Algorithmen und Datenstrukturen

Prof. Dr. Ralf Möller

Universität zu Lübeck

Institut für Informationssysteme

Felix Kuhr (Übungen)

sowie viele Tutoren

Sortierung in linearer Zeit

- Sortieren: Geht es doch noch schneller als in $\Omega(n \log n)$ Schritten?
- Man muss „schärfere" Annahmen über das Problem machen können ...
  - z.B. Schlüssel in n Feldelementen aus dem Bereich [1..n]
- ... oder Nebenbedingungen „abschwächen"
  - z.B. die In-situ-Einschränkung aufgeben
- Zentrale Idee: Vermeide Vergleiche!

Seward, H. H. (**1954**), "2.4.6 Internal Sorting by Floating Digital Sort", Information sorting in the application of electronic digital computers to business operations, Master's thesis, Report R-232, Massachusetts Institute of Technology, Digital Computer Laboratory, pp. 25–28

A. Andersson, T. Hagerup, S. Nilsson, R. Raman, Sorting in Linear Time?, J. Comput. Syst. Sci. 57(1): 74-93, **1998**

Sortieren durch Zählen / Counting-Sort

- **Wissen:**
  Schlüssel fallen in einen kleinen Zahlenbereich
- **Beispiel 1:** Sortiere eine Menge von Studierenden nach Examensbewertungen (Scores sind Zahlen)
  - 1000 Studenten
  - Maximum score: 100
  - Minimum score: 0
- **Beispiel 2:** Sortiere Studierende nach dem ersten Buchstaben des Nachnamens
  - Anzahl der Studierenden: viele
  - Anzahl der Buchstaben: 26

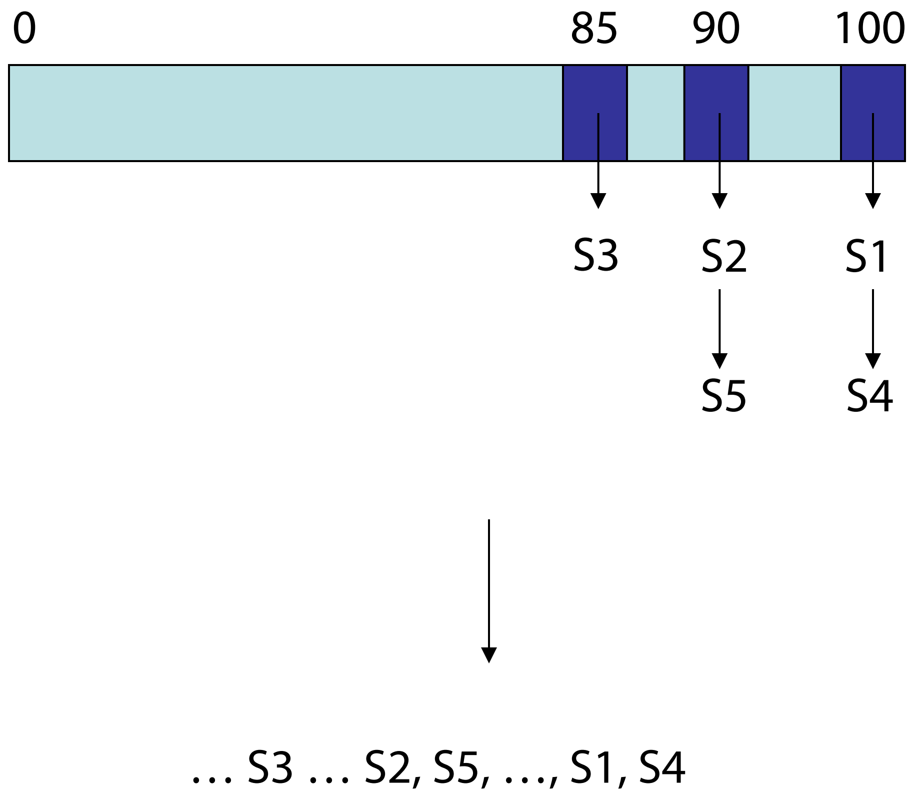
Counting-Sort

- ***Eingabe***: $A[1\,..\,n]$, wobei $A[j] \in \{1, 2, \ldots, k\}$.
- ***Ausgabe***: $B[1\,..\,n]$, sortiert.
- ***Hilfsspeicher***: $C[1\,..\,k]$.

