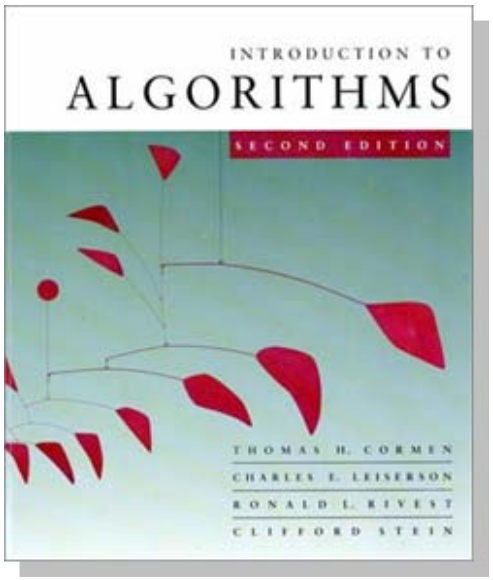
Algoritmalara Giriş

6.046J/18.401J



DERS 5

Alt Sınırları Sıralama

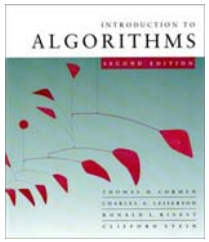
- Karar ağaçları

Doğrusal-Zaman Sıralaması

- Sayma sıralaması
- Taban sıralaması

Son ek: Delikli kartlar

Prof. Erik Demaine



Ne kadar hızlı sıralayabiliriz?

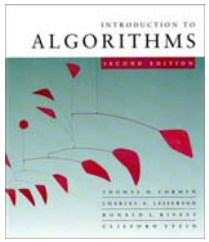
Şu ana kadar gördüğümüz tüm sıralama algoritmaları ***karşılaştırma sıralamalarıydı***: elemanların bağıl düzenlerini saptamakta yalnız karşılaştırma kullanırlar.

- *Örneğin*, araya yerleştirme, birleştirme sıralamaları, çabuk sıralama, yığın sıralaması.

Karşılaştırma sıralamalarında gördüğümüz en iyi en-kötü-durum koşma süresi $O(n \lg n)$ idi.

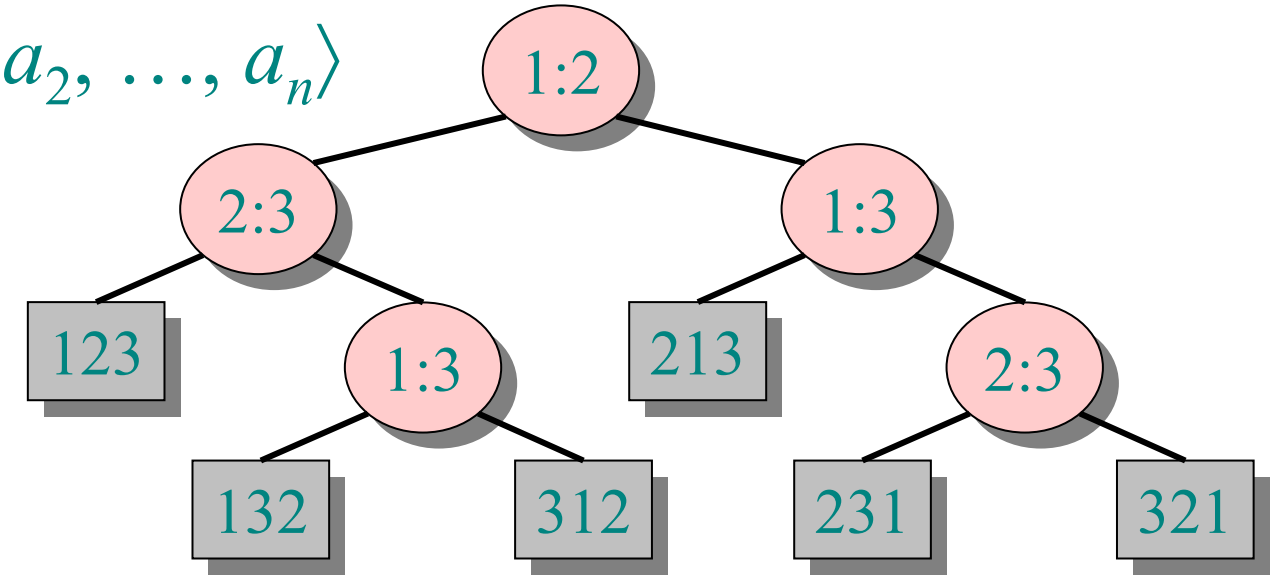
$O(n \lg n)$ elde edebileceğimizin en iyisi mi?

Karar ağaçları bu sorunun yanıtına yardımcı olur.



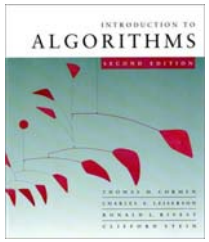
Karar-ağacı örneği

Sırala $\langle a_1, a_2, \ldots, a_n \rangle$



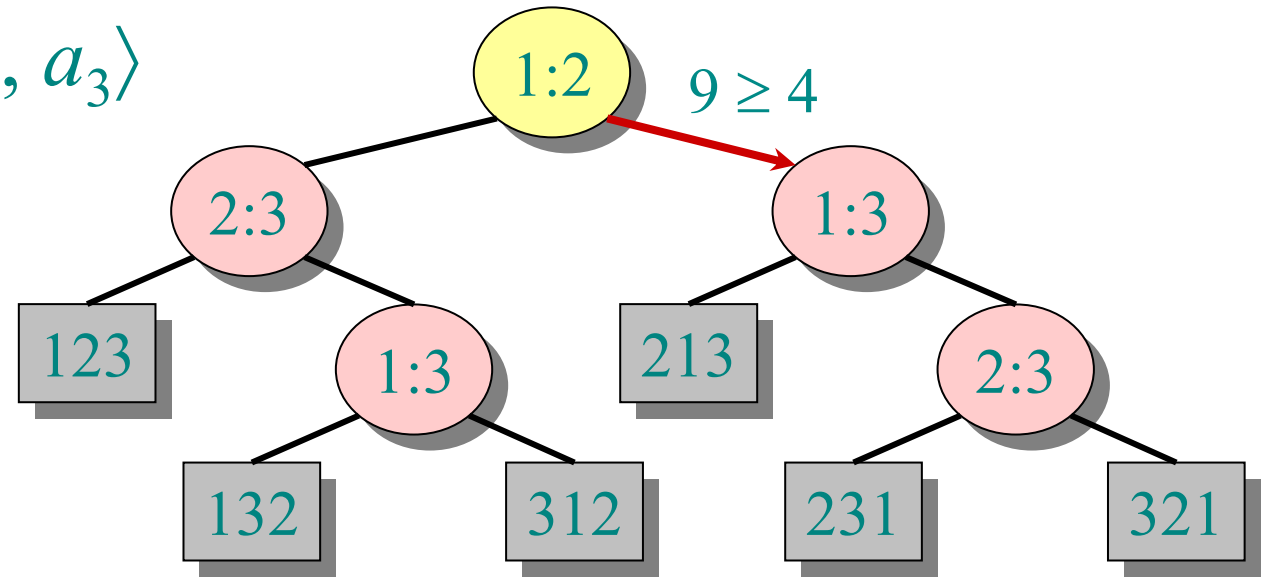
Her iç boğumun etiketlenmesi $i{:}j$; $i, j \in \{1, 2, \ldots, n\}$ için.

- Sol alt-ağaç, $a_i \le a_j$ ise, ardarda karşılaştırmaları gösterir.
- Sağ alt ağaç, $a_i \ge a_j$ ise, ardarda karşılaştırmaları gösterir.



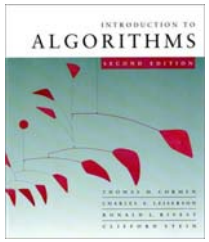
Karar-ağacı örneği

Sırala $\langle a_1, a_2, a_3 \rangle$
$= \langle 9, 4, 6 \rangle$:



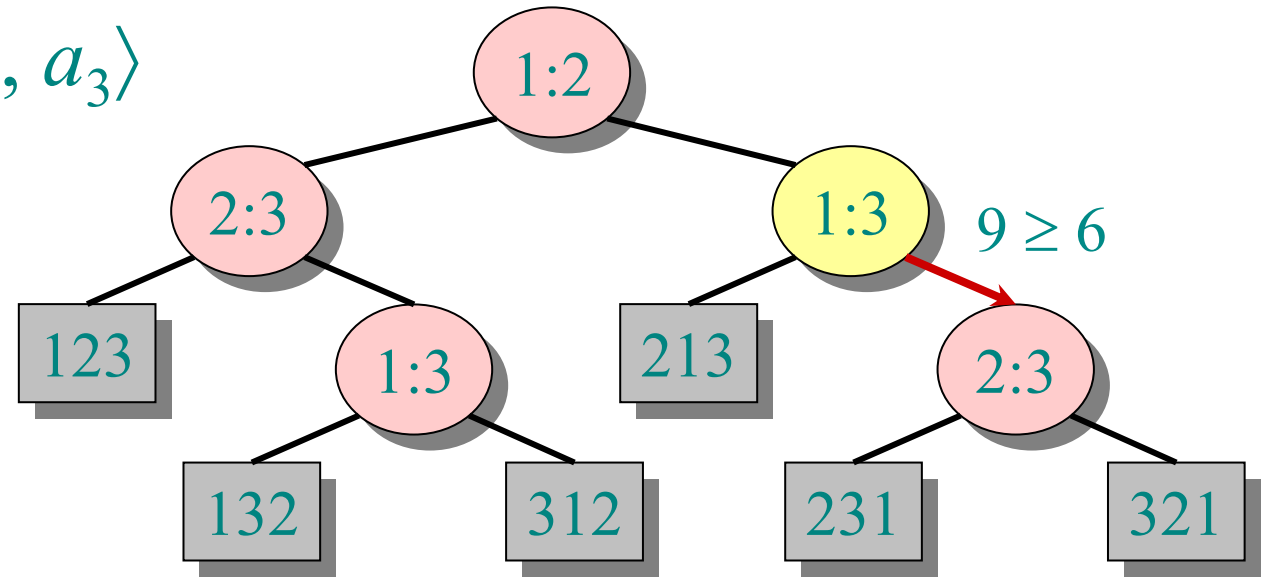
Her iç boğumun etiketlenmesi $i{:}j$; $i, j \in \{1, 2, \ldots, n\}$ için.

