

# 1. gyakorlat (2025. szeptember 10.)

## 1.1. Naiv algoritmus (nem feltétlen termináló)

Könyv: konyv.pdf#page=105

$$\begin{array}{ll} f_1 \rightarrow (l_2, l_1, l_3) & l_1 \rightarrow (f_1, f_3, f_2) \\ f_2 \rightarrow \text{tetszőleges, pl. } (l_1, l_2, l_3) & l_2 \rightarrow (f_3, f_1, f_2) \\ f_3 \rightarrow (l_1, l_2, l_3) & l_3 \rightarrow \text{tetszőleges, pl. } (f_1, f_2, f_3) \end{array}$$

### Algoritmus

1. Keressünk instabilitást a párosításban: egy fiúnak jobban tetszik egy másik lány ÉS aa lánynak jobban tetszik egy másik fiú.
2. Az (egyik) instabilitásbeli négyest felcseréljük.

**Kezdeti párosítás:**  $M_0 = \{(f_1, l_1), (f_2, l_2), (f_3, l_3)\}$ , ez nem stabil.

**Instabilitás:**  $[f_3 - l_2]$  és  $[f_1 - l_2]$  (több van, ezért választunk egyet)

$$M_1 = \{(f_1, l_2), (f_2, l_1), (f_3, l_3)\}$$

**Instabilitás:**  $[f_3 - l_2]$  és  $[f_3 - l_1]$

$$M_2 = \{(f_1, l_3), (f_2, l_1), (f_3, l_2)\}$$

**Instabilitás:**  $[f_1 - l_1]$  és  $[f_3 - l_1]$

$$M_3 = \{(f_1, l_3), (f_2, l_2), (f_3, l_1)\}$$

**Instabilitás:**  $[f_1 - l_2]$  és  $[f_1 - l_1]$

$$M_4 = \{(f_1, l_1), (f_2, l_2), (f_3, l_3)\} = M_0$$

Végtelen ciklus:  $M_0, M_1, M_2, M_3, M_4 = M_0, M_1, M_2, \dots$

## 1.2. Gale-Shapley algoritmus

(zh. feladat)

$$\begin{array}{ll} f_1 \rightarrow (l_3, l_2, l_5, l_1, l_4) & l_1 \rightarrow (f_3, f_5, f_2, f_1, f_4) \\ f_2 \rightarrow (l_1, l_2, l_5, l_3, l_4) & l_2 \rightarrow (f_5, f_2, f_1, f_4, f_3) \\ f_3 \rightarrow (l_4, l_3, l_2, l_1, l_5) & l_3 \rightarrow (f_4, f_3, f_5, f_1, f_2) \\ f_4 \rightarrow (l_1, l_3, l_4, l_2, l_5) & l_4 \rightarrow (f_1, f_2, f_3, f_4, f_5) \\ f_5 \rightarrow (l_1, l_2, l_4, l_5, l_3) & l_5 \rightarrow (f_2, f_3, f_4, f_1, f_5) \end{array}$$

	$l_1$	$l_2$	$l_3$	$l_4$	$l_5$
1.	$f_2, f_4, \boxed{f_5}$		$f_1$	$f_3$	
2.	$f_5$	$f_2$	$f_1, \boxed{f_4}$	$f_3$	
3.	$f_5$	$\boxed{f_2}, f_1$	$f_4$	$f_3$	
4.	$f_5$	$f_2$	$f_4$	$f_3$	$f_1$

**Észrevétel 1:** minden lányt csak az utolsó napon jelenik meg egy szerenádozó. Ezt a lányt senki sem húzza ki a preferencialistájáról.

**Észrevétel 2:** a többi lányt legalább 1 fiú nem húzza ki.

Kihúzások száma:  $n^2 - n - (n-1) = (n-1)^2 + 1$

**Hány nap alatt fejeződik be biztosan?** Előadás:  $n^2 + 1$  korlát ( minden nap legalább 1 lányt kihúznak).

**Észrevétel 1:**  $n^2 + 1 - n$

**Észrevétel 2:**  $n^2 + 1 - n - (n-1)$  (a végén n-1 mert az első észrevételbeli lányt nem számoljuk még egyszer)

**Összesen:**  $n^2 + 1 - n - (n-1) = (n^2 - 2n + 1) + 1 = (n-1)^2 + 1$

**Maximum hány stabil párosítás lehetséges n fiú és n lány között?** Az összes teljes párosítás száma:  $n!$ . Gale-Shapley algoritmus: minden lányt csak az utolsó napon jelenik meg egy szerenádozó.

**HF.** ha minden lánynak ugyanaz a preferencialistája, akkor csak 1 létezik. Észrevétel: néha 1-nél több van.

$$f_1 \rightarrow (l_1, l_2) \quad l_1 \rightarrow (f_2, f_1)$$



$$f_2 \rightarrow (l_2, l_1) \quad l_2 \rightarrow (f_1, f_2)$$

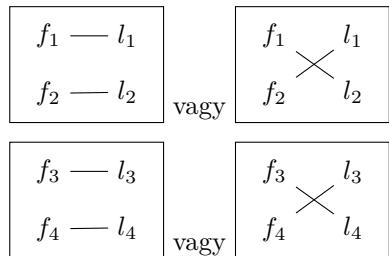
**Stabil párosítás:** minden **fiú** a preferencialistájának elsőjét kapja.

**Stabil párosítás:** minden **lány** a preferencialistájának elsőjét kapja.

**Meglepetés:** 2n fiú és 2n lány van, akár  $2^n$  párosítás is lehetséges.

$$\begin{array}{ll} f_1 \rightarrow (l_1, l_2, \dots) & l_1 \rightarrow (\dots, f_1) \\ f_2 \rightarrow (l_2, l_1, \dots) & l_2 \rightarrow (\dots, f_2) \\ f_3 \rightarrow (l_3, l_4, \dots) & l_3 \rightarrow (\dots, f_3) \\ f_4 \rightarrow (l_4, l_3, \dots) & l_4 \rightarrow (\dots, f_4) \\ f_{2i-1} \rightarrow (l_{2i-1}, l_{2i}, \dots) & l_1 \rightarrow (\dots, f_{2i-1}) \\ f_{2i} \rightarrow (l_{2i}, l_{2i-1}, \dots) & l_1 \rightarrow (\dots, f_{2i}) \\ \vdots & \vdots \\ f_{2n-1} \rightarrow (l_{2n-1}, l_{2n}, \dots) & l_1 \rightarrow (\dots, f_{2n-1}) \\ f_{2n} \rightarrow (l_{2n}, l_{2n-1}, \dots) & l_1 \rightarrow (\dots, f_{2n}) \end{array}$$

A sok-sok stabil párosítás:



Minden sorból választunk egyet:  $2^n$  lehetőség.

**HF.** az összes így kapott párosítás stabil (nincs instabilitás).

## 2. gyakorlat (2025. szeptember 17.)

### Miért lesz ez mind stabil párosítás?

Ha a felső kockát választottuk, akkor  $f_1$  párja a preferencialistáján az első lány, így  $f_1$  nem lehet instabilitás fiú tagja.

Ha az alsó kockát választottuk, akkor  $f_1$  párja a preferencialistáján a második lány, így  $f_1$  csak az  $l_1$  lánnyal lehet instabilitásban. Azonban  $f_1$  utolsó  $l_1$  preferencialistáján, így akárki is  $l_1$  párja, ő jobban tetszik  $l_1$ -nek, mint  $f_1$ . Így  $f_1$  most sem lehet instabilitás fiú tagja.

Ez a gondolatmenet minden fiúra alkalmazható, így minden a  $2^n$  párosítás stabil.

### Stabil szobatárs probléma

Nem feltétlenül létezik stabil párosítás.

$$A \rightarrow (B, C, D)$$

$$B \rightarrow (C, A, D)$$

$$C \rightarrow (A, B, D)$$

$$D \rightarrow \text{mindegy}$$

Három teljes párosítás létezik, A párja egyértelműen meghatározza a párosítást.

1.  $(A, B), (C, D) \rightarrow$  instabilitás: B és C
2.  $(A, C), (B, D) \rightarrow$  instabilitás: A és B
3.  $(A, D), (B, C) \rightarrow$  instabilitás: A és C

Így itt nincs stabil párosítás.

### Feladat

Legyen  $M_1$  és  $M_2$  két különböző stabil házasítás (visszatérünk az eredeti fiú-lány feladathoz). minden fiúhoz rendeljük hozzá az  $M_1$  és  $M_2$ -beli párja közül a neki jobban tetszőt. Mutassuk meg, hogy így teljes párosítást kapunk (nem triviális) ami ráadásul stabil.

### Teljes párosítás

Indirekt tegyük fel, hogy ez nem teljesül, következésképpen vannak olyan  $f'$  és  $f''$  különböző fiúk, hogy:

$$f' — l' \quad f' — \hat{l}'$$

$$M_1: f'' — l'' \quad \text{és} \quad M_2: f'' — \hat{l}''$$

és  $\text{JobbanTetszik}_{f'}(l', \hat{l}') = \text{JobbanTetszik}_{f''}(l'', \hat{l}'')$  Az általánosság megszorítása nélkül feltehetjük, hogy  $\text{JobbanTetszik}_{f'}(l', \hat{l}')$  és  $\text{JobbanTetszik}_{f''}(l', \hat{l}'') = l''$ .