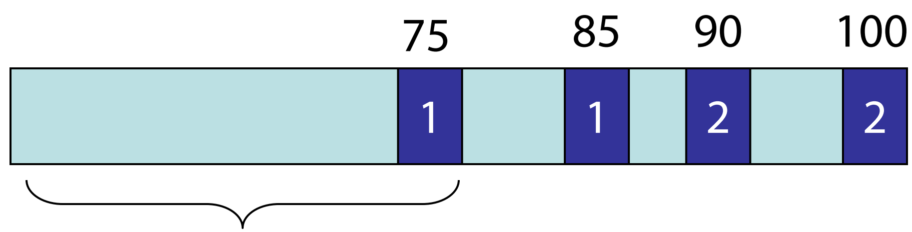
- Kein In-situ-Sortieralgorithmus
- Benötigt Θ(n+k) zusätzliche Speicherplätze

Intuition

- S1: 100
- S2: 90
- S3: 85
- S4: 100
- S5: 90
- …



Intuition



50 Studierende mit Score ≤ 75
Was ist der Rang (von klein auf groß) für einen Studenten mit Score 75?

50

200 Studierende mit Score ≤ 90
Was ist der Rang für einen Studenten mit Score 90?

200 or 199

Counting-Sort

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

Initialisiere

2. **for** $j \leftarrow 1$ **to** n
   do $C[A[j]] \leftarrow C[A[j]] + 1$ $\triangleright$ $C[i] = |\{\text{key} = i\}|$

Zähle

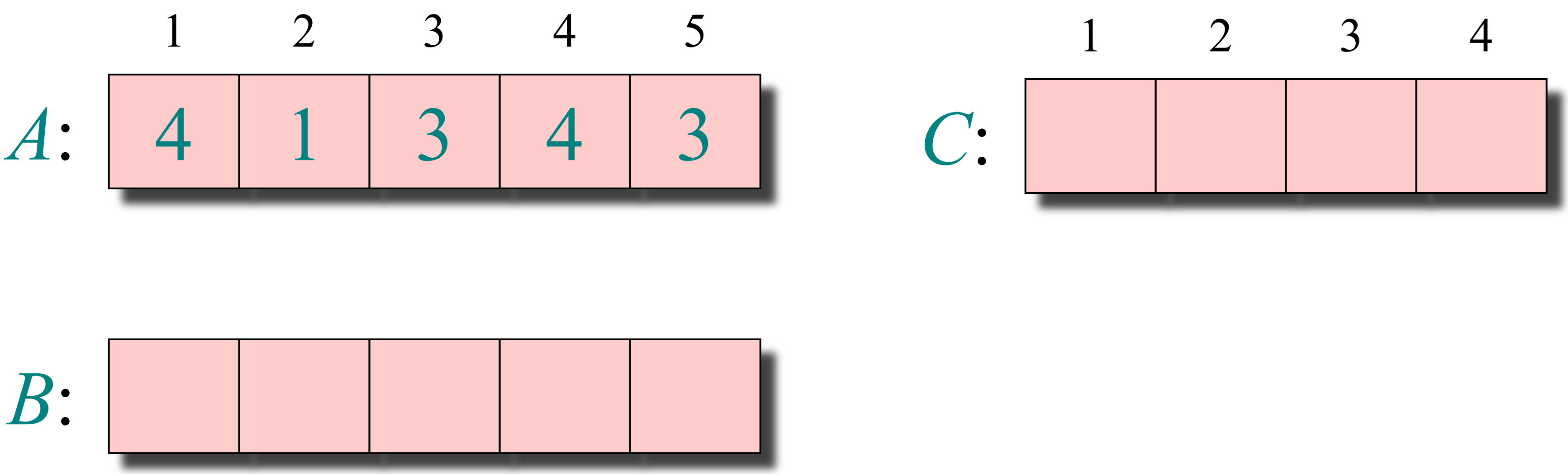
3. **for** $i \leftarrow 2$ **to** k
   do $C[i] \leftarrow C[i] + C[i-1]$ $\triangleright$ $C[i] = |\{\text{key} \leq i\}|$