- Sol alt-ağaç $a_i \le a_j$ ise, ardarda karşılaştırmaları gösterir.
- Sağ alt-ağaç $a_i \ge a_j$ ise, ardarda karşılaştırmaları gösterir.



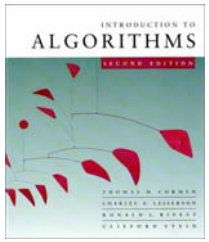
Karar-ağacı örneği

Sırala $\langle a_1, a_2, a_3 \rangle$ $= \langle\ 9, 4, 6\ \rangle$:



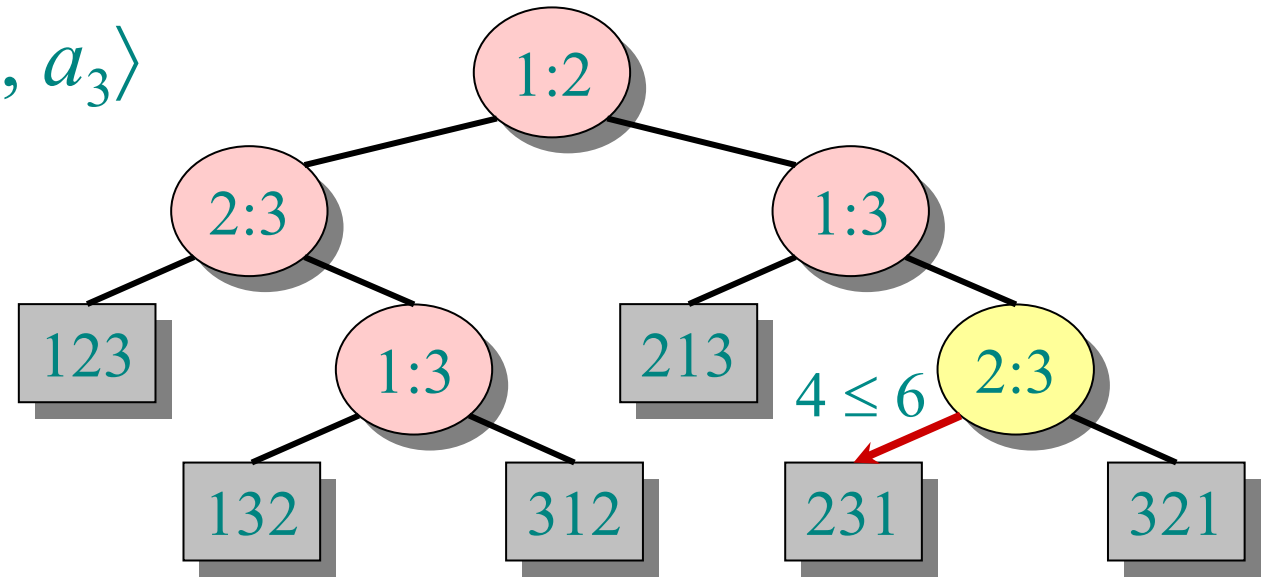
Her iç boğumun etiketlenmesi $i{:}j$; $i, j \in \{1, 2, \ldots, n\}$ için:

- Sol alt-ağaç $a_i \le a_j$ ise, ardarda karşılaştırmaları gösterir.
- Sağ alt-ağaç $a_i \ge a_j$ ise, ardarda karşılaştırmaları gösterir.



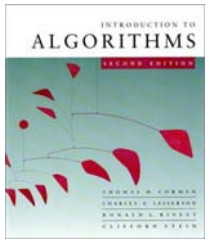
Karar-ağacı örneği

Sırala $\langle a_1, a_2, a_3 \rangle$
$= \langle 9, 4, 6 \rangle$:



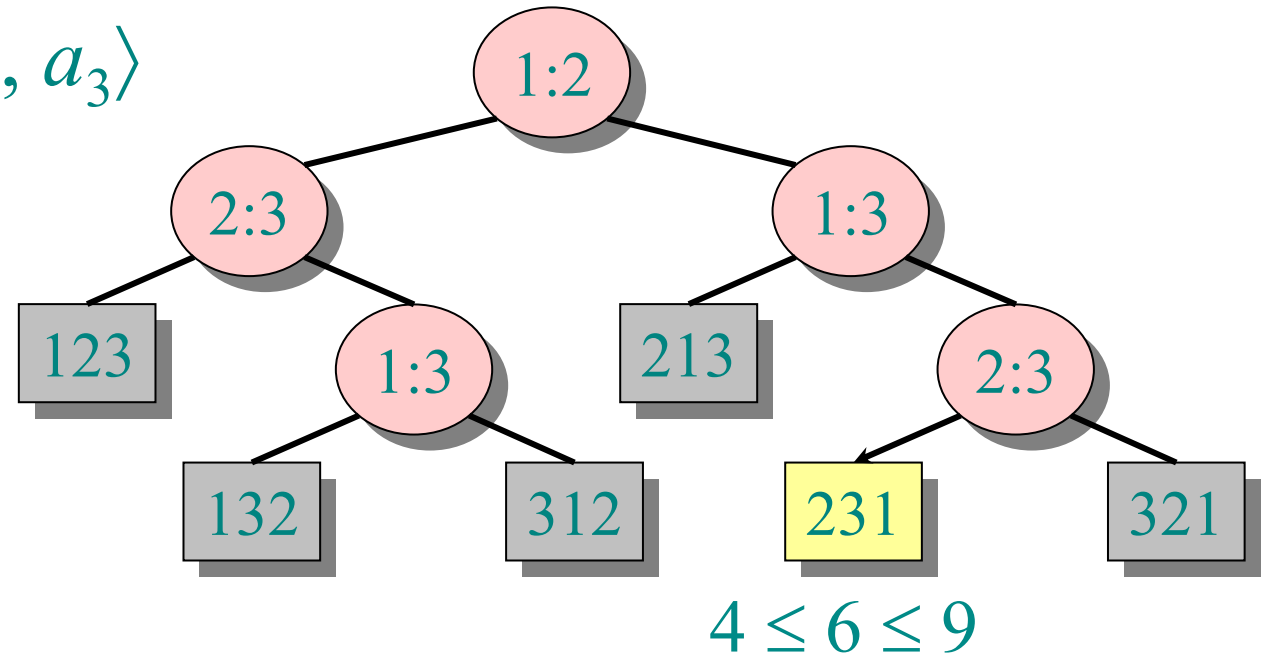
Her iç boğumun etiketlenmesi $i{:}j$; $i, j \in \{1, 2, \ldots, n\}$ için.

- Sol alt-ağaç $a_i \leq a_j$ ise, ardarda karşılaştırmaları gösterir.
- Sağ alt-ağaç $a_i \geq a_j$ ise, ardarda karşılaştırmaları gösterir.

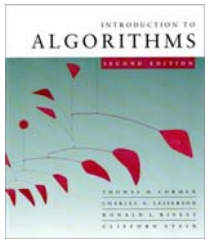


Karar-ağacı örneği

Sırala $\langle a_1, a_2, a_3 \rangle = \langle 9, 4, 6 \rangle$:



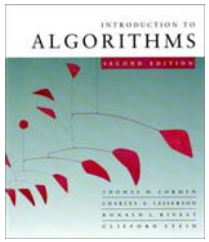
Her yaprakta $\langle \pi(1), \pi(2), \ldots, \pi(n) \rangle$ permütasyonu vardır bu $a_{\pi(1)} \le a_{\pi(2)} \le \cdots \le a_{\pi(n)}$ sıralamasının tamamlanmış olduğunu gösterir.



Karar-ağacı modeli

Bir karar ağacı her karşılaştırma sıralaması uygulanmasını modelleyebilir:

- Her n giriş boyutu için bir ağaç.
- Algoritmayı iki elemanı karşılaştırdığında bölünüyormuş gibi görün.
- Ağaç tüm olası komut izlerindeki karşılaştırmalar içerir.
- Algoritmanın koşma süresi = takip edilen yolun uzunluğu.
- En kötü-durum koşma süresi = ağacın boyu.

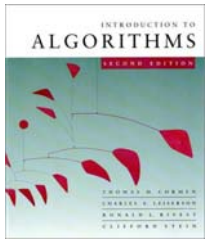


Karar-ağacı sıralamasında alt sınır

Teorem. n elemanı sıralayabilen bir karar-ağacının yüksekliği (boyu) $\Omega(n \lg n)$ olmalıdır.

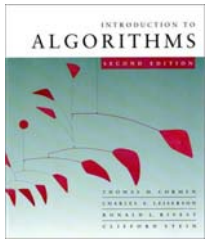
Kanıtlama. Ağacın $\geq n!$ yaprağı olmalıdır, çünkü ortada $n!$ olası permütasyon vardır. Boyu h olan bir ikili ağacın $\leq 2^h$ yaprağı olur. Böylece, $n! \leq 2^h$.

$$
\begin{aligned}
\therefore\ h &\geq \lg(n!) && (\lg \text{ monoton artışlı}) \\
&\geq \lg\left((n/e)^n\right) && (\text{Stirling'in formülü}) \\
&= n \lg n - n \lg e \\
&= \Omega(n \lg n)\,. \quad \square
\end{aligned}
$$



Karşılaştırma sıralamasında alt sınır

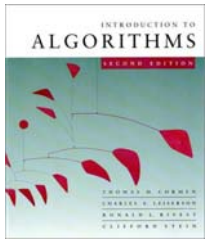
Doğal sonuç. Yığın sıralaması ve birleştirme sıralaması asimptotik olarak en iyi karşılaştırma sıralaması algoritmalarıdır. □



Doğrusal zamanda sıralama

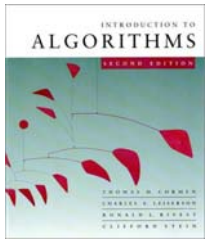
Sayma sıralaması: Elemanlar arası karşılaştırma yok.

