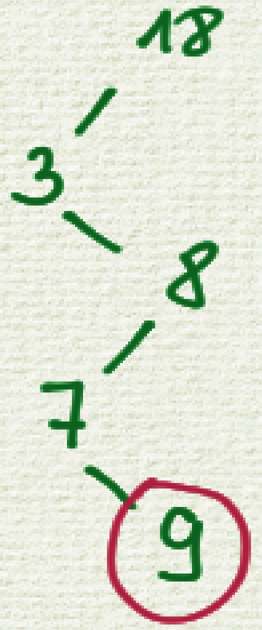


→ nem < 3

$x = 2$ nem lehet

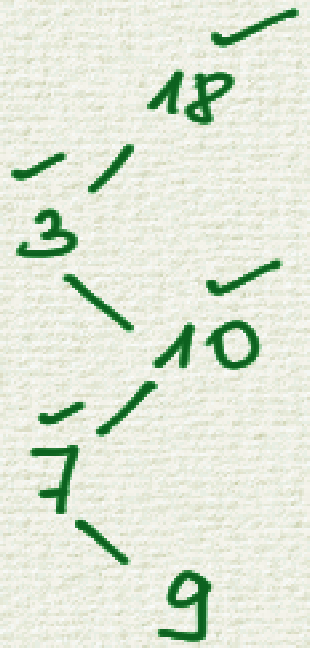


$x = 6$

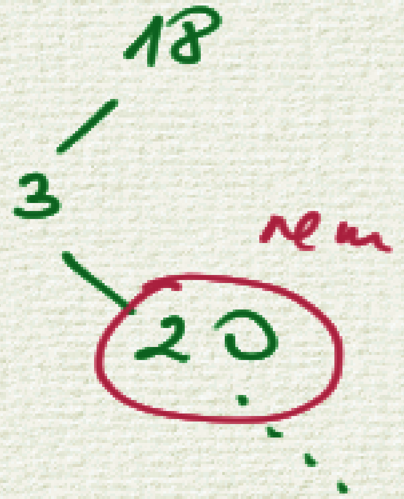


nem < 8

$x = 8$ nem lehet



$x = 10$



nem < 18

$x = 20$

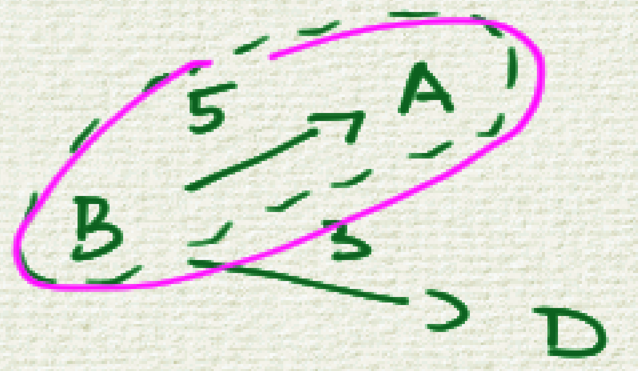
nem lehet

Dijkstra algoritmusát használjuk az A, B, C, D, E csúcsokból álló irányított, élsúlyozott G gráfban a B kezdőcsúcsból, eközben az *eddigi_legjobb* tömb így változik (az egyes sorok az *eddigi_legjobb* tömb változását mutatják egy-egy csúcs KÉSZ halmazba kerülése után).

A	B	C	D	E
5	*	∞	3	∞
4	*	7	*	∞
*	*	6	*	8
*	*	*	*	8
*	*	*	*	*

D megy a KÉSZ-be

Milyen csúcsokkal van biztosan összekötve az A csúcs (bejövő és kimenő élekre is gondoljon), mi ezeknek az élsúlya és miért?