Bestimme Summe

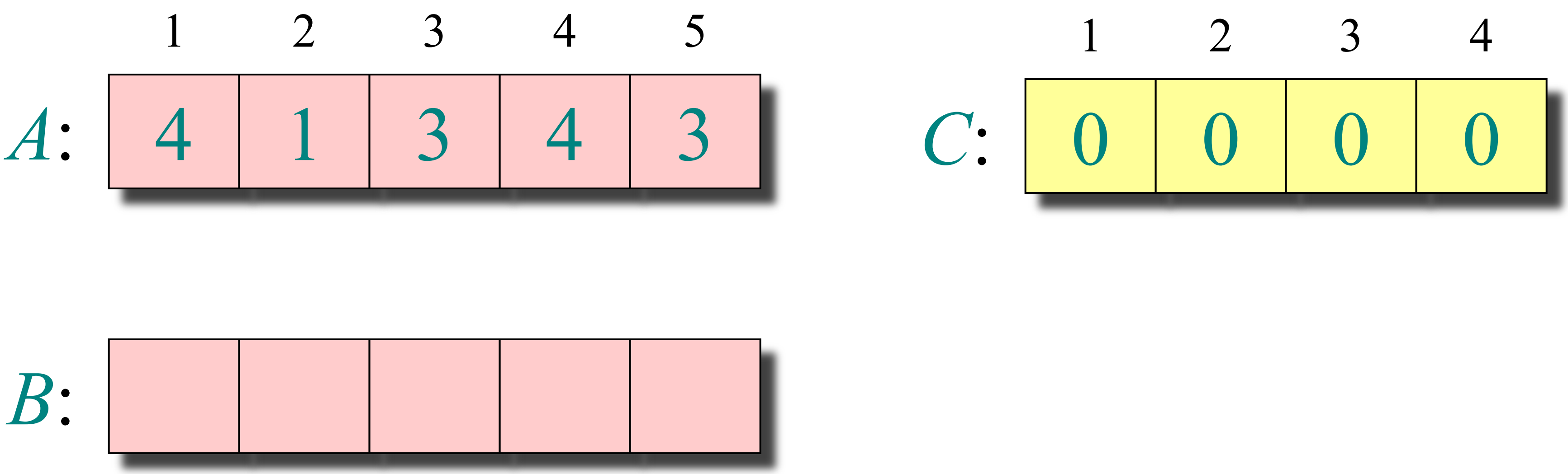
4. **for** $j \leftarrow n$ **downto** 1
   do $B[C[A[j]]] \leftarrow A[j]$
   $C[A[j]] \leftarrow C[A[j]] - 1$