- ***Girdi***: $A[1 \,..\, n]$, burada $A[j] \in \{1, 2, \ldots, k\}$.
- ***Çıktı***: $B[1 \,..\, n]$, sıralı.
- ***Yedek depolama***: $C[1 \,..\, k]$.

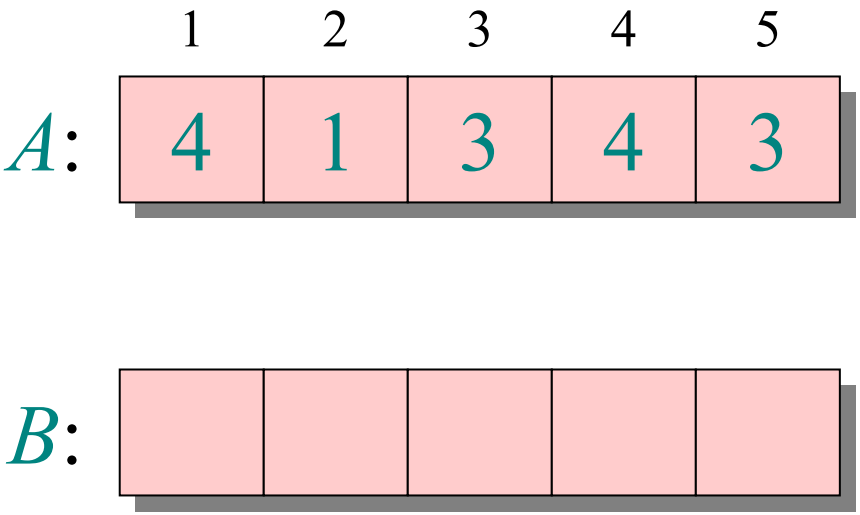


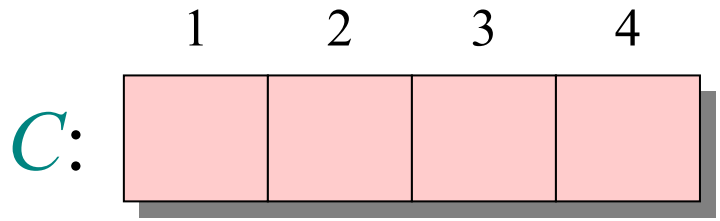
Sayma sıralaması

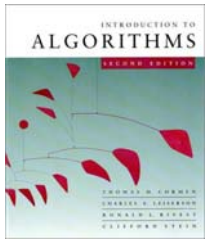
for $i \leftarrow 1$ **to** k
  do $C[i] \leftarrow 0$
for $j \leftarrow 1$ **to** n
  do $C[A[j]] \leftarrow C[A[j]] + 1$ $\triangleright$ $C[i] = |\{\text{key} = i\}|$
for $i \leftarrow 2$ **to** k
  do $C[i] \leftarrow C[i] + C[i–1]$ $\triangleright$ $C[i] = |\{\text{key} \le i\}|$
for $j \leftarrow n$ **down to** 1 (**down to 1**: 1'e inene kadar)
  do $B[C[A[j]]] \leftarrow \text{A}[j]$
    $C[A[j]] \leftarrow C[A[j]] – 1$



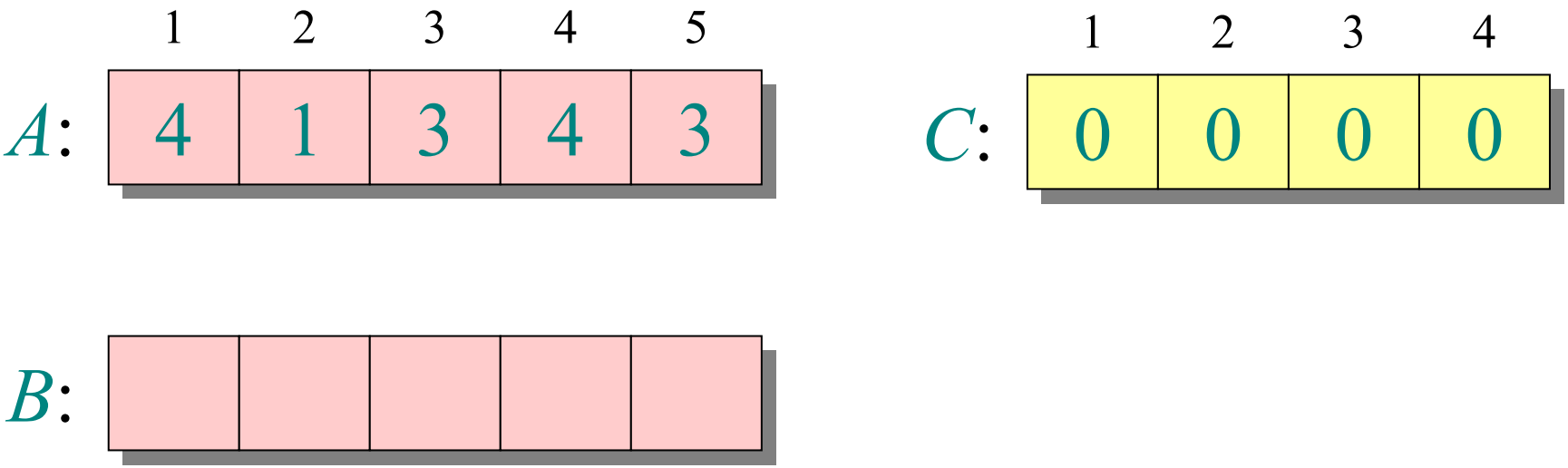
Sayma-sıralaması örneği



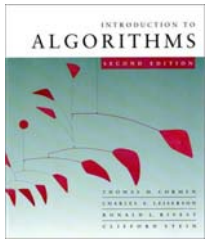




Döngü 1



for $i \leftarrow 1$ **to** k
do $C[i] \leftarrow 0$



Döngü 2

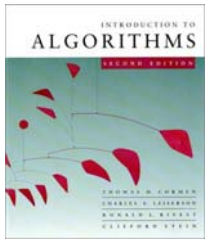
	1	2	3	4	5
A:	4	1	3	4	3

	1	2	3	4
C:	0	0	0	1

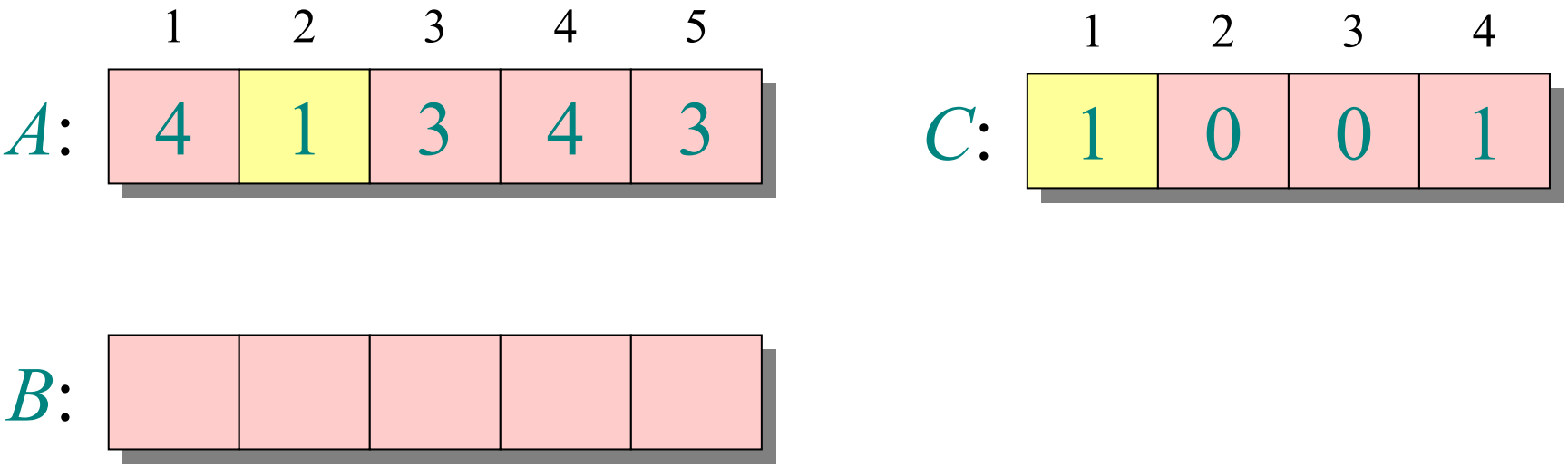
B:					