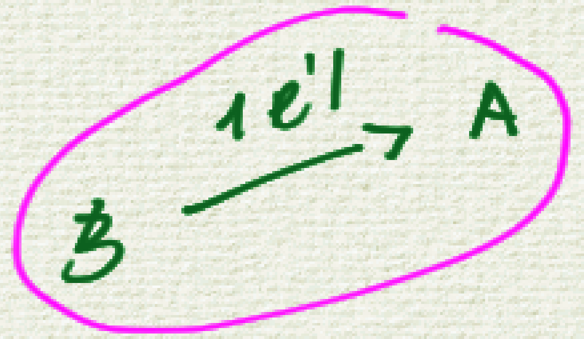
távolság

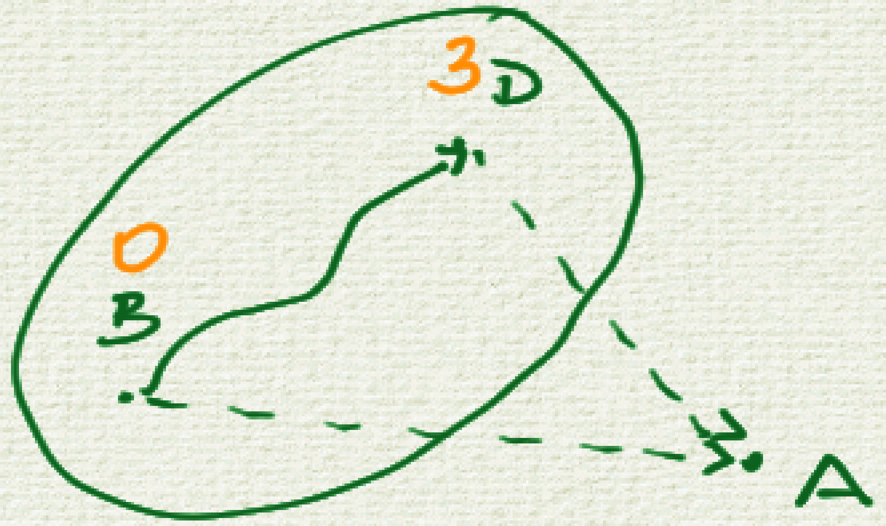
mert először
C
=
E
B-ből kijövő élek

eddigi_legjobb [v] = min olyan út hossza

$v \notin$ KÉSZ

=> elején





2. sor => D KÉSZ-be ment és a szomszédjaira változhat a tömb

A értéke csökkent => van $D \xrightarrow{1} A$ él és az új legjobb út hossza

$B \xrightarrow{3} D \xrightarrow{1} A$

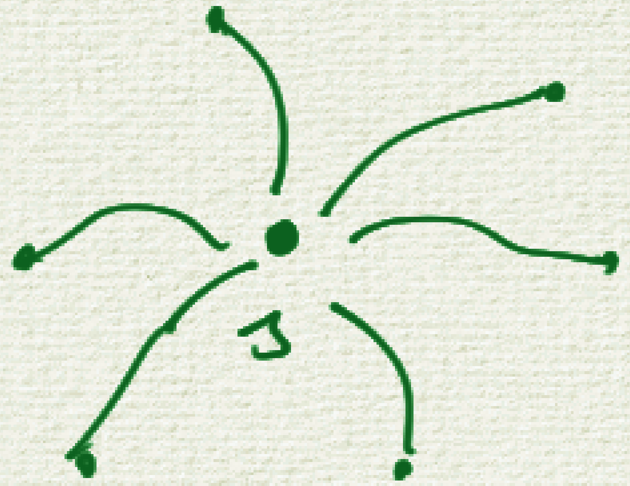
c(D,A) = 4 − távolság [D] = 1

3. sor => A kész-be és utána a kimenő szomszédjaira lehet csak változás => $A \xrightarrow{2} C$, $A \to E$ és

C értéke = 6 = távolság [A] + c(A,C) => c(A,C) = 2

E = 8 = 4 + c(A,E) => c(A,E) = 4

Szomszédossági mátrixával adott egy n csúcsú irányítatlan G gráf.
(a) Melyik tanult algoritmust lehet alkalmazni és hogyan, ha $O(n^2)$ lépésben el akarjuk dönteni, hogy G összefüggő-e?
(b) Adjon algoritmust, ami $O(n^2)$ lépésben eldönti, hogy G-re igaz-e, hogy nem öszefüggő, de van olyan hiányzó él, aminek a behúzásával összefüggővé tehető.
(c) Adjon algoritmust, ami $O(n^2)$ lépésben nemcsak megválaszolja a (b) rész kérdését, de meg is ad egy olyan élet, aminek behúzásával a nem összefüggő G gráf összefüggővé válik.