Ha ez nem elég meggyőző: négy eset van:

1.  $JT_{f'}(l', \hat{l}') = l'$  és  $JT_{f''}(l'', \hat{l}'') = l''$
2.  $JT_{f'}(l', \hat{l}') = l'$  és  $JT_{f''}(l'', \hat{l}'') = \hat{l}''$
3.  $JT_{f'}(l', \hat{l}') = \hat{l}'$  és  $JT_{f''}(l'', \hat{l}'') = l''$
4.  $JT_{f'}(l', \hat{l}') = \hat{l}'$  és  $JT_{f''}(l'', \hat{l}'') = \hat{l}'' \rightarrow$  nem lehet

**HF.** Keressünk instabilitást  $M_1$ -ben vagy  $M_2$ -ben.

**Feladat.** Hosszú Gale-Shapley algoritmus futás. Először adott 5 fiú - 5 lány.

	$l_1$	$l_2$	$l_3$	$l_4$	$l_5$
1.	$f_1, f_5$	$f_2$	$f_3$	$f_4$	
2.	$f_1$	$f_2, f_5$	$f_3$	$f_4$	
3.	$f_1$	$f_2$	$f_3, f_5$	$f_4$	
4.	$f_1$	$f_2$	$f_3$	$f_4, f_5$	
5.	$f_1, f_4$	$f_2$	$f_3$	$f_5$	
6.	$f_1$	$f_2, f_4$	$f_3$	$f_5$	
7.	$f_1$	$f_2$	$f_3, f_4$	$f_5$	
8.	$f_1$	$f_2$	$f_4$	$f_5, f_3$	
9.	$f_1, f_3$	$f_2$	$f_4$	$f_5$	
10.	$f_1$	$f_2, f_3$	$f_4$	$f_5$	
11.	$f_1$	$f_3$	$f_4, f_2$	$f_5$	
12.	$f_1$	$f_3$	$f_4$	$f_5, f_2$	
13.	$f_1, f_2$	$f_3$	$f_4$	$f_5$	
14.	$f_2$	$f_3, f_1$	$f_4$	$f_5$	
15.	$f_2$	$f_3$	$f_4, f_1$	$f_5$	
16.	$f_2$	$f_3$	$f_4$	$f_5, f_1$	
17.	$f_2$	$f_3$	$f_4$	$f_5$	$f_1$

Ez általánosítható bármely számú fiúra és lányra  $\rightarrow (n - 1)^2 + 1$  nap.

**HF.** "Konstruáljuk meg" a preferencialistákat.

**HF.** Nem létezik olyan teljes párosítás (stabil vagy nem stabil), amelyben minden fiúnak szigorúan jobban tetszik a párja, mint a Gale-Shapley algoritmus által szolgáltatott párja.

### 3. gyakorlat (2025. szeptember 24.)

**Állítás (pareto optimalitás):** Jelölje  $M$  a Gale-Shapley algoritmus által szolgáltatott (fiú-optimális) párosítást. Ekkor nem létezik olyan  $M'$  teljes párosítás (nem stabil sem), ahol minden fiú jobban jár mint  $M$ -ben. (Olyat valószínűleg tudunk mutatni, amiben valamelyik fiú jobban jár mint  $M$ -ben, mivel a Gale-Shapley algoritmus esetén lehet olyan fiú aki nem a listájáról az első lányt kapta. Ha ez a fiú megkapja a listájának első lányát (és a többi fiú is valamilyen módon kap egy új párt), akkor márás mutattunk egy ilyen párosítást.)

**Bizonyítás:** Indirekt tegyük fel, hogy létezik olyan  $M'$  teljes párosítás, ahol minden fiú jobban jár, mint  $M$ -nél. Nézzük az  $M$ -et előállító Gale-Shapley algoritmus lefutását. Ekkor van olyan  $l$  lány, ainek csak az utolsó napon jelenik meg az ablaka alatt szerenádozó, aki végül a párja lesz. Legyen ez a fiú  $f$ . Most  $f$   $M'$ -beli  $l'$  párja jobban tetszik  $f$ -nek, mint  $l$  (spec.  $l' \neq l$ ). Jelölje  $l$   $M$ -beli párját  $f'$ . Ismét,  $f'$   $M'$ -beli  $l$  párja jobban tetszik  $f'$ -nek, mint az  $M$ -beli párja.

Igen ám, de ekkor  $f'$  az utolsó nap előtt szerenádozott  $l$ -nél és kosarat kapott, ellentmondva annak, hogy  $l$ -nél csak az utolsó nap szerenádozik valaki.

### Megoldandó probléma

$T(n) = aT\left(\frac{n}{b}\right) + f(n)$  alakú rekurziók.

#### Összefűréses rendezés

$$T(n) = 2T\left(\frac{n}{2}\right) + n \rightarrow T(n) = \Theta(n \log n)$$

#### Maximális növekedés

Adott pozitív számoknak egy  $A[1 : n]$  tömbje. Keressünk olyan  $1 \leq i \leq j \leq n$  indexeket, hogy  $A[j] - A[i]$  maximális.

### Oszd meg és uralkodj algoritmusok

Oszd meg és uralkodj algoritmust tervezünk (van más, hatékony módszer is).

1. Bontsuk a feladatot két feleakkora méretű feladatra:

- $A[1 : \frac{n}{2}]$ -ben hol van a maximális növekedés
- $A[\frac{n}{2} + 1 : n]$ -ben hol van a maximális növekedés

2. Rekurzívan megoldjuk a részfeladatokat:

- $A[1 : \frac{n}{2}] \rightarrow 1 \leq i' \leq j' \leq \frac{n}{2}$
- $A[\frac{n}{2} + 1 : n] \rightarrow \frac{n}{2} + 1 \leq i'' \leq j'' \leq n$

Egyelemű tömbökre direkt megoldás: a két index megegyezik az elem indexével.

3. A maximális növekedést adó  $i$  és  $j$  meghatározása. Három eset van:

- (a)  $A[1 : \frac{n}{2}]$ -ben (az első felében) van a maximális növekedés  $\rightarrow i = i'$  és  $j = j'$
- (b)  $A[\frac{n}{2} + 1 : n]$ -ben (a második felében) van a maximális növekedés  $\rightarrow i = i''$  és  $j = j''$
- (c)  $1 \leq i \leq \frac{n}{2} < j \leq n$  (az egyik az első, a másik a második felében). Ekkor  $i$ -t célszerű úgy választani, hogy  $A[i]$  az  $A[1 : \frac{n}{2}]$  legkisebb értéke,  $j$ -t pedig úgy, hogy  $A[j]$  az  $A[\frac{n}{2} + 1 : n]$  legnagyobb értéke.

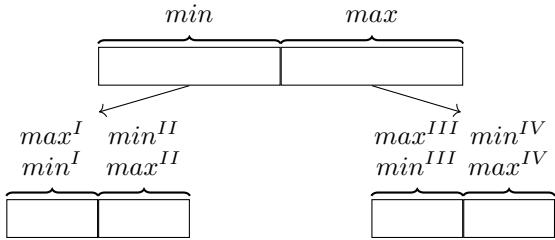
Nem tudjuk előre, hogy (a), (b) és (c) közül melyik adja az optimumot, ezért minden minden megvizsgáljuk és a legkedvezőbbet választjuk.

### Költség

$$T(n) = \underbrace{2T\left(\frac{n}{2}\right)}_{\text{két rekurzív hívás}} + \underbrace{\Theta(n)}_{\begin{array}{l} \min A[1 : \frac{n}{2}] \text{-ben} \\ \max A[\frac{n}{2} + 1 : n] \text{-ben} \\ (\text{a}), (\text{b}), (\text{c}) \text{ "közül a legnagyobb"} \end{array}}$$

Ez egy összefűréses rendezés rekurzió  $\rightarrow T(n) = \Theta(n \log n)$

**Van hatékonyabb?** Nem meglepő módon igen:



A rekurzív hívásokba süllyesztve a min és max kiválasztást  $\Theta(n)$   $\Theta(1)$ -re csökken a rekurzióban.

Zárt formula erre:  $T(n) = 2T\left(\frac{n}{2}\right) + 1$

**HF.** direkt számolás (önmagába helyettesítés)

Még egy érdekes dolog:

$$T(n) = T\left(\frac{n}{2}\right) + 1 \rightarrow T(n) = \Theta(\log n)$$

$$T(n) = T\left(\frac{n}{2}\right) + n \rightarrow T(n) = \Theta(n)$$

### 3.1. Mester-tétel

Könyv: konyv.pdf#page=10

$$f(n) \leftrightarrow n^{\log_b a}$$

1.
  - $T(n) = aT\left(\frac{n}{b}\right) + f(n) = 9T\left(\frac{n}{3}\right) + n$
  - $n^{\log_b a} = n^{\log_3 9} = n^2$  polinomiálisan nagyobb, mint  $f(n)$
  - $[f(n) = n = O(n^{2-\epsilon})]$ , például  $\epsilon = \frac{1}{2}$  esetén]
  - Mester téTEL első eset  $\rightarrow T(n) = \Theta(n^{\log_b a}) = \Theta(n^2)$
2.
  - $T(n) = T\left(\frac{2n}{3}\right) + 1$  ( $a = 1$ ,  $b = \frac{3}{2}$ ,  $f(n) = 1$ )
  - $n^{\log_b a} = n^{\log_{\frac{3}{2}} 1} = n^0 = 1$  aszimptotikusan megegyezik  $f(n)$ -nel.
  - $[f(n) = 1 = \Theta(n^0) = \Theta(1)]$
  - Mester téTEL második eset  $\rightarrow T(n) = \Theta(n^{\log_b a} \log n) = \Theta(\log n)$
3.
  - $T(n) = 3T\left(\frac{n}{4}\right) + n \log n$  ( $a = 3$ ,  $b = 4$ ,  $f(n) = n \log n$ )
  - $n^{\log_b a} = n^{\log_4 3} \approx n^{0,793}$  ( $\log_4 3 < \log_4 4 = 1$ )
  - $f(n)$  polinomiálisan nagyobb, mint  $n^{0,793}$
  - $f(n) = n \log n = \Omega(n^{0,793+\epsilon})$  pl.  $\epsilon = 0,1$  esetén
  - A Mester téTEL harmadik esetének néz ki, de ahhoz hogy tényleg az legyen, még egy dolgot ellenőrizni kell:
    - ◊ a  $f\left(\frac{n}{b}\right) \leq cf(n)$  alkalmas  $c < 1$  konstanssal
    - ◊  $3f\left(\frac{n}{4}\right) = 3\left(\frac{n}{4} \log \frac{n}{4}\right) = \frac{3}{4}n \log \frac{n}{4} \leq \frac{3}{4}n \log n$
    - ◊  $c = \frac{3}{4}$  megfelelő
  - ◊ Így tényleg mester téTEL harmadik eset  $\rightarrow T(n) = \Theta(f(n)) = \Theta(n \log n)$