Ordne neu

Counting-Sort Beispiel



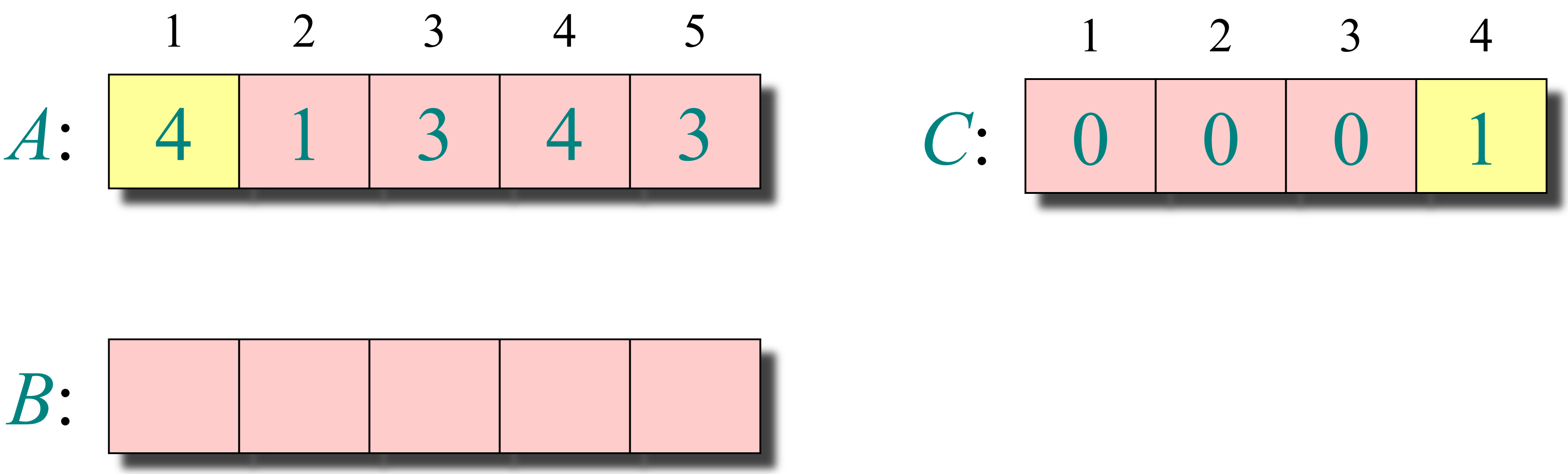
Schleife 1: Initialisierung



1. **for** $i \leftarrow 1$ **to** k

do $C[i] \leftarrow 0$

Schleife 2: Zähle



$$\textbf{2. for } j \leftarrow 1 \textbf{ to } n$$
$$\textbf{do } C[A[j]] \leftarrow C[A[j]] + 1 \quad \triangleright\ C[i] = |\{\text{key} = i\}|$$

Schleife 2: Zähle

A:

1	2	3	4	5
4	1	3	4	3

C:

1	2	3	4
1	0	0	1

B:

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

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

Schleife 2: Zähle

A:

1	2	3	4	5
4	1	3	4	3

C:

1	2	3	4
1	0	1	1

B:

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

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

Schleife 2: Zähle

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

	1	2	3	4
C:	1	0	1	2

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

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

Schleife 2: Zähle

A:

1	2	3	4	5
4	1	3	4	3

C:

1	2	3	4
1	0	2	2

B:

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

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

Schleife 3: Berechne Summme

A:

1	2	3	4	5
4	1	3	4	3

C:

1	2	3	4
1	0	2	2

B:

C':

1	1	2	2

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

do $C[i] \leftarrow C[i] + C[i\text{–}1]$ $\quad \triangleright\ C[i] = |\{\text{key} \leq i\}|$

Schleife 3: Berechne Summe

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

	1	2	3	4
C:	1	0	2	2
C':	1	1	3	2

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

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

Schleife 3: Berechne Summe

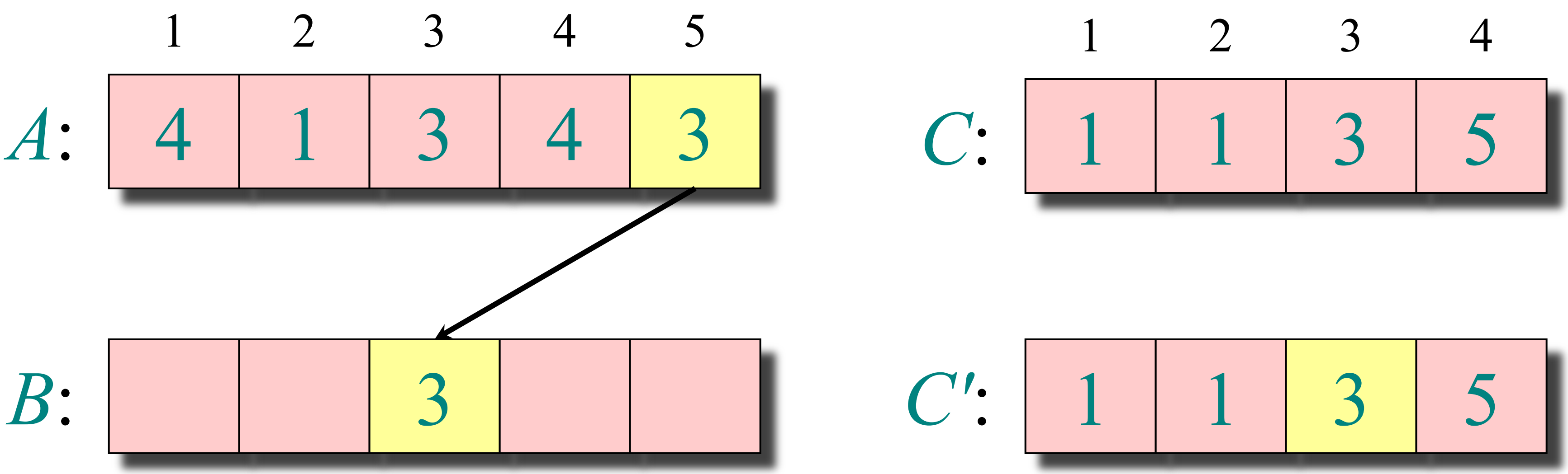
	1	2	3	4	5
A:	4	1	3	4	3
B:					