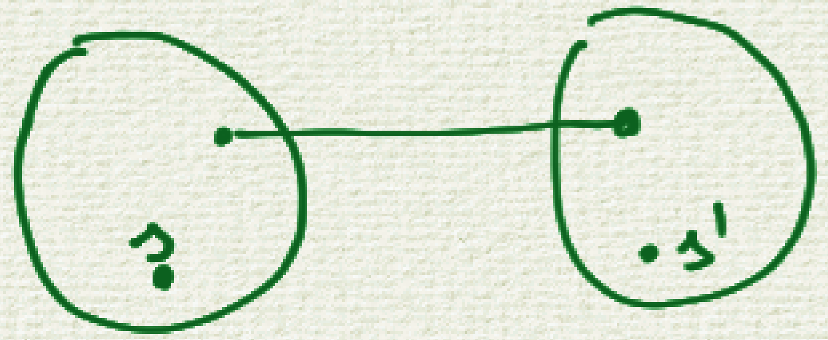


a) BFS / DFS bármely csúcsból és ha bejárva tömb csupa 1 ⟺ öf"

→ $O(n^2)$, mert tanultuk LSZ

LSZ BFS egy + $O(n)$ ⇒ $O(n^2)$

b)



feltétel ⟺ 2 komponense van

jóság

BFS-sel: futás egy s-ből és - ha ∀ csúcs bejárt ⇒ "Nem" → 1 komponens

$O(n^2)$

- ha nem ∀ csúcs bejárt ⇒ újraindítás s'-ből (nem bejárt) és ha ezután ∀ csúcs bejárt ⇒ 2 komponens ⇒ "Igen"

$O(n)$

$O(n^2)$

$O(n)$

≥ 3 komponens ← Kben "Nem"

LSZ

$O(n^2)+O(n)+O(n^2)+O(n)$

c) s és s' közti él jó lesz → ugyanaz algo ✓

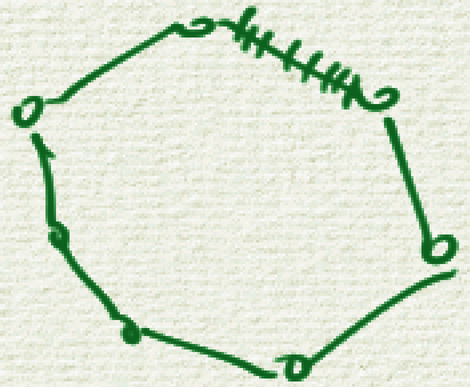
Egy nagyvárosban jelenleg egyáltalán nincsenek még bicikliutak, de az új városvezetés szeretné elérni, hogy néhány út külső sávjában biciklisávok kerüljenek felfestésre (a kiválasztott utak mindkét oldalán, azaz a bicikliutak mindkét irányba használhatóak). Ez csak úgy lehetséges, hogy ezen utak mentén néhány (de legalább egy) parkolóhely megszűnik. A város vezetése szeretné úgy megtervezni a bicikliutakat, hogy a városházától mindenhova el lehessen jutni bicikliúton, de ehhez a lehető legkevesebb pakolóhelyet kelljen megszüntetni (tartva az autósok haragjától). Szomszédossági mátrixával adott a város úthálózatának összefüggő, élsúlyozott, irányítatlan gráfja: a csúcsok a csomópontok (a városháza a H jelű csomópontban van), az élek a csomópontok közötti közvetlen utak (melyek mindkét irányban használhatók, nincsenek egyirányú utcák), az élek súlya pedig azt mutatja, hogy mennyi parkolóhely szűnik meg, ha ezen a szakaszon bicikliút készül.
Melyik tanult algoritmust lehet alkalmazni, hogyan és miért, ha $O(n^2)$ lépésben meg akarjuk oldani ezt a feladatot (a szokásos módon n a csomópontok számát jelöli)?

bicikliutak: • $\boxed{\text{öfő}}$ gráf

• $\boxed{\forall \text{ csúcsot lefednek}}$

$\boxed{\text{• körmentes}}$

mert ha lenne kör => ≥ 1 út feleslegesen biciklis



fa

feszítőfa

gráf bizonyos éleit választom

megszűnt

fa súlya = parkolók száma

minimalizálni => min. feszfa

$\boxed{\text{Prim } O(n^2)}$ → min feszfa → ide kell építeni az utakat

bárhonnan