**HF.** mindenféle ilyen rekurziók:

- $T(n) = 2T\left(\frac{n}{2}\right) + n^3$
- $T(n) = 2T\left(\frac{n}{2}\right) + n^2$
- $T(n) = 2T\left(\frac{n}{2}\right) + n \log n$

## 4. gyakorlat (2025. október 1.)

### ”Oszd meg és uralkodj”

1. részproblémákra bontjuk
2. a részproblémákat megoldjuk
3. ezek eredményéből kiszámoljuk a végeredményt

### Inverziószámok keresése/számlálása

Legyen  $A[1..n]$  egy tömb egyedi számokkal. Az  $(i, j)$  indexpár inverzió, ha  $1 \leq i < j \leq n$  és  $A[i] > A[j]$ . Számoljuk meg, hány inverzió van a tömbben.

#### Naiv módszer:

Minden indexpárt ellenőrzünk:  $O(n^2)$ .

#### Oszd meg és uralkodj (könyv: konyv.pdf#page=8):

1. Részproblémákra bontjuk: keressük az inverziókat  $A[1 : \frac{n}{2}]$  és  $A[\frac{n}{2} + 1 : n]$ -ben külön
2. A részproblémák megoldása (rekurzívan): ha a részprobléma 1 elemű, akkor nincs inverzió, adjunk vissza 0-t
3. Válaszok egyesítése: keressük az összes inverziót.
  - (a)  $1 \leq i < j \leq \frac{n}{2} \rightarrow A[1 : \frac{n}{2}]$  részprobléma
  - (b)  $\frac{n}{2} + 1 \leq i < j \leq n \rightarrow A[\frac{n}{2} + 1 : n]$  részprobléma
  - (c)  $1 \leq i \leq \frac{n}{2} < j \leq n \rightarrow ?$

Ötlet: rendezzünk menet közben (merge sort). Ha rendezett a két résztömb, könnyebb a (c) esetet ellenőrizni. Két eset van:

- $A[i] < A[j] \rightarrow$  nincs inverzió. Mivel rendezett a tömb, ezért minden  $j \leq k \leq n$  k index se lesz inverzió.
- $A[i] > A[j] \rightarrow$  inverzió. Mivel rendezettek a tömbök, ezért minden  $i \leq l \leq \frac{n}{2}$  l index is inverzió lesz j-vel.

#### Végeredmény

A két részprobléma megoldásai + a köztük végzett ellenőrzés eredménye. Futási idő:  $O(n \cdot \log n)$  mint a merge sort.

### Többségi elem keresése

Egy elem többségi elem egy  $A[1 : n]$  tömbben, ha  $\frac{n}{2}$ -nél többször fordul elő benne (pl. minimum 6/10, minimum 4/7, stb.). Ha van többségi elem egy tömbben, akkor biztosan egy van. Keressük meg a többségi elemet, amennyiben van ilyen.

#### Naiv módszer:

Minden elemet megszámolunk:  $O(n^2)$ .

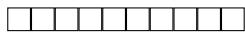
#### Oszd meg és uralkodj:

1. Részproblémák:
  - Ötlet: ha  $x$  többségi elem  $A[1 : n]$  tömbben, akkor (mivel  $\frac{n}{2}$ -nél többször fordul elő) többségi elem lesz  $A[1 : \frac{n}{2}]$ -ben vagy  $A[\frac{n}{2} + 1 : n]$ -ben.
  - Szóval a részproblémák: vegyük a tömb egyik ( $A[1 : \frac{n}{2}]$ ) és másik ( $A[\frac{n}{2} + 1 : n]$ ) felét.
2. Részproblémák megoldása rekurzívan:
  - Ha 1 elemű a tömb, akkor az az elem a többségi elem.
  - $A[1 : \frac{n}{2}] \rightarrow m'$ ,  $A[\frac{n}{2} + 1 : n] \rightarrow m''$  ( $m'$  és  $m''$  lehet üres is (ha nincs többségi elem az adott résztömbben))
  - $m'$  és  $m''$  közül valamelyik az egész tömb többségi eleme is lesz.
3. Számoljuk meg az előfordulásokat, hogy  $m'$  és  $m''$  hányszor szerepel  $A[1 : n]$ -ben.

- Ha  $m'$  többször fordul elő, mint  $m''$ , akkor  $m'$  lesz a többségi elem.
- Ha  $m''$  többször fordul elő, mint  $m'$ , akkor  $m''$  lesz a többségi elem.
- Ha  $m'$  és  $m''$  ugyanannyiszor fordul elő, akkor nincs többségi elem.

**Futási idő:**  $O(n \cdot \log n)$

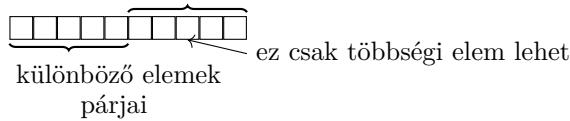
**Tudunk-e erre a problémára hatékonyabb (lineáris) algoritmust?** Igen, de nem oszd meg és ural-kodj.

  $A[1 : n]$

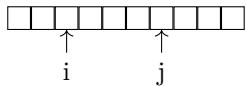
Ha tudjuk, hogy x többségi elem  $A[1 : n]$  és  $A[1] \neq A[2]$ , akkor többségi elem  $A[3 : n]$ -ben.  $A[1 : 10]$ , ha x többségi elem, akkor legalább 6-szor van jelen  $A[3 : 10]$ -ben.

**Ötlet:** rendezzük át az  $A[1 : n]$  tömböt úgy, hogy párokat kapunk a tömb elején, amik nem azonos értékek.

minden elem ugyanaz



**Hogyan?**



Tegyük fel, hogy  $A[j]$ -nél vagyunk, és  $A[1 : i]$  a párok sora, míg  $A[i + 1 : j]$  homogén. Ekkor ha  $A[j] = A[i + 1]$ , akkor a homogén részhez hozzávesszük  $A[j]$ -t. Ha viszont  $A[j] \neq A[i + 1]$ , akkor megcseréljük  $A[j]$ -t  $A[i + 2]$ -vel, és ezzel növeljük a párok számát (az első rész 2-vel hosszabb). Innentől a homogén rész az  $A[i + 3 : j]$ . A rendezés végén ha marad homogén rész, akkor az a többségi elem.

**Futási idő:** mivel egyszer megyünk végig, ez lineáris,  $O(n)$ .

**Mester téTEL gyakorlás**

- $T(n) = 2T(\frac{n}{2}) + n^3$
- $a = 2, b = 2, f(n) = n^3$
- $n^{\log_b a} = n^{\log_2 2} = n^1 = n$
- $f(n) = n^3 > n$

3. eset:  $f(n) = \Omega(n^{\log_b a + \epsilon}) \rightarrow \epsilon = 2 \checkmark$

Regularitás:

$$af\left(\frac{n}{b}\right) \leq cf(n)$$

$$2f\left(\frac{n}{2}\right) \leq cf(n)$$

$$2\frac{n^3}{8} \leq cn^3$$

$$f\left(\frac{n^3}{8}\right) \leq cn^3$$

$$\frac{n^3}{4} \leq cn^3$$

$$c < 1, c \geq \frac{1}{4} \checkmark$$

$$T(n) = \Theta(f(n)) = \Theta(n^3)$$

## 5. gyakorlat (2025. október 8.)

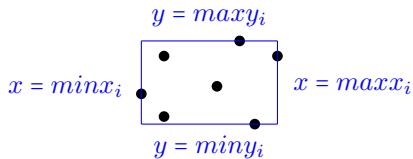
### Oszd meg és uralkodj algoritmusok

#### Geometria

Adott  $n$  pont a síkon. Határozzuk meg

1. a két legközelebbiit
2. a két legtávolabbiit.

Kicsit távolabbról indulunk.



$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$  a pontok. Határozzuk meg a legkisebb olyan téglalapot amelynek oldalai párhuzamosak a koordinátatengelyekkel és az összes pontot tartalmazza.

Két min és két max számolás  $\rightarrow 4n - 4$  összehasonlítás. Lehet kevesebbel is? Egyszerre  $\min x_i$  és  $\max x_i$ , illetve  $\min y_i$  és  $\max y_i$ ? Lássuk a  $\min x_i$  és  $\max x_i$  esetet:

$$\underbrace{x_1, x_2, x_3, x_4, x_5, x_6, \dots, x_{n-2}, x_{n-1}}_{\text{underbrace}}, x_n$$

Az egyszerűség kedvéért tegyük fel, hogy az értékek páronként különbözőek.