for $j \leftarrow 1$ **to** n

do $C[A[j]] \leftarrow C[A[j]] + 1$ ▷ $C[i] = |\{\text{key} = i\}|$

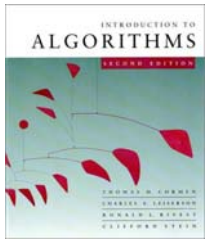


Döngü 2

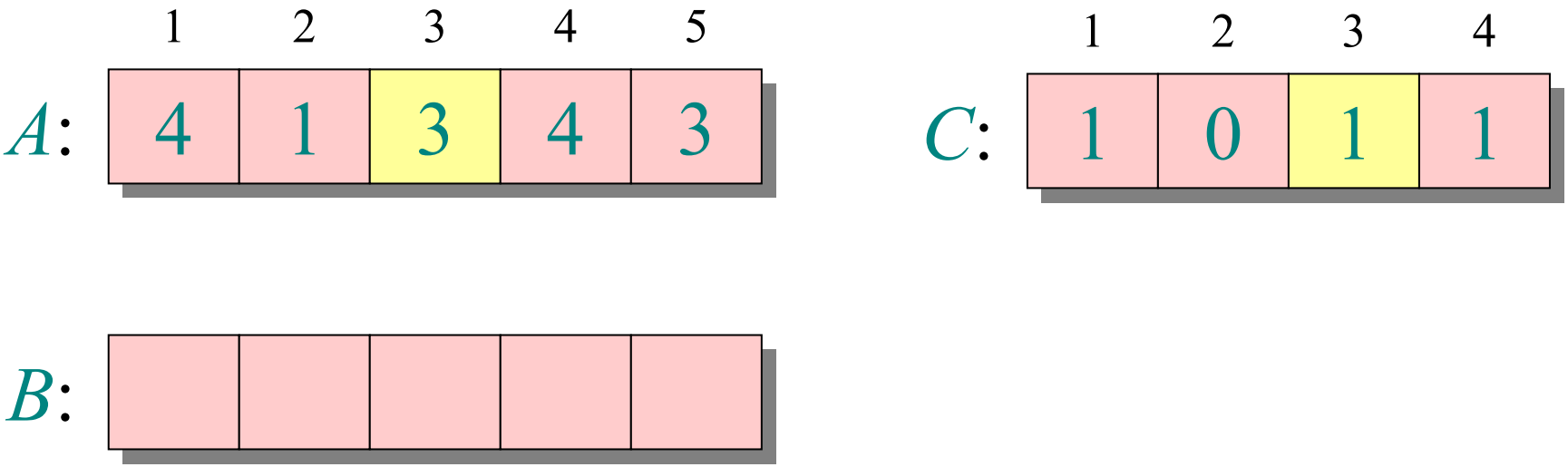


for $j \leftarrow 1$ **to** n

do $C[A[j]] \leftarrow C[A[j]] + 1$ ▷ $C[i] = |\{\text{key} = i\}|$

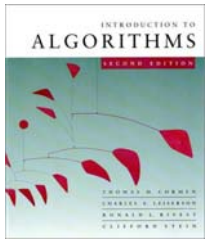


Döngü 2

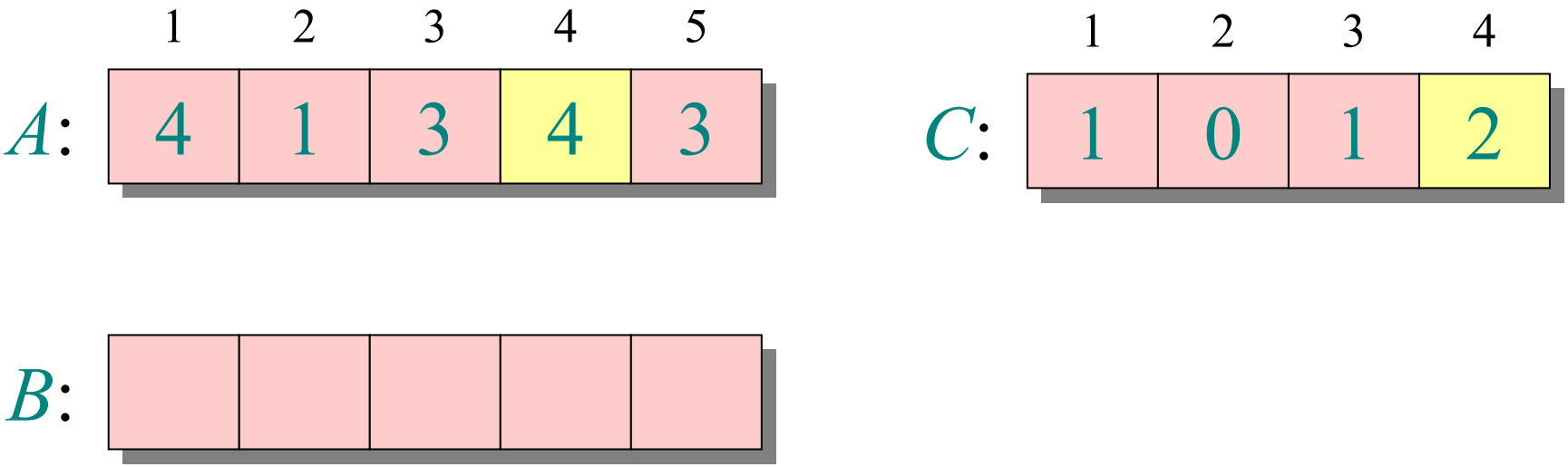


for $j \leftarrow 1$ **to** n

do $C[A[j]] \leftarrow C[A[j]] + 1$ ▷ $C[i] = |\{\text{key} = i\}|$

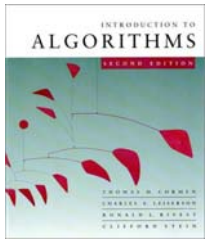


Döngü 2

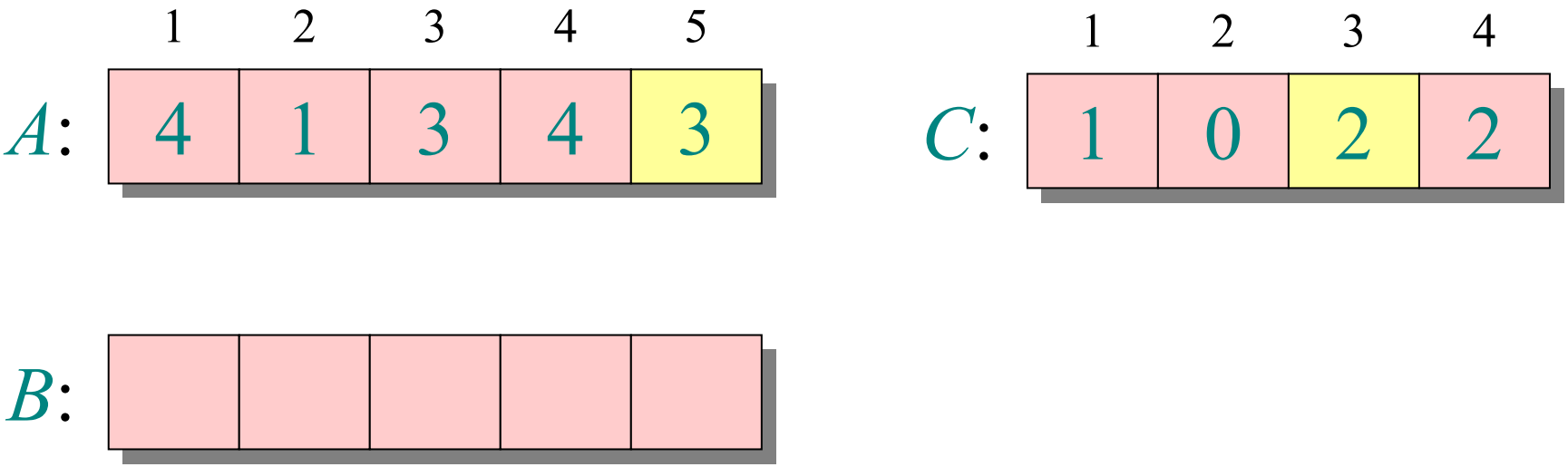


for $j \leftarrow 1$ **to** n

do $C[A[j]] \leftarrow C[A[j]] + 1$ ▷ $C[i] = |\{\text{key} = i\}|$

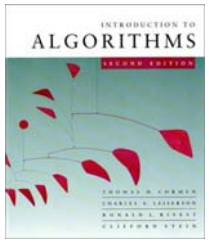


Döngü 2



for $j \leftarrow 1$ **to** n

do $C[A[j]] \leftarrow C[A[j]] + 1$ ▷ $C[i] = |\{\text{key} = i\}|$



Döngü 3

	1	2	3	4	5
A:	4	1	3	4	3

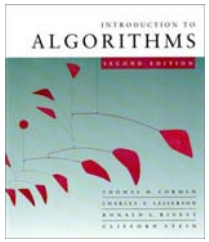
	1	2	3	4
C:	1	0	2	2

B:					

	1	2	3	4
C':	1	1	2	2

for $i \leftarrow 2$ **to** k

do $C[i] \leftarrow C[i] + C[i-1]$ ▷ $C[i] = |\{\text{key} \le i\}|$



Döngü 3

	1	2	3	4	5
A:	4	1	3	4	3

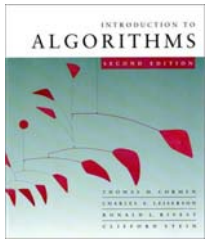
	1	2	3	4
C:	1	0	2	2

B:					