	1	2	3	4
C:	1	0	2	2
C':	1	1	3	5

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

do $C[i] \leftarrow C[i] + C[i–1]$ $\quad \triangleright\ C[i] = |\{\text{key} \leq i\}|$

Schleife 4: Ordne neu

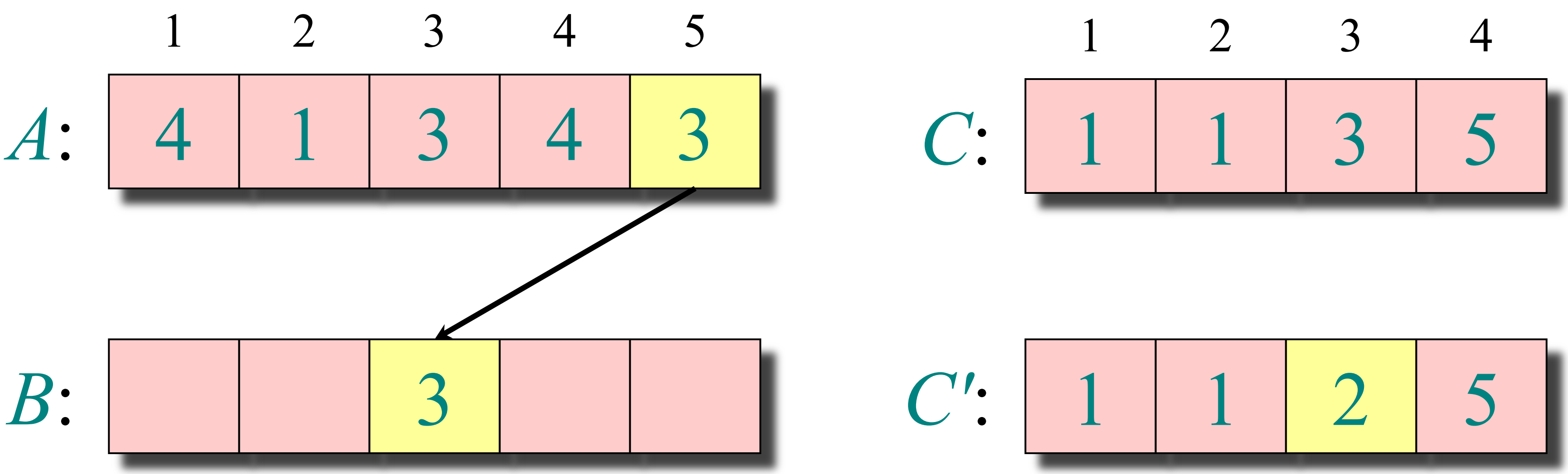


4. for $j \leftarrow n$ **downto** 1

do $B[C[A[j]]] \leftarrow \mathrm{A}[j]$

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

Schleife 4: Ordne neu