Ha  $x_1 < x_2$ , akkor  $x_1 \neq \max x_i$  és  $x_2 \neq \min x_i \Rightarrow \left[ \frac{n}{2} \right]$  összehasonlítással a feladat visszavezethető  $\left[ \frac{n}{2} \right]$  szám közül a maximum és a minimum meghatározására. Ezzel az összehasonlítások számát  $\left[ \frac{n}{2} \right]$ -vel tudtuk csökkenteni. Igazából nem számít, ha az elemek között vannak megegyezők.

Legkisebb téglalap  $\rightarrow$  legkisebb sokszög (amely az összes pontot tartalmazza).



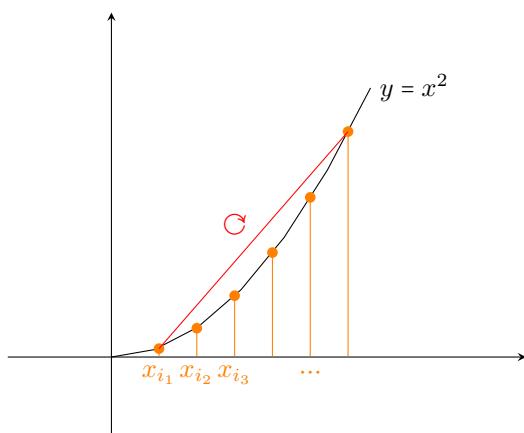
Nem nehéz belátni, hogy a két legtávolabbi pont a konvex burok két csúcsa.

Első lépés a két legtávolabbi pont meghatározásához a konvex burok meghatározása.

$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n) \rightarrow$  a konvex burok csúcsainak felsorolása a konvex burkon az óramutató járása szerint.

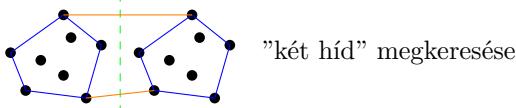
Milyen "bonyolult" a konvex burok meghatározása? Legalább annyira, mint a rendezés.

$(x_1, x_1^2), (x_2, x_2^2), \dots, (x_n, x_n^2)$  bemenethez:



Egy konvex burok algoritmus rendezi is az  $x_1, x_2, \dots, x_n$  számokat (csökkenően).

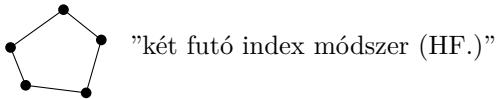
$\mathcal{O}(n \log n)$  költségűek a "jó" konvex burok algoritmusok. Több ilyen is van, oszd meg és uralkodj is:



”két híd” megkeresése

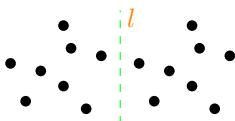
1. két feleakkora feladat
2. rekurzívan megoldjuk a részfeladatokat
3. ”egyesítjük” a két konvex burkot (két híd)

Ezek után a két legtávolabbi csúcs a konvex burkon (nem oszd meg és uralkodj algoritmus):



### 1. minimális távolság

Oszd meg és uralkodj algoritmus

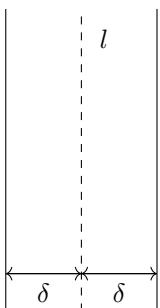


1. két feleakkora feladatra bontás  $\rightarrow P_{bal}, P_{jobb}$
2. rekurzívan meghatározzuk a minimális távolságot  $P_{bal}$ -ban és  $P_{jobb}$ -ban:
  - $P_{bal} \rightarrow \delta_{bal}$
  - $P_{jobb} \rightarrow \delta_{jobb}$
3. min távolság meghatározása

A minimális távolság vagy baloldalon vagy jobboldalon van, vagy keresztezi az  $l$  felező egyenest.

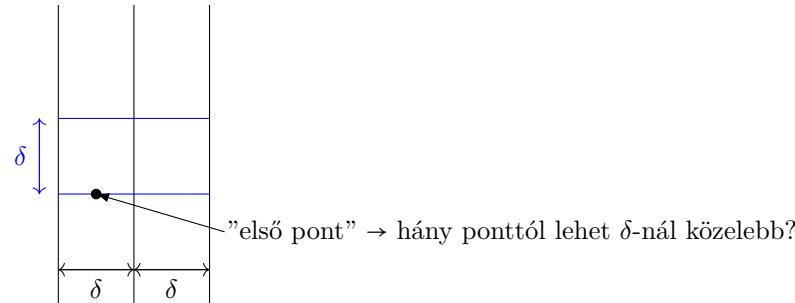
A harmadik esetben  $l$ -től nem túl messze lévő pontokra elég szorítkozni.

Legyen  $\delta = \min(\delta_{bal}, \delta_{jobb})$ .

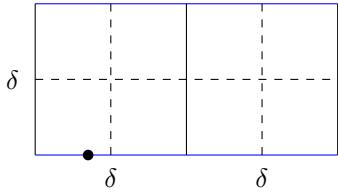


Ebben a  $\delta$  sugarú sávban kell a két pontnak lenni, amelyek a min távolságot minimalizálják.

Itt van még valami, amit érdemes észrevenni. Tekintsük a sávbeli pontokat az  $y$  koordinátájuk szerint monoton növekvően rendezetten.



Minden ilyen pont szükségképpen ebben a téglalapban van:



Tekintsük a fenti 8 kis négyzetet. Mivel ezek átmérője  $\frac{8}{2}\sqrt{2} = \frac{8}{\sqrt{2}} < \delta$ , és az  $l$  bal, illetve jobb oldalán a min. távolság legalább  $\delta$ , így a 8 kis négyzet egyikében sem lehet egynél több  $P$ -beli pont. Így a legalsó ponttól  $\delta$ -nál kisebb távolságra csak a következő 7 pont valamelyike lehet. Felfelé haladva ez mindenig igaz lesz.

$$\text{Költség: } T(n) = 2T\left(\frac{n}{2}\right) + \underbrace{\mathcal{O}(n \log n)}_{\text{rendezések}} \rightarrow T(n)(\mathcal{O}(n \log^2 n))$$

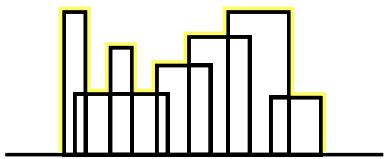
$$\mathcal{O}(n \log^2 n) \rightarrow \mathcal{O}(n \log n)$$

Előfeldolgozás → előrendezés (x és y koordináták szerint is) → rekurzív hívásokban már csak kiválogatás

$$T(n) = 2T\left(\frac{n}{2}\right) + \mathcal{O}(n) \rightarrow T(n) = \mathcal{O}(n \log n)$$

## 6. gyakorlat (2025. október 15.)

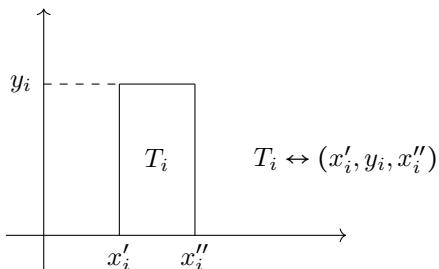
Még egy geometriai oszd meg és uralkodj algoritmus (igazából összefésülés variáció)



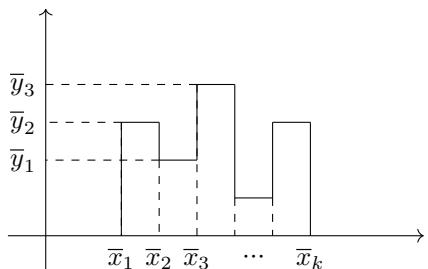
$n$  téglalap, amelyek az  $x$  tengelyen állnak

Feladat: sziluett (felső burkoló törötvonal) meghatározása.

Téglalapok:  $T_1, T_2, \dots, T_n$



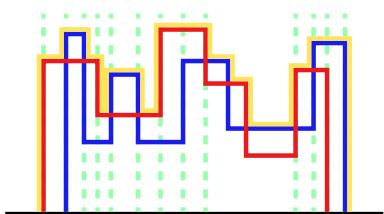
Sziluett



$(\bar{x}_1, \bar{y}_1, \bar{x}_2, \bar{y}_2, \dots, \bar{y}_{k-1}, \bar{x}_k)$

**Oszd meg és uralkodj algoritmus**

1. Két feleakkora méretű részfeladat
  - $T_1, T_2, \dots, T_{n/2}$  sziluettjének meghatározása
  - $T_{n/2+1}, T_{n/2+2}, \dots, T_n$  sziluettjének meghatározása
2. Rekurzívan megoldjuk a részfeladatokat (egy téglalap esetén triviális)
3. Összekombináljuk a két sziluettet:



(ez már egy összefésülés)

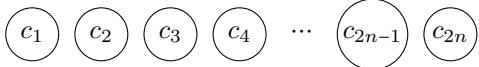
Minden szakaszban a magasabban fekvő szakaszt választjuk a két sziluetten.

Implementáljuk ezt ha tényleg minden részletében érteni szeretnénk.

Költség:  $T(n) = 2T(\frac{n}{2}) + \mathcal{O}(n) \rightarrow T(n) = (O)(n \log n)$

## 6.1. Érmék egy sorban (tegnapi előadás)

(Ez már nem oszd meg és uralkodj)



Egy sorban  $2n$  (nem feltétlenül páronként) különböző érmék vannak. Két játékos felváltva elvesz a (megmaradt) érmék sorának valamelyik végéről egy érmét. Az győz, aki nél az elvett érmék értékének összege nagyobb.

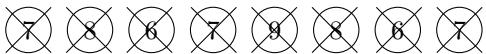


A nyerő stratégia a nem vesztő stratégia.