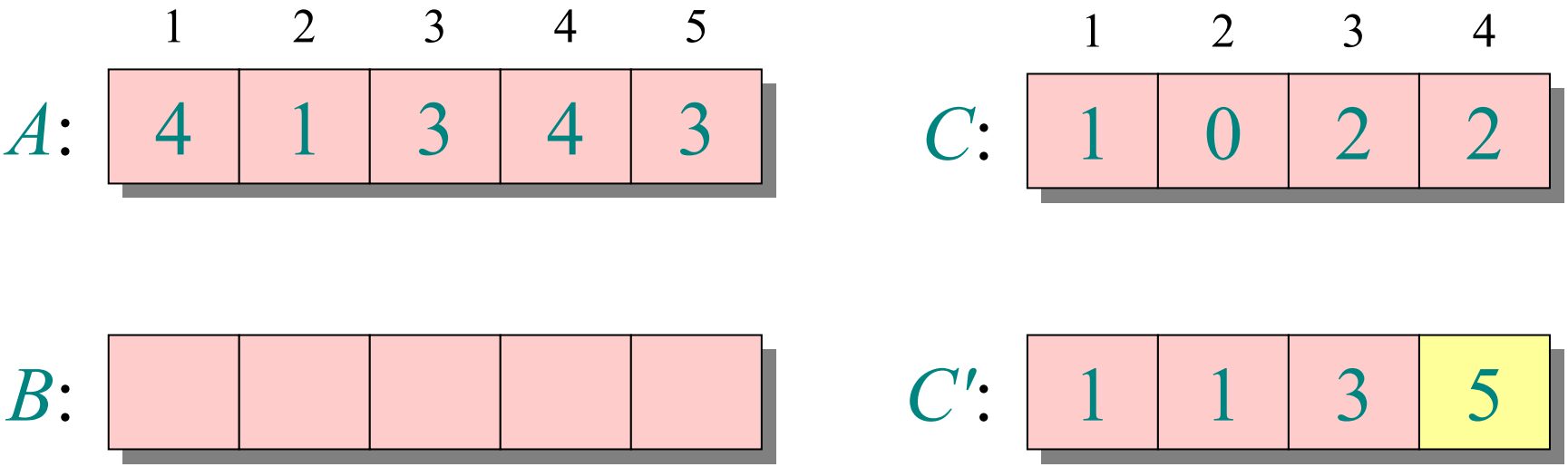
	1	2	3	4
C':	1	1	3	2

for $i \leftarrow 2$ **to** k

do $C[i] \leftarrow C[i] + C[i-1]$ ▷ $C[i] = |\{\text{key} \le i\}|$

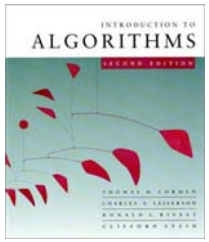


Döngü 3

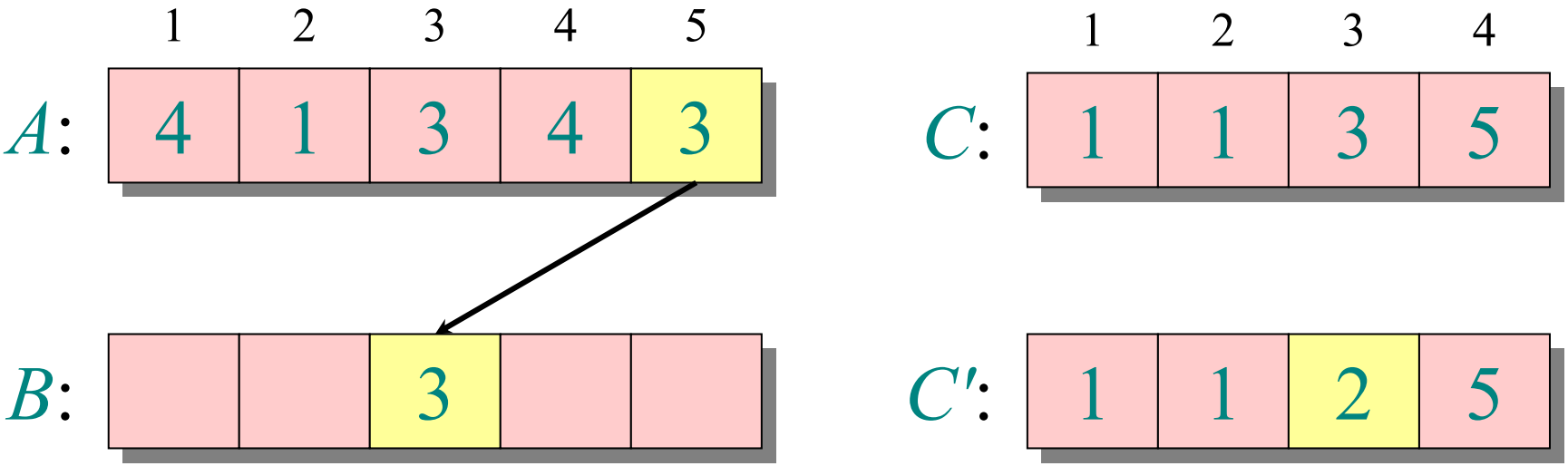


for $i \leftarrow 2$ **to** k

do $C[i] \leftarrow C[i] + C[i-1]$ ▷ $C[i] = |\{\text{key} \le i\}|$



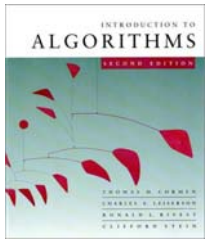
Döngü 4



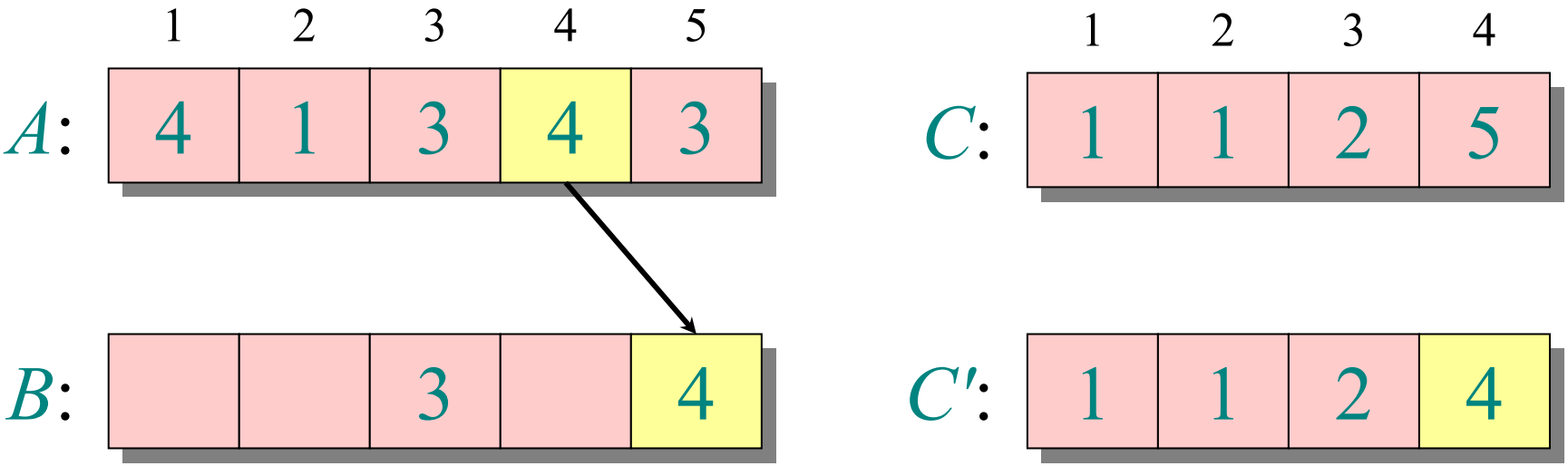
for $j \leftarrow n$ **down to** 1

do $B[C[A[j]]] \leftarrow A[j]$

$C[A[j]] \leftarrow C[A[j]] - 1$



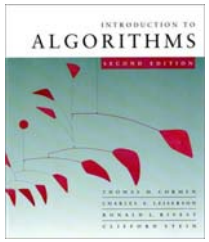
Döngü 4



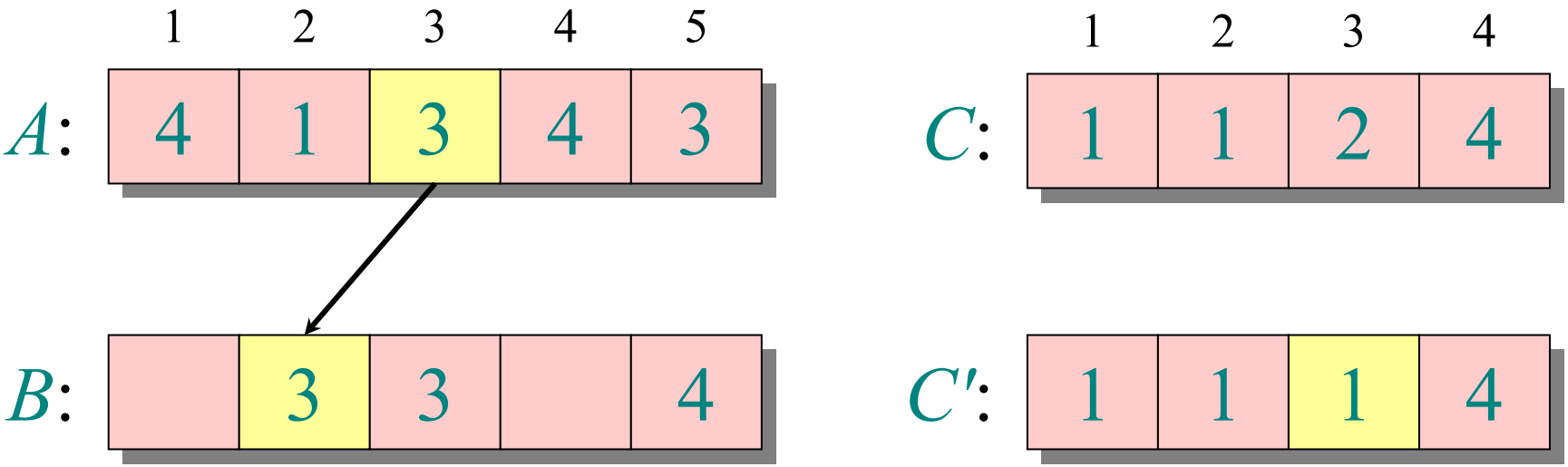
for $j \leftarrow n$ **down to** 1

do $B[C[A[j]]] \leftarrow A[j]$

$C[A[j]] \leftarrow C[A[j]] - 1$



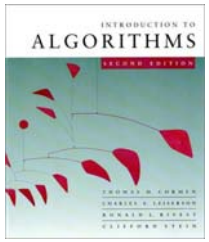
Döngü 4



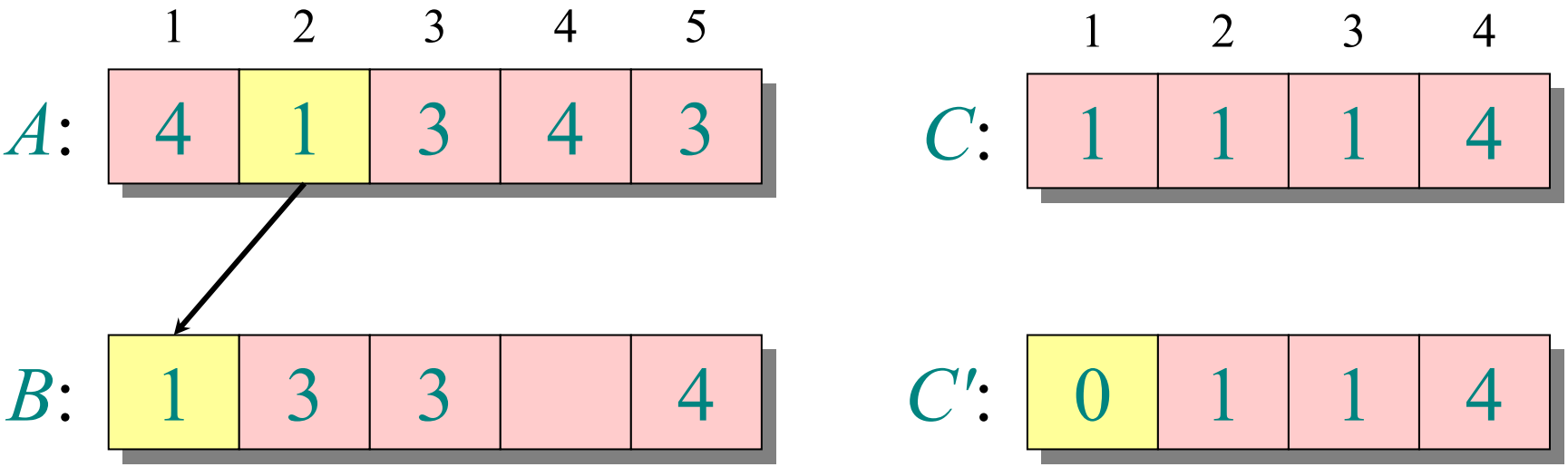
for $j \leftarrow n$ **down to** 1

do $B[C[A[j]]] \leftarrow A[j]$

$C[A[j]] \leftarrow C[A[j]] - 1$



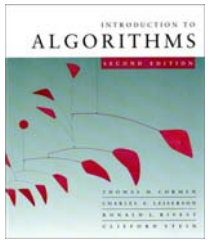
Döngü 4



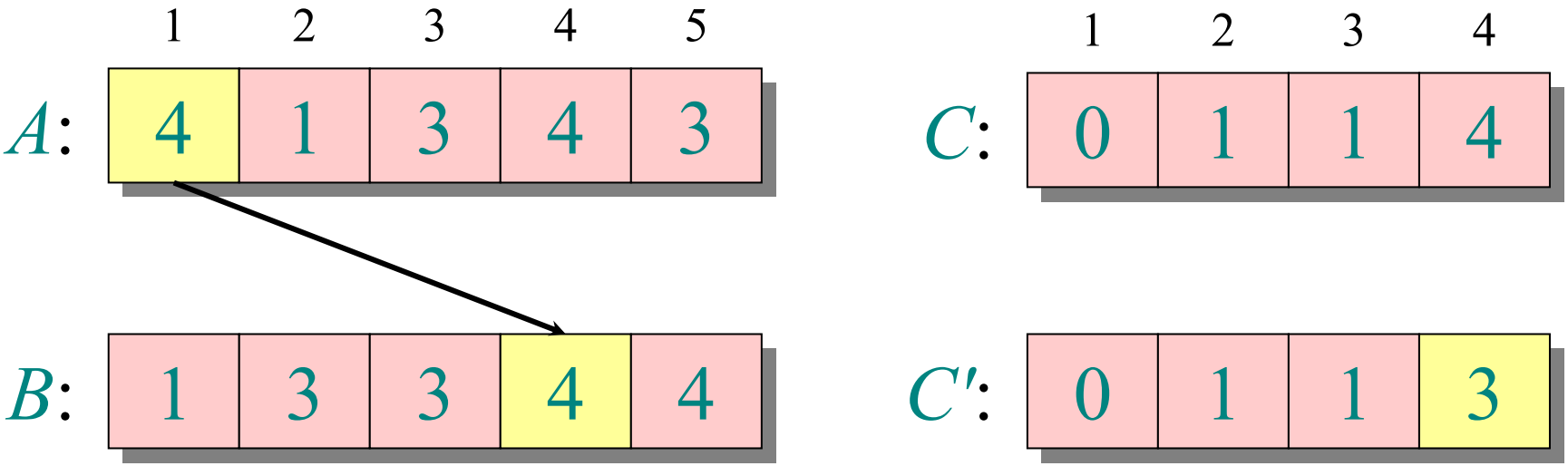
for $j \leftarrow n$ **down to** 1

do $B[C[A[j]]] \leftarrow A[j]$

$C[A[j]] \leftarrow C[A[j]] - 1$



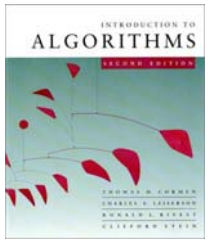
Döngü 4