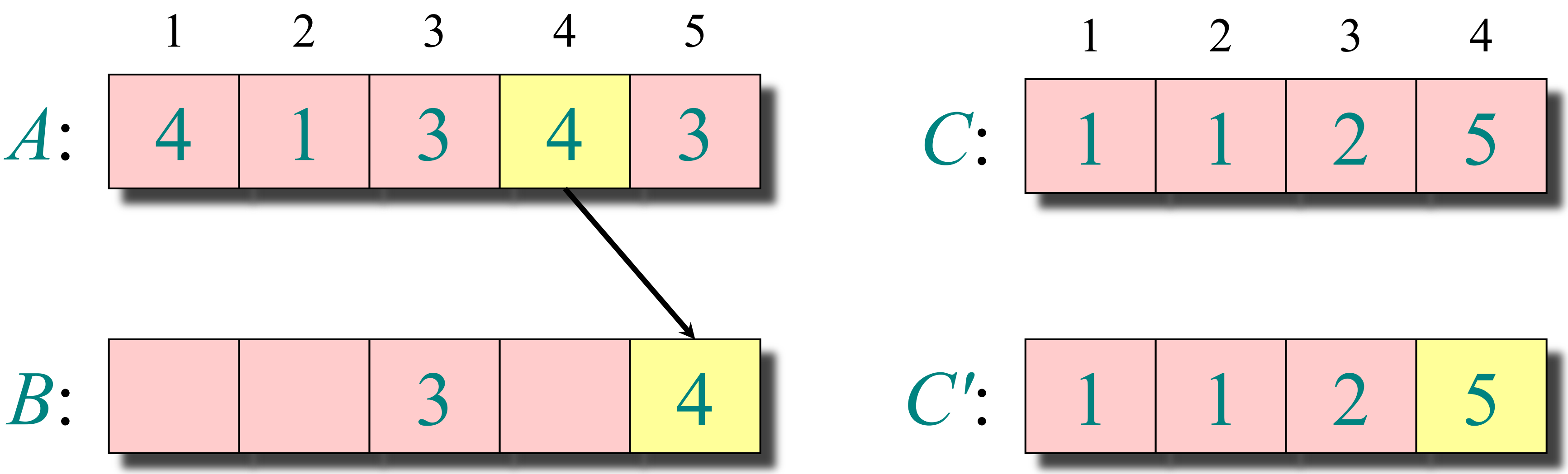


4. for $j \leftarrow n$ **downto** 1

do $B[C[A[j]]] \leftarrow \mathrm{A}[j]$

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

Schleife 4: Ordne neu

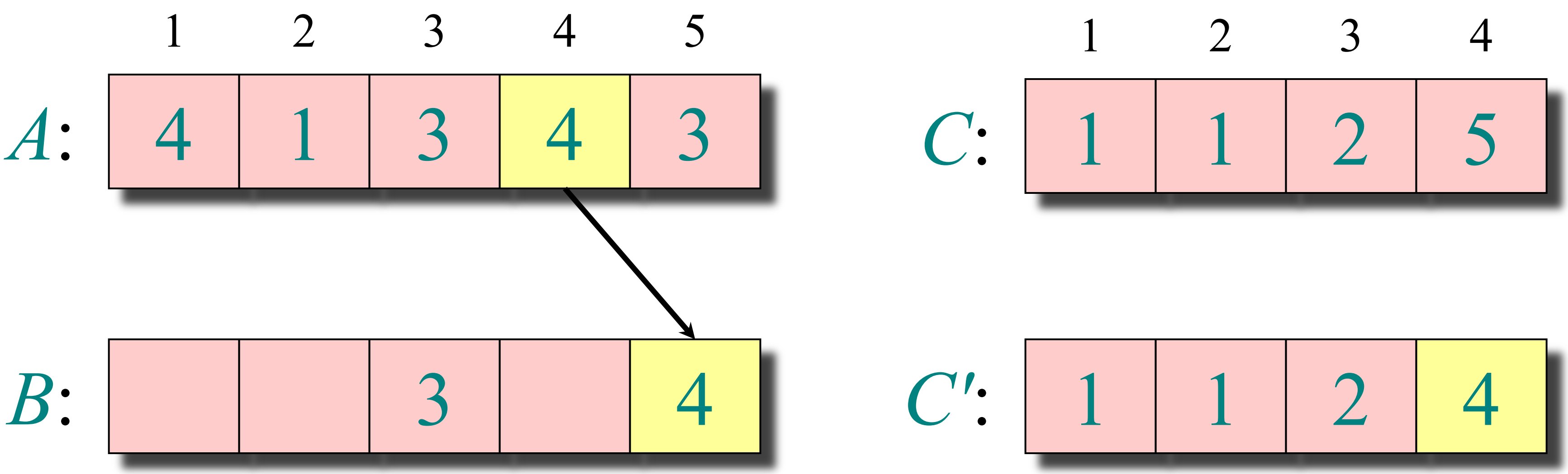


4. for $j \leftarrow n$ **downto** 1

do $B[C[A[j]]] \leftarrow \mathrm{A}[j]$

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

Schleife 4: Ordne neu