Nyerő stratégia a kezdő számára: tud úgy érmét elvenni először, hogy erre "bármit lép" a másik játékos, a kezdő megint tud úgy érmét elvenni, hogy erre "bármit lép" a másik játékos, ..., a kezdő nyer (nem veszít)

Nyerő stratégia a másik játékosnak: bármilyen lépést a kezdeti játékos, a másik játékos tud úgy érmét elvenni, hogy erre bármilyen lépést a kezdeti játékos, a másik tud úgy érmét elvenni, ..., a másik játékos nyer (nem veszít).

Ebben a játékban a kezdőnek van nyerő (nem vesztő) stratégiája!

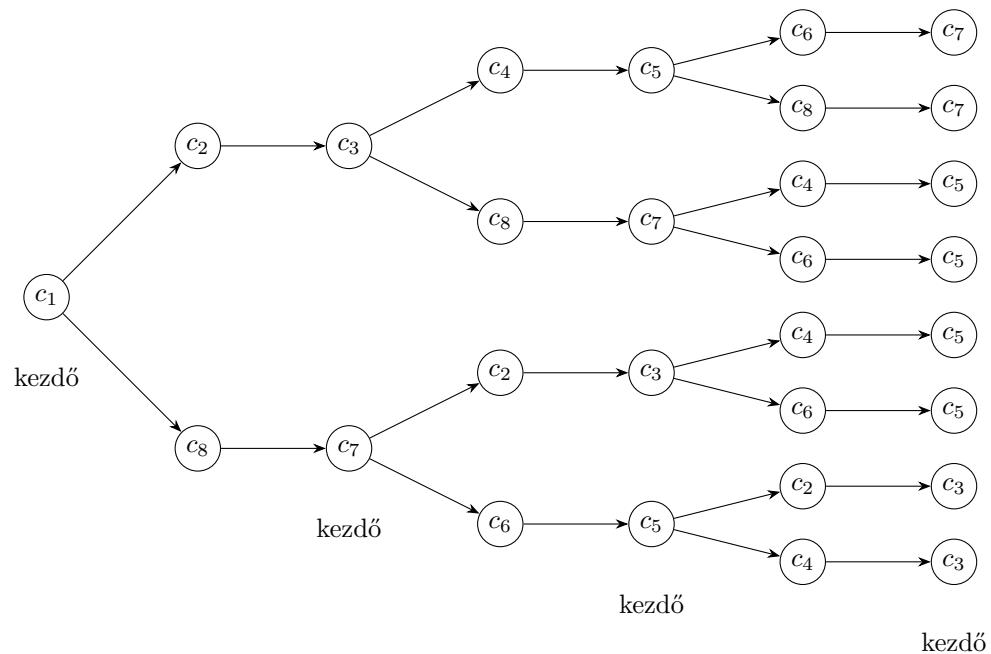


Kezdő (összesen 30): 7 8 6 9

Másik (összesen 28): 

A kezdő játékos ki tudja kényszeríteni, hogy a másik csak a páros indexű  $c_i$ -ket és azt is hogy csak a páratlan indexű  $c_i$ -ket vegye el (az összeset).

Pl. a páros indexekre:



Ezek után a nyerő stratégia:

$$S_{ps} := c_2 + c_4 + c_6 + \cdots + c_{2n}$$

$$S_{plan} := c_1 + c_3 + c_5 + \cdots + c_{2n-1}$$

Ha  $S_{ps} \leq S_{plan}$ , akkor a kezdő minden páratlan indexű érméket vegye el, különben minden páros indexűet.

Példánkban:

$$S_{ns} = 8 + 7 + 8 + 7 = 30$$

$$S_{plan} = 7 + 6 + 9 + 6 = 28$$

A kezdőnek a páros indexű érméket kell elvenni.

Optimális ez? Nem feltétlenül.

Új kérdés: milyen stratégiát kövessen a kezdő, hogy a különbség a javára maximális legyen? Az előző példánál a második játékosnak (globálisan) nem volt mozgástere. Általában ez nincs így.

Optimális stratégia a kezdő számára: bárhogy játszik a másik, a különbség mindenképp meglesz. Aprópénzre váltva: a másik "optimális stratégiája" mellett is (a másik játékos optimális stratégiája: minimalizálni akarja a különbséget).

## 7. gyakorlat (2025. október 22.)

### 7.1. Dinamikus programozás

#### Mátrixok optimális zárójelezése

$A_1 A_2 \cdots A_n$

A szorzat nem függ a zárójelezéstől, de a kiszámításhoz szükséges elemi szorzások száma általában nagyon is.

**Példa** (zh. feladat):

$A_1 : 25 \times 30$

$A_2 : 30 \times 10$

$A_3 : 10 \times 5$

$A_4 : 5 \times 10$

$A_5 : 10 \times 15$

$A_6 : 15 \times 20$

Feladat:  $A_1 A_2 A_3 A_4 A_5 A_6$  optimális zárójelezése.

Megoldás: két  $6 \times 6$ -os táblázat

0	7500	5250	6500	7875	10000
	0	1500	3000	4500	6750
		0	500	1500	3250
			0	750	2250
				0	3000
					0

$l[i, j]$

	1	1	3	3	3
		2	3	3	3
			3	3	3
				4	5
					5

$k[i, j]$  ("az a bizonyos  $k$ ")

$$\begin{aligned} l[1,1] &= 0 & l[1,2] &= 25 \times 30 \times 10 = 7500 \\ l[2,2] &= 0 & l[2,3] &= 30 \times 10 \times 5 = 1500 \\ l[3,3] &= 0 & l[3,4] &= 10 \times 5 \times 10 = 500 \\ l[4,4] &= 0 & l[4,5] &= 5 \times 10 \times 15 = 750 \\ l[5,5] &= 0 & l[5,6] &= 10 \times 15 \times 20 = 3000 \\ l[6,6] &= 0 \end{aligned}$$

**Általános formula:**

$$l[i, j] = \min_{i \leq k \leq j-1} \{ l[i, k] + l[k+1, j] + \underbrace{q_{i-1} q_k q_j}_{\text{dimenziók}} \}$$

$k[i, j]$  = az a  $k$ , ahol a fenti zárójeles összeg a legkisebb.

**"1. lépés: átlós kitöltés"**: amikor egy adott  $l[i, j]$ -t számítjuk, már minden  $l[i, k]$  és  $l[k+1, j]$  ismert.

$$l[1, 3] = \min \left\{ l[1, \boxed{1}] + l[2, 3] + 25 \cdot 30 \cdot 5 = \boxed{5250} \right. \\ \left. l[1, 2] + l[3, 3] + 25 \cdot 10 \cdot 5 = \boxed{8750} \right\}$$

$$l[2, 4] = \min \left\{ l[2, \boxed{2}] + l[3, 4] + 30 \cdot 10 \cdot 10 = \boxed{3500} \right. \\ \left. l[2, \boxed{3}] + l[4, 4] + 30 \cdot 5 \cdot 10 = \boxed{3000} \right\}$$

$$l[3,5] = \min \left\{ \begin{array}{l} l[3,\boxed{3}] + l[4,5] + 10 \cdot 5 \cdot 15 = \boxed{1500} \\ l[3,4] + l[5,5] + 10 \cdot 10 \cdot 15 = 2000 \end{array} \right\}$$

$$l[4,6] = \min \left\{ \begin{array}{l} l[4,4] + l[5,6] + 5 \cdot 10 \cdot 20 = 4000 \\ l[4,\boxed{5}] + l[6,6] + 5 \cdot 15 \cdot 20 = \boxed{2250} \end{array} \right\}$$

$$l[1,4] = \min \left\{ \begin{array}{l} l[1,1] + l[2,4] + 25 \cdot 30 \cdot 10 = 10500 \\ l[1,2] + l[3,4] + 25 \cdot 10 \cdot 10 = 10500 \\ l[1,\boxed{3}] + l[4,4] + 25 \cdot 5 \cdot 10 = \boxed{6500} \end{array} \right\}$$

Ha minden sor egyenlő, akkor bármelyiket választhatjuk minimumnak, majd a backtracking során látjuk, hogy ekkor több optimális zárójelezés is létezik.

$$l[2,5] = \min \left\{ \begin{array}{l} l[2,2] + l[3,5] + 30 \cdot 10 \cdot 15 = 6000 \\ l[2,\boxed{3}] + l[4,5] + 30 \cdot 5 \cdot 15 = \boxed{4500} \\ l[2,4] + l[5,5] + 30 \cdot 10 \cdot 15 = 7500 \end{array} \right\}$$

$$l[3,6] = \min \left\{ \begin{array}{l} l[3,\boxed{3}] + l[4,6] + 10 \cdot 5 \cdot 20 = \boxed{3250} \\ l[3,4] + l[5,6] + 10 \cdot 10 \cdot 20 = 5500 \\ l[3,5] + l[6,6] + 10 \cdot 15 \cdot 20 = 4500 \end{array} \right\}$$

$$l[1,5] = \min \left\{ \begin{array}{l} l[1,1] + l[2,5] + 25 \cdot 30 \cdot 15 = 15750 \\ l[1,2] + l[3,5] + 25 \cdot 10 \cdot 15 = 12750 \\ l[1,\boxed{3}] + l[4,5] + 25 \cdot 5 \cdot 15 = \boxed{7875} \\ l[1,4] + l[5,5] + 25 \cdot 10 \cdot 15 = 10250 \end{array} \right\}$$

$$l[2,6] = \min \left\{ \begin{array}{l} l[2,2] + l[3,6] + 30 \cdot 10 \cdot 20 = 9250 \\ l[2,\boxed{3}] + l[4,6] + 30 \cdot 5 \cdot 20 = \boxed{6750} \\ l[2,4] + l[5,6] + 30 \cdot 10 \cdot 20 = 12000 \\ l[2,5] + l[6,6] + 30 \cdot 15 \cdot 20 = 13500 \end{array} \right\}$$

$$l[1,6] = \min \left\{ \begin{array}{l} l[1,1] + l[2,6] + 25 \cdot 30 \cdot 20 = 21750 \\ l[1,2] + l[3,6] + 25 \cdot 10 \cdot 20 = 15750 \\ l[1,\boxed{3}] + l[4,6] + 25 \cdot 5 \cdot 20 = \boxed{10000} \\ l[1,4] + l[5,6] + 25 \cdot 10 \cdot 20 = 14500 \\ l[1,5] + l[6,6] + 25 \cdot 15 \cdot 20 = 15375 \end{array} \right\}$$