for $j \leftarrow n$ **down to** 1

do $B[C[A[j]]] \leftarrow A[j]$

$C[A[j]] \leftarrow C[A[j]] - 1$



Çözümleme

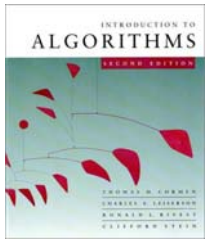
$\Theta(k)$ { **for** $i \leftarrow 1$ **to** k
  do $C[i] \leftarrow 0$

$\Theta(n)$ { **for** $j \leftarrow 1$ **to** n
  do $C[A[j]] \leftarrow C[A[j]] + 1$

$\Theta(k)$ { **for** $i \leftarrow 2$ **to** k
  do $C[i] \leftarrow C[i] + C[i–1]$

$\Theta(n)$ { **for** $j \leftarrow n$ **down to** 1
  do $B[C[A[j]]] \leftarrow \mathrm{A}[j]$
  $C[A[j]] \leftarrow C[A[j]] – 1$

$\Theta(n + k)$



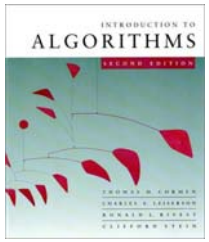
Koşma süresi

$k = O(n)$ ise, sayma sıralaması $\Theta(n)$ süresi alır.

- Ama sıralama $\Omega(n \lg n)$ süresi alıyor!
- Hata nerede?

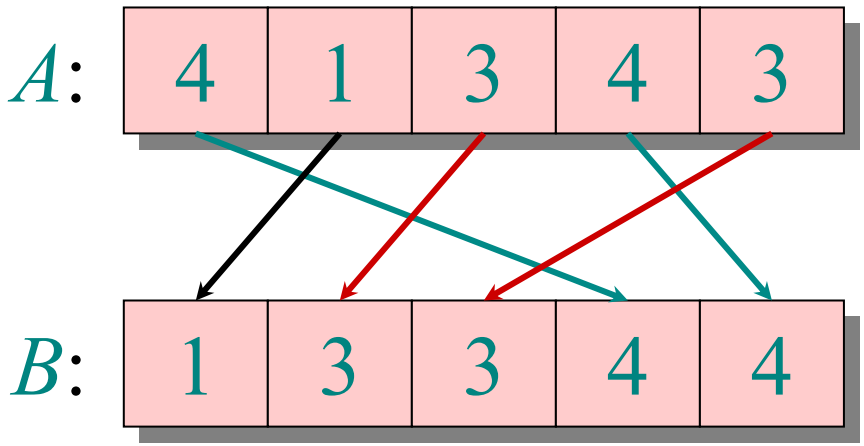
Yanıt:

- ***Karşılaştırma sıralaması*** $\Omega(n \lg n)$ süre alır.
- Sayma sıralaması bir ***karşılaştırma sıralaması*** değildir.
- Aslında elemanlar arasında bir tane bile karşılaştırma yapılmaz!

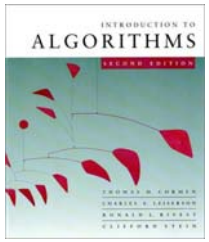


Kararlı sıralama

Sayma sıralaması ***kararlı*** bir sıralamadır: eşit eşit elemanlar arasındaki düzeni korur.

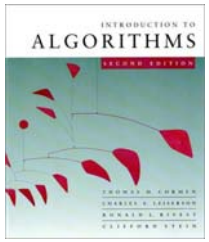


Alıştırma: Bu özelliği olan diğer sıralamalar hangileridir?