**2. lépés: "korábbi szorzások":** optimális zárójelezése  $A_1A_2A_3$ -nak és  $A_4A_5A_6$ -nak

$$A_1A_2A_3 \rightarrow k[1,3] = 1 \rightarrow A_1(A_2A_3)$$

$$A_4A_5A_6 \rightarrow k[4,6] = 5 \rightarrow (A_4A_5)A_6$$

Szumma szummárumban:  $(A_1(A_2A_3))((A_4A_5)A_6)$

## 8. gyakorlat (2025. november 5.)

”Hasonló” feladatok

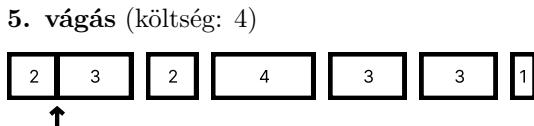
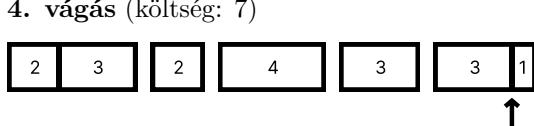
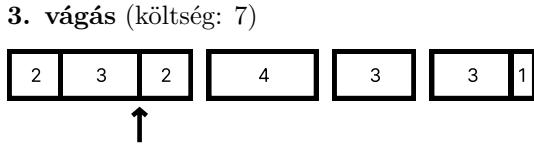
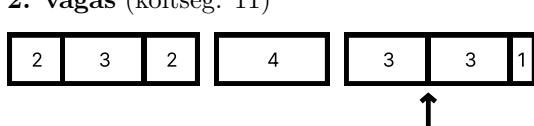
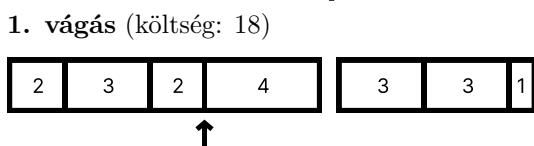
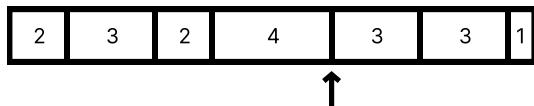
### 1. Rúd szétvágása

Adott egy  $l$  hosszúságú rúd, amit  $l_1, l_2, \dots, l_n$  hosszú darabokra akarunk felvágni, adott vágási pontknál.



Egy vágás úgy zajlik, hogy a már korábban ”levágott” darabok valamelyikét felemeljük a vágóeszközre, és az ottani vágási pontok valamelyikén vágunk. Egy ilyen vágás költsége legyen az épp vágott darab részeinek összege.

Példa:



Költség összesen: 52

Van ennél kisebb összköltségű vágássorozat? Mi a legkisebb összköltség?

Vegyük észre az analógiát az optimális zárójelzés feladattal! Várhatóan DP egy alkalmas módszer lesz a megoldásra, analóg módon a zárójelzéshez.

(margó szélére: ha  $n$  darabra vágunk, összesen  $(n-1)!$  lehetőség van erre, persze ezek közül nem minden ”lényegesen különböző”)

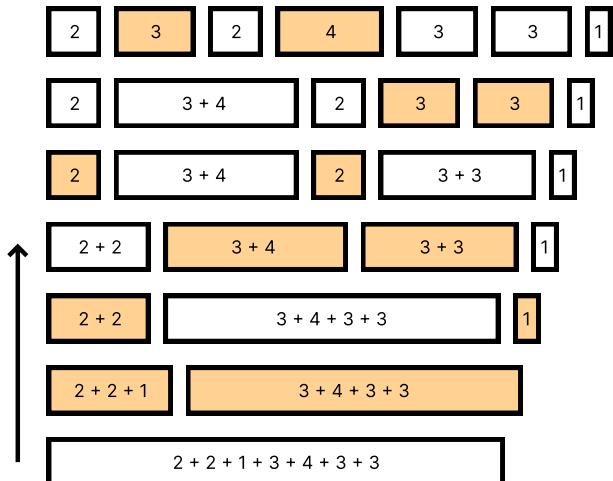
Részproblémák → ”infixek optimális szétvágása”

HF. részletek kidolgozása

**Változat:** a vágási pontokat is nekünk kell meghatározni (persze  $l = l_1 + l_2 + \dots + l_n$  továbbra is). A példát tekintve itt megengedett első vágásnak a bal szélről levágni egy 4 hosszú darabot.

Érdemes visszafelé elképzelni a vágássorozatot (”ragasztások sorozata”).

Példánkban:



Mohó heurisztika: mindenkor a két legrövidebb darabot ragasztjuk összeadjuk. Bizonyítani kell, hogy ezek a lokálisan optimális lépések elvezetnek a globális optimumhoz.

Nagyon hatékony  $\rightarrow$  kupaccal  $\mathcal{O}(n \log n)$  lépés (eredeti változat DP-vel  $\rightarrow \mathcal{O}(n^3)$ ).

Nagyon hasonló ez az egész az információtömörítésnél megismert Huffman algoritmushoz (Huffman-kódhoz).

## 2. Optimális bináris keresőfa

Általában bináris keresőfákat dinamikusan változó adathalmazokra alkalmazunk. Itt most statikus lesz az adathalmaz ezzel szemben. Itt ismert az adatelemek keresési gyakorisága is. Olyan bináris keresőfa megépítése a cél az adatelemekből, ahol a keresési út hosszának várható értéke minimális.

## 9. gyakorlat (2025. november 12.)

### 9.1. "Sztringológia"

Karaktersorozatok hasonlósága

- Levenshtein távolság (optimális szekvenciaillesztés)
- leghosszabb közös részsorozat (LKR)

Mindkét esetben DP

$X = (x_1, x_2, \dots, x_m)$  és  $Y = (y_1, y_2, \dots, y_n) \rightarrow \mathcal{O}(mn)$

**Leghosszabb közös részsorozat (LKR, előadáson "vázlatosan" volt)**

**Optimális részstruktúra tulajdonság**

$X_i = (x_1, x_2, \dots, x_i)$  és  $Y_j = (y_1, y_2, \dots, y_j)$  prefixek LKR-jének meghatározása van éppen terítéken.

Tfh.  $X_i$  és  $Y_j$  egy LKR-je  $Z_h = (z_1, z_2, \dots, z_h)$

Ekkor

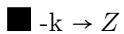
- (A) Ha  $x_i = y_j$ , akkor  $z_h = x_i = y_j$  és  $Z' = (z_1, z_2, \dots, z_{h-1})$  egy LKR  $X_{i-1}$ -hez és  $Y_{j-1}$ -hez. A másik két esetben  $x_i \neq y_j$ , így biztosan  $z_h \neq x_i$  vagy  $z_h \neq y_j$ .
- (B) Ha  $z_h \neq x_i$ , akkor  $Z$  egy LKR  $X_{i-1}$ -hez és  $Y_j$ -hez.
- (C) Ha  $z_h \neq y_j$ , akkor  $Z$  egy LKR  $X_i$ -hez és  $y_{j-1}$ -hez.

**Bizonyítás**

(A) Itt két állítás van! Kezdjük azzal, hogy  $z_h = x_i = y_j$ . Indirekt tfh. ez nem igaz, vagyis  $z_h \neq x_i, y_j$ .

  $X_i$

  $Y_j$

 -k →  $Z$

Ám ekkor  $Z$  biztos nem LKR  $x_i$ -hez és  $Y_j$ -hez, hiszen  $Z$  végére írva az  $x_i = y_j$  karaktert egy  $Z$ -nél hosszabb közös részsorozatát látjuk  $X_i$ -nek és  $Y_j$ -nek.

Ezek után jöjjön a második rész. Most  $Z'$  nyilván közös részsorozata  $X_{i-1}$ -nek és  $Y_{j-1}$ -nek. Indirekt tfh.  $Z'$  nem LKR  $X_{i-1}$ -hez és  $Y_{j-1}$ -hez, vagyis van olyan  $Z'$ -nél hosszabb  $Z''$  sorozat, amely közös részsorozata  $X_{i-1}$ -nek és  $Y_{j-1}$ -nek. Ha most  $Z''$  végére írjuk az  $x_i = y_j$  karaktert, az egy  $Z$ -nél hosszabb közös részsorozata lesz  $X_i$ -nek és  $Y_j$ -nek, **ellentmondás**.

(B) Most egyszerűen  $Z$  közös részsorozata  $X_{i-1}$ -nek és  $Y_j$ -nek, másrészt ha lenne egy  $Z$ -nél hosszabb közös részsorozata  $X_{i-1}$ -nek és  $Y_j$ -nek, az egy  $Z$ -nél hosszabb közös részsorozata lenne  $X_i$ -nek és  $Y_j$ -nek, ami nem lehet.

(C) analóg módon (B)-hez.

**Rekurzió**

$$l[i, j] = \min \begin{cases} 0 & \text{ha } i = 0 \text{ vagy } j = 0 \\ l[i-1, j-1] + 1 & \text{ha } i, j \geq 1 \text{ és } x_i = y_j \\ \max\{l[i-1, j], l[i, j-1]\} & \text{ha } i, j \geq 1 \text{ és } x_i \neq y_j \end{cases}$$

**Példa (zh. feladat):**

$X = (A, B, C, B, C, A, B, B, A, C)$

$Y = (B, A, A, C, B, B, A, A, C, B, C)$

**Kitöltés menete:**

Összehasonlítjuk a sorhoz és az oszlophoz tartozó két betűt.

- Ha megegyeznek (rekurzív képlet 2. esete): az átlósan egygyel fentebb és vízszintesen előző elem + 1
- Ha különböznek (rekurzív képlet 3. esete): a sorban előtte lévő elem és az oszlopban felette lévő elem közül a maximális (ha egyenlők akkor a fölötté lévő)

A nyilak mutatják, hogy az adott cellában lévő elemet honnan vettük. Ezek mentén visszafelé haladva meghatározható a LKR.

	B	A	A	C	B	B	A	A	C	B	C	
	0	1	2	3	4	5	6	7	8	9	10	11
0	0	0	0	0	0	0	0	0	0	0	0	0
A 1	0	$\uparrow$ 0	$\nwarrow$ 1	$\nwarrow$ 1	$\leftarrow$ 1	$\leftarrow$ 1	$\leftarrow$ 1	$\nwarrow$ 1	$\nwarrow$ 1	$\leftarrow$ 1	$\leftarrow$ 1	$\leftarrow$ 1
B 2	0	$\nwarrow$ 1	$\uparrow$ 1	$\uparrow$ 1	$\uparrow$ 1	$\nwarrow$ 2	$\nwarrow$ 2	$\leftarrow$ 2	$\leftarrow$ 2	$\leftarrow$ 2	$\nwarrow$ 2	$\leftarrow$ 2
C 3	0	$\uparrow$ 1	$\uparrow$ 1	$\uparrow$ 1	$\nwarrow$ 2	$\uparrow$ 2	$\uparrow$ 2	$\uparrow$ 2	$\nwarrow$ 3	$\leftarrow$ 3	$\leftarrow$ 3	$\nwarrow$ 3
B 4	0	$\nwarrow$ 1	$\uparrow$ 1	$\uparrow$ 1	$\uparrow$ 2	$\nwarrow$ 3	$\nwarrow$ 3	$\leftarrow$ 3	$\leftarrow$ 3	$\uparrow$ 3	$\nwarrow$ 4	$\leftarrow$ 4
C 5	0	$\uparrow$ 1	$\uparrow$ 1	$\uparrow$ 1	$\nwarrow$ 2	$\uparrow$ 3	$\uparrow$ 3	$\uparrow$ 3	$\uparrow$ 3	$\uparrow$ 4	$\uparrow$ 4	$\nwarrow$ 5
A 6	0	$\uparrow$ 1	$\nwarrow$ 2	$\nwarrow$ 2	$\uparrow$ 2	$\uparrow$ 3	$\uparrow$ 3	$\nwarrow$ 4	$\nwarrow$ 4	$\uparrow$ 4	$\uparrow$ 4	$\uparrow$ 5
B 7	0	$\nwarrow$ 1	$\uparrow$ 2	$\uparrow$ 2	$\uparrow$ 2	$\nwarrow$ 3	$\nwarrow$ 4	$\uparrow$ 4	$\uparrow$ 4	$\uparrow$ 4	$\nwarrow$ 5	$\uparrow$ 5
B 8	0	$\nwarrow$ 1	$\uparrow$ 2	$\uparrow$ 2	$\uparrow$ 2	$\nwarrow$ 3	$\nwarrow$ 4	$\uparrow$ 4	$\uparrow$ 4	$\uparrow$ 4	$\nwarrow$ 5	$\uparrow$ 5
A 9	0	$\uparrow$ 1	$\nwarrow$ 2	$\nwarrow$ 3	$\leftarrow$ 3	$\uparrow$ 3	$\uparrow$ 4	$\nwarrow$ 5	$\nwarrow$ 5	$\leftarrow$ 5	$\uparrow$ 5	$\uparrow$ 5
C 10	0	$\uparrow$ 1	$\uparrow$ 2	$\uparrow$ 3	$\nwarrow$ 4	$\leftarrow$ 4	$\uparrow$ 4	$\uparrow$ 5	$\uparrow$ 5	$\nwarrow$ 6	$\leftarrow$ 6	$\nwarrow$ 6

**LKR hossz:** 6

**LKR:** ABBCBC (van más is!)

Az előbbi  $X$  és  $Y$  Levenshtein távolsága.

Itt még kell valami:

$$g = 3$$

$$c["p", "q"] = 5 \text{ (a csere } 5 \text{ költségű)}$$

$$c["p", "p"] = 0 \text{ (a nem-csere } 0 \text{ költségű)}$$

## 10. gyakorlat (2025. november 19.)

### 10.1. Előadás végéhez némi háttér

$6^{1589}$  mennyi maradékot ad 31-gyel osztva? Naiv módszer:  $\underbrace{6 \cdot 6 \cdot 6 \cdot 6 \cdots \cdot 6}_{1589\text{-szer}} \rightarrow 1588$  szorzás

31-gyel oszthatóság

$$\begin{aligned} & \overbrace{6 \cdot 6 \cdot 6 \cdot 6 \cdot 6 \cdot \cdots \cdot 6}^{\text{31 gyerek}} \\ & 36 \bmod 31 = 5 \\ & \overbrace{5}^{\text{5 gyerek}} \\ & \overbrace{30 \bmod 31 = 30}^{\text{30 gyerek}} \\ & \overbrace{30 \cdot 6 = 180 \bmod 31 = 25}^{\text{30 gyerek}} \\ & 25 \cdot 6 = 150 \bmod 31 = 26 \end{aligned}$$

Hogyan lehet ezt felgyorsítani?

$6^{1589} \rightarrow$  kitevő kettes számrendszerben

$$\begin{array}{r} 1589 : 2 = 794 \\ 18 \quad \quad \quad 19 \quad \quad \quad 19 \\ 09 \quad \quad \quad 14 \quad \quad \quad 17 \\ \boxed{1} \quad \quad \quad \boxed{0} \quad \quad \quad \boxed{1} \end{array}$$

$$\begin{array}{r} 198 : 2 = 99 \\ 18 \quad \quad \quad 19 \quad \quad \quad 09 \\ \boxed{0} \quad \quad \quad \boxed{1} \quad \quad \quad \boxed{1} \end{array}$$

$$\begin{array}{r} 24 : 2 = 12 \\ 04 \quad \quad \quad 12 : 2 = 6 \quad \quad \quad 6 : 2 = 3 \\ \boxed{0} \quad \quad \quad \boxed{0} \quad \quad \quad \boxed{0} \end{array}$$

$$\begin{array}{r} 3 : 2 = 1 \\ \boxed{1} \quad \quad \quad \boxed{1} \end{array}$$

$$\begin{array}{cccccccccccc} 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ \hline 1024 & 512 & 256 & 128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \end{array}$$

$$1024 + 512 + 32 + 16 + 4 + 1 = 1589 \text{ (ahol } 1\text{-esek vannak)}$$

Észrevétel:  $\mathcal{O}(\log \text{kitevő})$  "elemi" művelet a kettek számrendszerre átírás

Ezek után jön a piszkos trükk:  $6^{1589} = 6^{1024+512+32+16+4+1} = 6^{1024} \cdot 6^{512} \cdot 6^{32} \cdot 6^{16} \cdot 6^4 \cdot 6^1$

$$6 \bmod 31 = 6$$

$$6^2 \bmod 31 = 5$$

$$6^4 = (6^2)^2 \bmod 31 = 5^2 \bmod 31 = 25 \bmod 31 = 25$$

$$6^8 = (6^4)^2 \bmod 31 = 25^2 \bmod 31 = 225 \bmod 31 = 8$$

$$6^{16} = (6^8)^2 \bmod 31 = 8^2 \bmod 31 = 64 \bmod 31 = 2$$

$$6^{32} = (6^{16})^2 \bmod 31 = 2^2 \bmod 31 = 4$$

$$6^{64} = (6^{32})^2 \bmod 31 = 4^2 \bmod 31 = 16$$

$$6^{128} = (6^{64})^2 \bmod 31 = 16^2 \bmod 31 = 256 \bmod 31 = 8$$

$$6^{256} = (6^{128})^2 \bmod 31 = 8^2 \bmod 31 = 64 \bmod 31 = 2$$

$$6^{512} = (6^{256})^2 \bmod 31 = 2^2 \bmod 31 = 4$$

$$6^{1024} = (6^{512})^2 \bmod 31 = 4^2 \bmod 31 = 16$$

$$\begin{aligned} & \text{Így } 6^{1024} \cdot 6^{512} \cdot 6^{32} \cdot 6^{16} \cdot 6^4 \cdot 6^1 \bmod 31 = \\ & 16 \cdot 4 \cdot 4 \cdot 2 \cdot 25 \cdot 6 \bmod 31 \end{aligned}$$

$$\begin{array}{r}
 16 \cdot 4 \cdot 4 \cdot 2 \cdot 25 \cdot 6 \\
 \underbrace{2} \\
 \underbrace{8} \\
 \hline
 16 \\
 \hline
 \underbrace{16 \cdot 25 = 400 \rightarrow 28} \\
 \hline
 \underbrace{28 \cdot 6 = 168 \rightarrow 13}
 \end{array}$$

$\Rightarrow 6^{1589} \equiv \underbrace{13}_{\text{maradék}} \pmod{31}$

Ez megint  $\mathcal{O}(\log \text{kitevő})$  elemi művelet, azaz  $\Theta(\text{kitevő})$  elemi művelet  $\mathcal{O}(\log \text{kitevő})$  elemi műveletre csökkent.