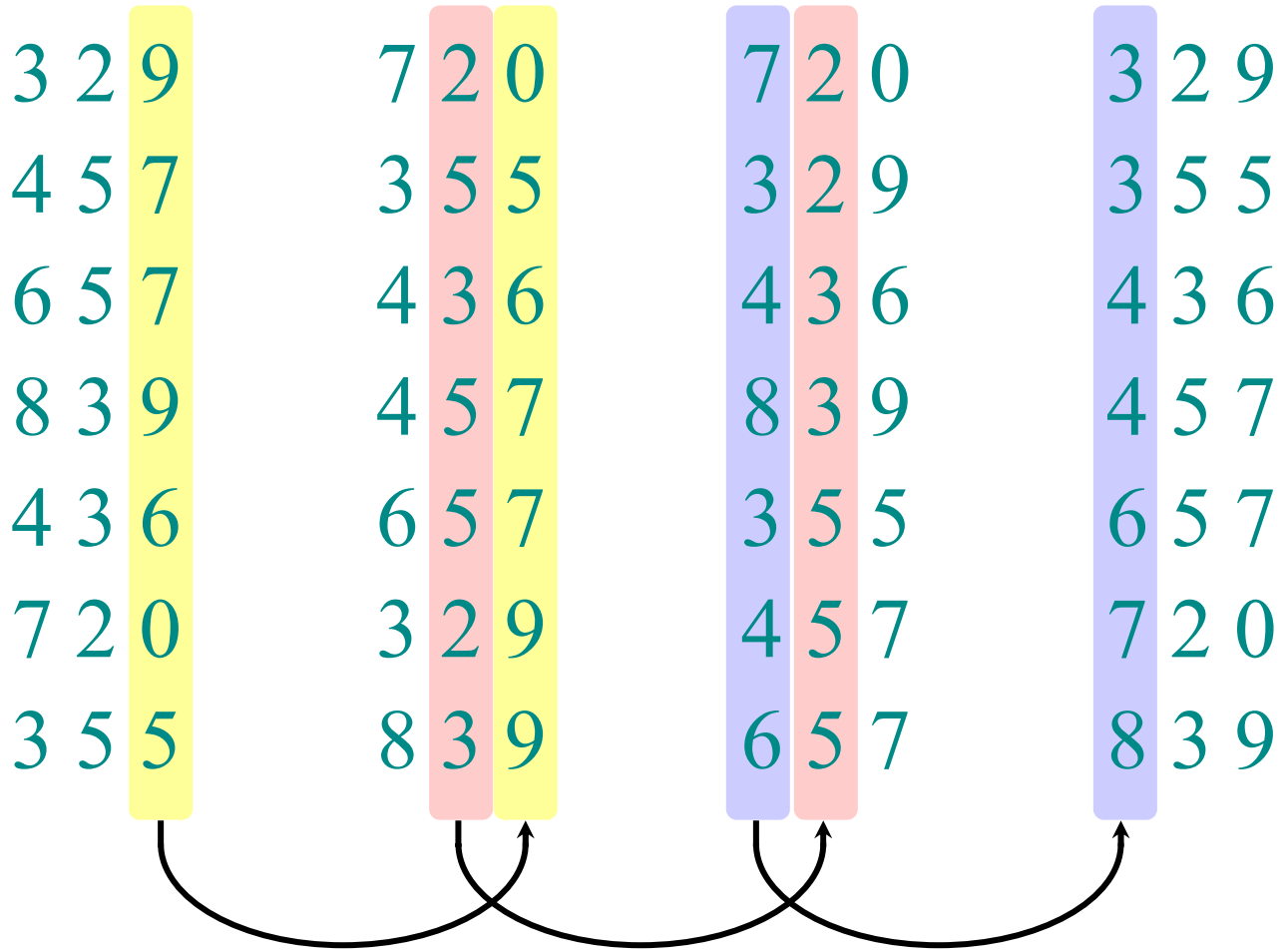


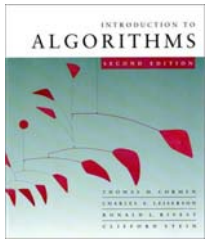
Taban (Radix) sıralaması

- ***Kaynak***: Herman Hollerith'in 1890 ABD seçimleri için icat ettiği makine. (Son eke bakınız. )
- Basamak basamak sıralama.
- Hollerith'in ilk (kötü) fikri: sıralamaya önceli en önemli basamaktan başlamak.
- İyi fikir: Sıralamaya ***en önemsiz basamaktan*** başlamak ve ek ***kararlı*** sıralama uygulamak.



Taban sıralaması uygulaması

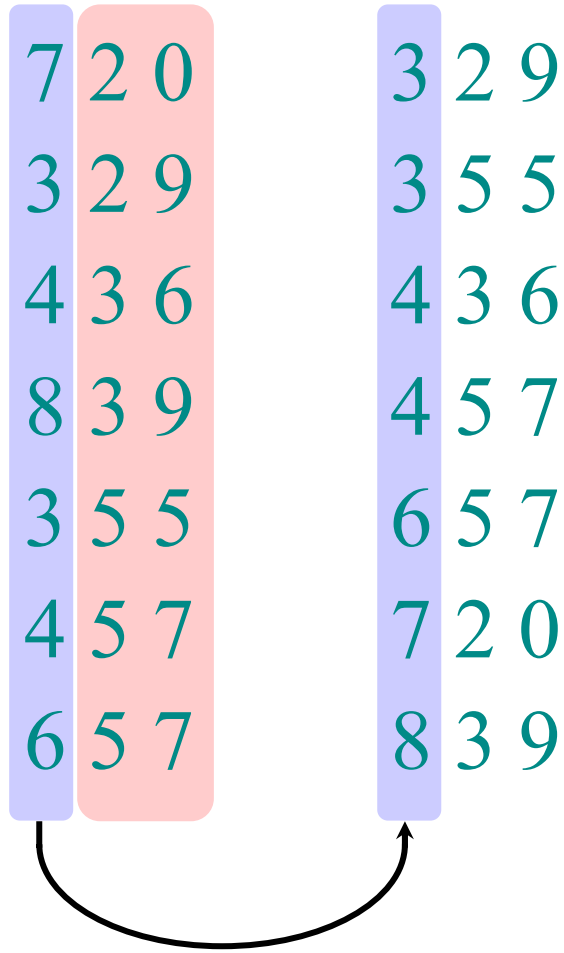


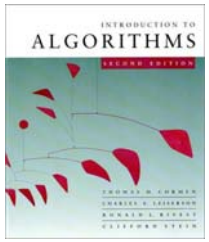


Taban sıralamasının doğruluğu

Basamak konumunda tümevarım

- Sayıların düşük düzeyli $t - 1$ basamaklarına göre sıralandığını varsayın.
- t basamağında sıralama yapın.

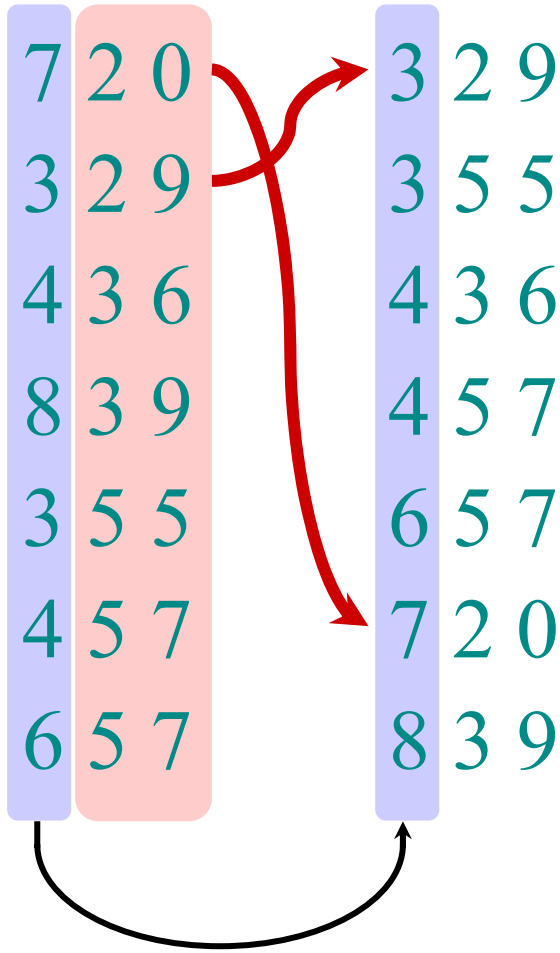


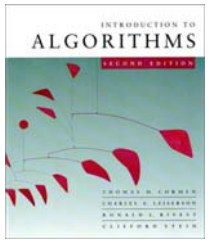


Taban sıralamasının doğruluğu

Basamak konumunda tümevarım

- Sayıların düşük düzeyli $t-1$ basamaklarına göre sıralandığını varsayın.
- t basamağında sıralama yapın.
  - t basamağında farklı olan iki sayı doğru sıralanmış.

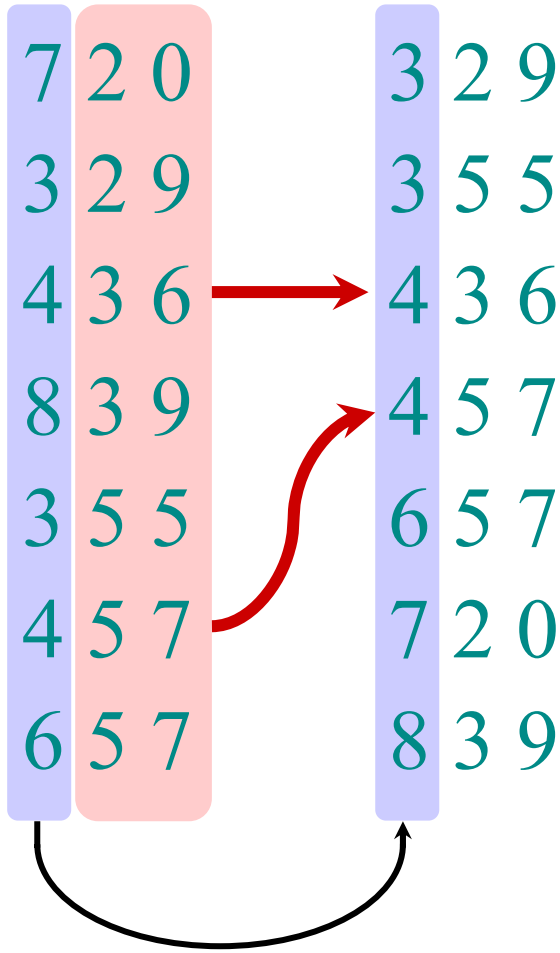


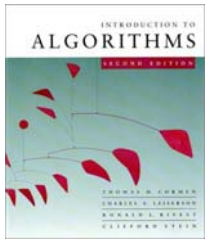


Taban sıralamasının doğruluğu

Basamak konumunda tümevarım

- Sayıların düşük düzeyli $t-1$ basamaklarına göre sıralandığını varsayın.
- t basamağında sıralama yapın.
  - t basamağında farklı olan iki sayı doğru sıralanmış.
  - t basamağındaki iki eşit sayının girişteki sıraları muhafaza edilmiş $\Rightarrow$ doğru sıra.