Mi ennek a jelentősége?

Kitevő, mint input mérete  $\mathcal{O}(\log \text{kitevő})$ . Ehhez képest  $\Theta(\text{kitevő})$  exponenciálisan nagy. Így a naiv algoritmus exponenciális a bemenet méretében. A "piszkos trükk" ezt viszi le lineárisra (a bemenet méretében).

Előadásra visszatérve:  $a_1, a_2, \dots, a_n \rightarrow B$

Részhalmaozösszeg: mindenkit egyszer használhatunk

Pénzváltás: többször is. Nyilván  $a_1$ -et maximum  $\left\lfloor \frac{B}{a_1} \right\rfloor$ ,  $a_2$ -t maximum  $\left\lfloor \frac{B}{a_2} \right\rfloor$ , stb. alkalommal.

Pénzváltás naiv visszavezetése a részhalmaozösszegre:

$$\underbrace{a_1, \dots, a_1}_{\left\lfloor \frac{B}{a_1} \right\rfloor}, \underbrace{a_2, \dots, a_2}_{\left\lfloor \frac{B}{a_2} \right\rfloor}, \underbrace{a_n, \dots, a_n}_{\left\lfloor \frac{B}{a_n} \right\rfloor} \rightarrow B$$

Probléma: az input mérete a bemenet méretében, különös tekintettel  $B$ -re exponenciálisan nagy lett. Nyilván azok az  $a_i$ -k, amelyek nagyobban  $B$ -nél nem sok vizet zavarnak, így a bemenet mérete  $\mathcal{O}(n \log B)$  (nagyon precízen  $\mathcal{O}((n+1) \log B)$ ) a részhalmaozösszegnél.

Ha most  $n$  lineáris  $B$ -ben, akkor ez  $B \cdot \log B$  nagyságrendű, ami  $2^{\log_2 B} \cdot B$ , és ez exponenciálisan nagy  $B$  méretében.

Ez azt jelenti, hogy hiába is lenne mondjuk egy kvadratikus algoritmusunk a részhalmaozösszegre (senki nem ismer ilyet), az exponenciális ideig futna a pénzváltásból előállított inputon.

A piszkos trükk itt:  $a_1, \dots, a_1 \rightarrow$  amit ezekből elő tudunk állítani összegként, azt mind elő tudjuk állítani sokkal

kevesebb számból is:  $a_1, 2a_1, 4a_1, 8a_1, \dots, 2^{\lfloor \log_2 B \rfloor} a_1$  Ezzel a  $\left\lfloor \frac{B}{a_1} \right\rfloor$  db szám lecsökken  $\lfloor \log_2 B \rfloor$  darabra.

Így a transzformált bemenet már nem exponenciális  $\log B$ -hez képest (igazából lineáris)!

## 10.2. "Sztringológia" (folytatás)

### Optimális szekvenciaillesztés

**Példa** (zh. feladat):

Kitöltés menete:

- Ha megegyeznek: átlósan átvesszük az eggyel fentebb és vízszintesen előző elemet
- Ha különbözőek:  $\min((\text{felette lévő} + 3), (\text{átlós elem} + 5), (\text{balra lévő} + 3))$

Itt a minimális értéket akarjuk átvenni. Megállapodás: "óramutató járásával ellentétesen" (egyenlőségkor első-sorban felülről, aztán átlóból, majd vízszintesen)

Átlós út: lecseréljük az egyik betűt a másikra.

Függőleges út: beszúrom a másik betűt, törlöm az elsőt.

Az átlós út hatékonyabb, mint a függőleges út.

	B	A	A	C	B	B	A	A	C	B	C
0	0	3	6	9	12	15	18	21	24	27	30
A	1	2	3	4	5	6	7	8	9	10	11
0	0	3	6	9	12	15	18	21	24	27	33
A	3	↖ 5	↖ 3	↖ 6	← 9	← 9	← 15	↖ 18	↖ 21	← 24	← 27
B	6	↖ 3	↑ 6	↖ 8	↖ 11	↖ 9	↖ 12	← 15	← 18	← 21	↖ 24
C	9	↑ 6	↖ 8	↑ 11	↖ 8	← 11	↖ 14	↖ 17	↖ 20	↖ 18	← 21
B	12	↑ 9	↑ 11	↖ 13	↑ 11	↖ 8	↖ 11	← 14	← 17	← 20	↖ 18
C	15	↑ 12	↑ 14	↑ 16	↖ 13	↑ 11	↖ 13	↖ 16	↖ 19	↖ 17	← 20
A	18	↑ 15	↖ 12	↖ 14	↑ 16	↑ 14	↑ 16	↖ 13	↖ 16	← 19	↖ 22
B	21	↑ 18	↑ 15	↑ 17	↑ 19	↖ 16	↖ 14	↑ 16	↖ 18	↖ 21	135 19
B	24	↑ 21	↑ 18	↑ 20	↑ 22	↑ 19	↖ 16	↑ 19	↑ 21	↖ 23	↖ 21
A	27	↑ 24	↑ 21	↖ 18	← 21	↑ 22	↑ 19	↖ 16	↖ 19	← 22	↑ 24
C	30	↑ 27	↑ 24	↑ 21	↖ 18	← 21	↑ 22	↑ 19	↖ 21	↖ 19	← 22
											↖ 24

Optimális szekvenciaillesztés:  $\{(1, 2), (2, 3), (3, 4), (4, 6), (5, 7), (6, 8), (7, 9), (8, 10), (10, 11)\}$

-	A	B	C	-	B	C	A	B	B	A	C
B	A	A	C	B	B	A	A	C	B	-	C
ins	rep	ins	rep	rep	rep	del					

(*ins* - beszúrás, *rep* - csere, *del* - törlés)

Még egy jópofa sztringes feladat (egy időben a programozási versenyek egyik kedvence volt)

Adott egy  $T$  karakterszorozat, keressünk ebben egy leghosszabb olyan karakterszorozatot, amely balról jobbra olvasva ugyanaz, mint jobbról balra olvasva (palindrom)

Pl.  $T \rightarrow C [A] D [B] C E [B] E [A] B$

Ez egy részsorozat, de nem a leghosszabb. Ez is egy részsorozat, a leghosszabb.

$T$ -ből készítünk palindromot a legkevésesebb karakter törlésével!

Hasonló feladat:  $T$ -ből készítünk palindromot a legkevésesebb karakter beszúrásával!

# 11. gyakorlat (2025. november 19.)

## 11.1. Hátizsák feladat

Példa (zh. feladat):

i	1	2	3	4	5	6	7
$w_i$	4	5	3	1	1	4	3
$v_i$	40	60	10	10	3	20	60

$W = 10$  (hátizsák kapacitása)

Részfeladatok: az első néhány tárggyal kell megtölteni optimálisan kisebb hátizsákokat.

Kitöltés menete:

Az oszlop a hátizsák kapacitása, a sor a hátizsákba rakott érték. minden sorhoz tartozik egy tárgy (az  $i$ . sorhoz  $t_i$  és  $v_i$ ) egy súly/érték párral. Megnézzük milyen érték van a jelenlegi cellától eggyel fentebből cellában. Visszalépünk abban a sorban súly cellát. Amelyik cellához jutottunk, azt hozzáadjuk az értékhez. Az összeg és a cella feletti érték közül a nagyobb lesz a cella tartalma.

$$\text{Előadásról: } h[i, j] = \max \begin{cases} v_i + h[i - 1, j - w_i] \\ h[i - 1, j] \end{cases}$$

Ha pl. az sorhoz tartozó tárgy súlya 4, akkor a 0, 1, 2, 3 indexű oszlopok csak másolódnak, mert nem tudunk 4-öt visszalépni a felette lévő sorban. Azaz, 0, 1, 2, 3 kapacitású hátizsákba nem fér bele a 4 súlyú tárgy. minden sorban azt nézzük, hogy a sorhoz tartozó tárgyat érdemes-e belerakni a hátizsákbba.

i\j	0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	0	0	0	0	0	0
4/40	1	0	0	0	40	40	40	40	40	40	40
5/60	2	0	0	0	40	60	60	60	60	max(40, (40+60))=100	100
3/10	3	0	0	0	10	40	60	60	70	100	100
1/10	4	0	10	10	10	40	60	70	70	100	110
1/3	5	0	10	13	13	40	60	70	73	100	110
4/20	6	0	10	13	13	40	60	70	73	100	110
3/60	7	0	10	13	60	70	73	73	100	120	130
											133

Eredmény leolvasása:

A jobb alsó sarokból indulunk szokás szerint. A 7. sorban még nőtt az érték, tehát a 7. tárgyra szükség van, ezt "beragasztjuk" a hátizsákbba. Ekkor már beragaszottunk 3 kapacitásnyi tárgyat a 10 kapacitású hátizsákbba. Az első 7 kapacitást kell még valahogy kitölteni. Ezért nézzük a 7. oszlopot és felfelé haladva a következő sort. Ez a cella az előző sorhoz képest nem javult, azaz a 6. tárgyra nincs szükség. Megyünk megint egy sorral fentebb, ugyanabban az oszlopbán. Ez javított az őt megelőző sorhoz képest, ezért erre a tárgyra szükség van. Ezt ismételjük tovább.

Optimális megoldás:  $t_2, t_4, t_5, t_7$