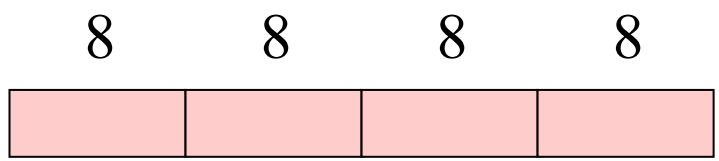




Taban sıralamasının çözümlemesi

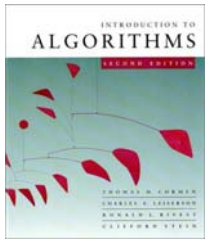
- Sayma sıralamasını ek kararlı sıralama varsayın.
- Herbiri b bit olan n bilgiişlem sözcüğünü sıralayın.
- Her sözcüğün basamak yapısı b/r taban-2^r olarak görülebilir.

Örnek: 32-bit sözcük



$r = 8 \Rightarrow b/r = 4$ ise, taban-2^8 basamak durumunda sıralama 4 geçiş yapar; veya $r = 16 \Rightarrow b/r = 2$ ise, taban-2^{16} basamakta 2 geçiş yapar.

Kaç geçiş yapmalıyız?



Çözümleme (devam)

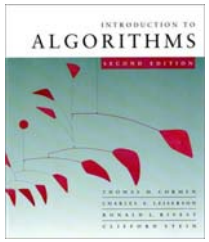
Hatırla: Sayma sıralaması $\Theta(n + k)$ süresini alır; (0 ile $k - 1$ aralığında n sayıyı sıralamak için).

Her b-bitlik sözcük r-bitlik parçalara ayrılırsa, sayma sıralamasının her geçişi $\Theta(n + 2^r)$ süre alır. Bu durumda b/r geçiş olduğundan, elimizde:

$$T(n,b) = \Theta\left(\frac{b}{r}\left(n + 2^r\right)\right)$$ olur.

r' yi, $T(n,b)$' yi en aza düşürecek gibi seçin:

- r'yi arttırmak daha az geçiş demektir, ama $r >> \lg n$ olduğundan, süre üstel olarak artar.



r' yi seçmek

$$T(n,b) = \Theta\left(\frac{b}{r}\left(n + 2^r\right)\right)$$

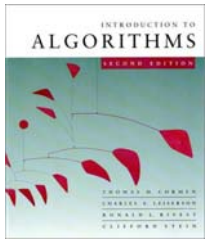
$T(n,b)$'yi türevini alıp 0' a eşitleyerek en aza düşürün..

Veyahut da , istemediğimiz değer $2^r >> n$ olduğundan, bu sınırlamaya bağlı kalarak r'yi olabildiğince büyük seçmenin asimptotik bir sakıncası olmadığını gözleyin.

$r = \lg n$ seçimi $T(n,b) = \Theta(bn/\lg n)$ anlamına gelir.

- 0 ile $n^d - 1$ aralığındaki sayılarla $b = d \lg n$ 'yi

elde ederiz. $\Rightarrow$ taban sıralaması $\Theta(dn)$ süresini alır.



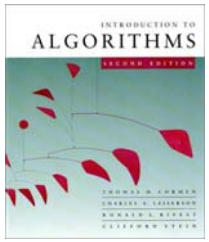
Sonuçlar

Pratikte taban sıralaması büyük girişler için hızlıdır; aynı zamanda kod yazması ve bakımı kolaydır.

Örnek (32-bitlik sayılar için):

- En çok 3 geçiş (≥ 2000 sayının sıralanmasında).
- Birleştirme sıralaması /çabuk sıralama $\lceil \lg 2000 \rceil$ en az 11 geçiş yaparlar.

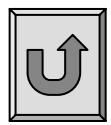
Dezavantajı: Çabuk sıralamanın aksine, taban sıralamasının yer referansları zayıftır ve bu nedenle ince ayarlı bir çabuk sıralama, dik bellek sıradüzeni olan günümüz işlemcilerinde daha iyi çalışır.

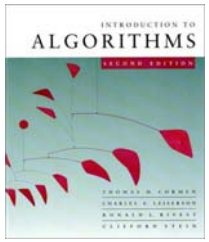


Ek Not: Delikli kart teknolojisi

- Herman Hollerith (1860-1929)
- Delikli kartlar
- Hollerith'in tablolama sistemi
- Sıralayıcının çalışması
- Taban sıralamasının kaynağı
- "Modern" IBM kartı
- Delikli kart teknolojisi ile ilgili Web kaynakları

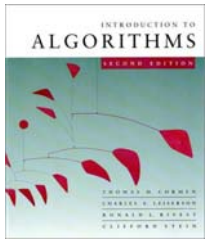
Son izlenen slide'a dön





Herman Hollerith (1860-1929)

- 1880 ABD Nüfus Sayımının değerlendirmesi yaklaşık 10 yılda yapıldı.
- MIT' de ders verdiği dönemde, Hollerith delikli-kart teknolojisinin prototipini geliştirdi.
- Aralarında bir “kart sıralayıcısı” nın da olduğu makineleri, 1890 sayımını 6 haftada raporladı.
- 1911' de "Tablolama Makineleri Şirketi" ni kurdu. Bu şirket 1924'de başkalarıyla birleşerek IBM'i (International Business Machines) kurdu.

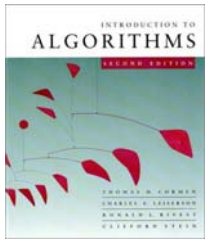


Delikli kartlar

- Delikli kart = veri kaydı.
- Delik = değer.
- Algoritma = makine + insan.

Hollerith'in tablolama sistemi ve delikli kartını internette Genbilim makalesinde bulabilirsiniz.

*R*esim telif nedeniyle kaldırılmıştır.
1900 ABD sayımında kullanılan kartın örneği. [Howells 2000]

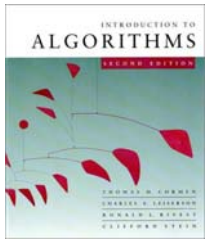


Hollerith'in tablolama sistemi

- Pantograph kart delicisi
- El baskısı okuyucusu
- Kadran sayaçları
- Sıralama kutusu

Resim telif kısıtlaması nedeniyle kaldırılmıştır.

"Hollerith'in Tablolayıcı ve Sıralayıcısı: Mekanik sayıcı ve tablolama baskıcısının detayları." Resim: [Howells 2000].

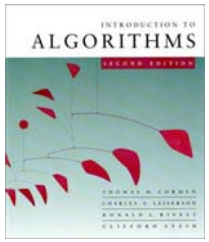


Sıralayıcının çalışması

- İşletmen baskıcıya bir kart sokar.
- Baskıcıdaki iğneler delinmiş boşluklardan geçerek kartın altındaki civa dolu kaplarla elektrik kontağı kurarlar.
- Belirli bir basamağın değeri delinmişse, ilgili sıralama kutusunun kapağı açılır.
- İşletmen kartı depolama kutusuna koyar ve kapağı kapatır.
- Tüm kartların işlenmesi bittiğinde ön pano açılır ve kartlar sırasıyla toplanır; böylece kararlı sıralamanın bir geçişi tamamlanır.

Resim telif kısıtlaması nedeniyle kaldırılmıştır.

Hollerith'in Tablolama, Pantograf, Baskıcı ve Sıralayıcısı (http://www.columbia.edu/acis/history/census-tabulator.html)

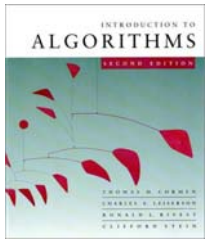


Taban sıralamasının kaynağı

Hollerith'in orijinal 1889 patenti taban sıralamasında en-önemli-basamak-en-önce mantığını ima ediyor:

> *"The most complicated combinations can readily be counted with comparatively few counters or relays by first assorting the cards according to the first items entering into the combinations, then reassorting each group according to the second item entering into the combination, and so on, and finally counting on a few counters the last item of the combination for each group of cards."*

En-önemsiz-basamak-en-önce mantığı olan taban sıralaması makine işletmenlerinin anonim buluşu gibi...



“Modern” IBM kartı

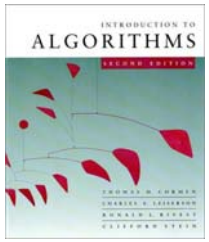
- Her sütuna bir karakter.

Resim telif kısıtlaması nedeniyle kaldırılmıştır..

Resmi görmek için:
http://www.museumwaalsdorp.nl/computer/images/ibmcard.jpg

Üretici:
WWW Virtual Punch-Card Server.

Bu nedenle yazı pencerelerinde 80 sütun vardır!



Delikli-kart teknolojisi ile ilgili Web kaynakları

- Doug Jones's punched card index
- Biography of Herman Hollerith
- The 1890 U.S. Census
- Early history of IBM
- Pictures of Hollerith's inventions
- Hollerith's patent application (Gordon Bell's CyberMuseum'un katkısıyla)
- Impact of punched cards on U.S. history