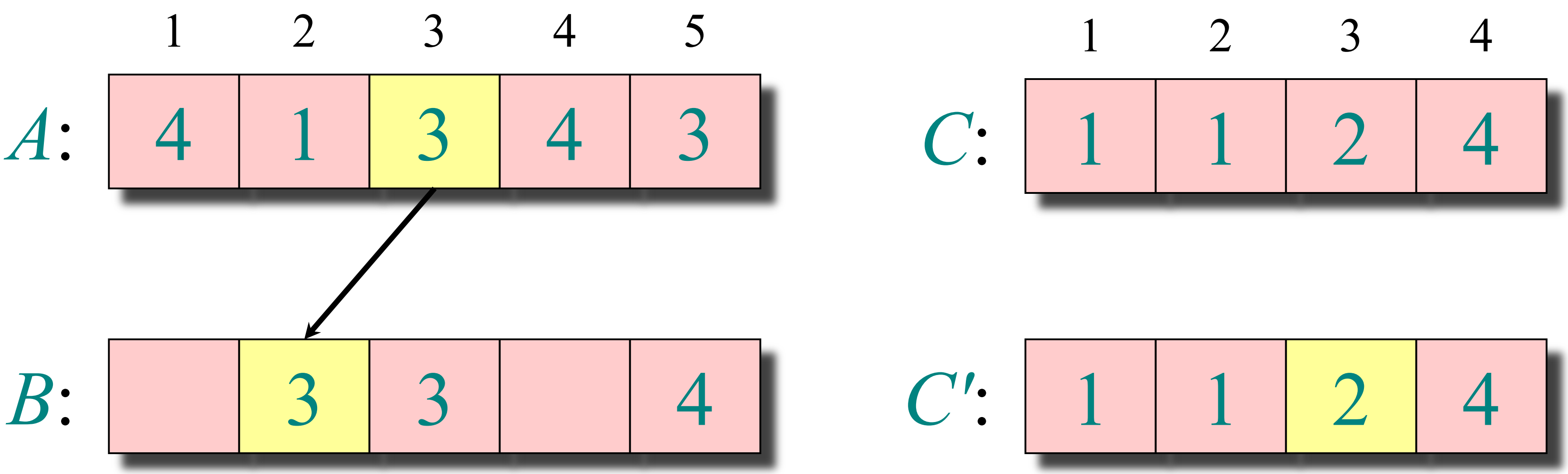


4. for $j \leftarrow n$ **downto** 1

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

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

Schleife 4: Ordne neu

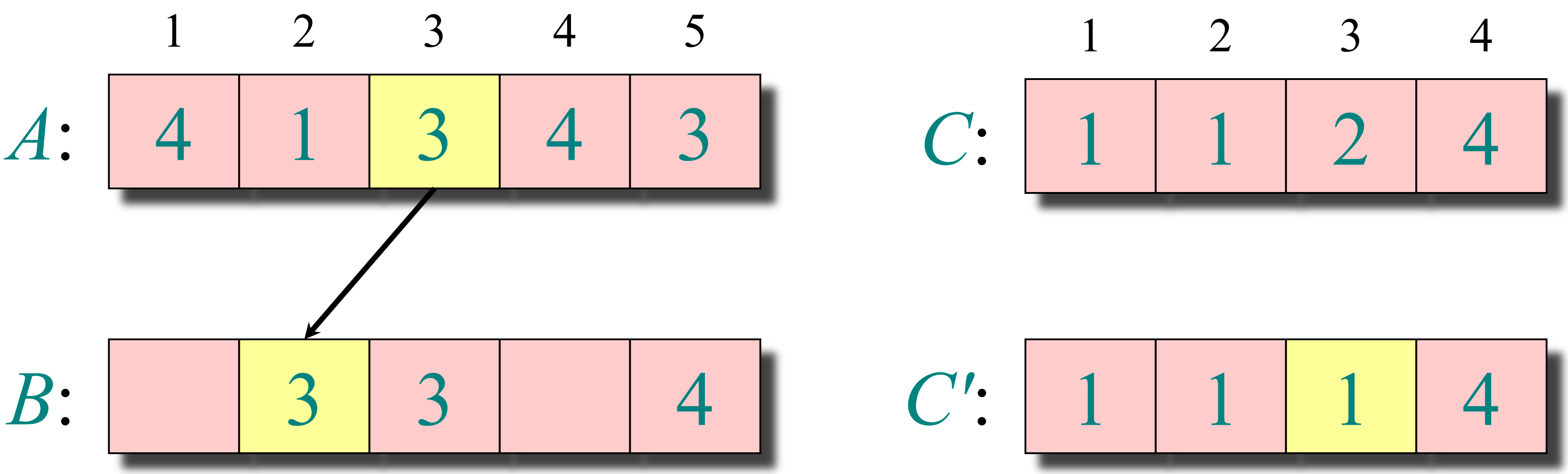


4. for $j \leftarrow n$ **downto** 1

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

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

Schleife 4: Ordne neu

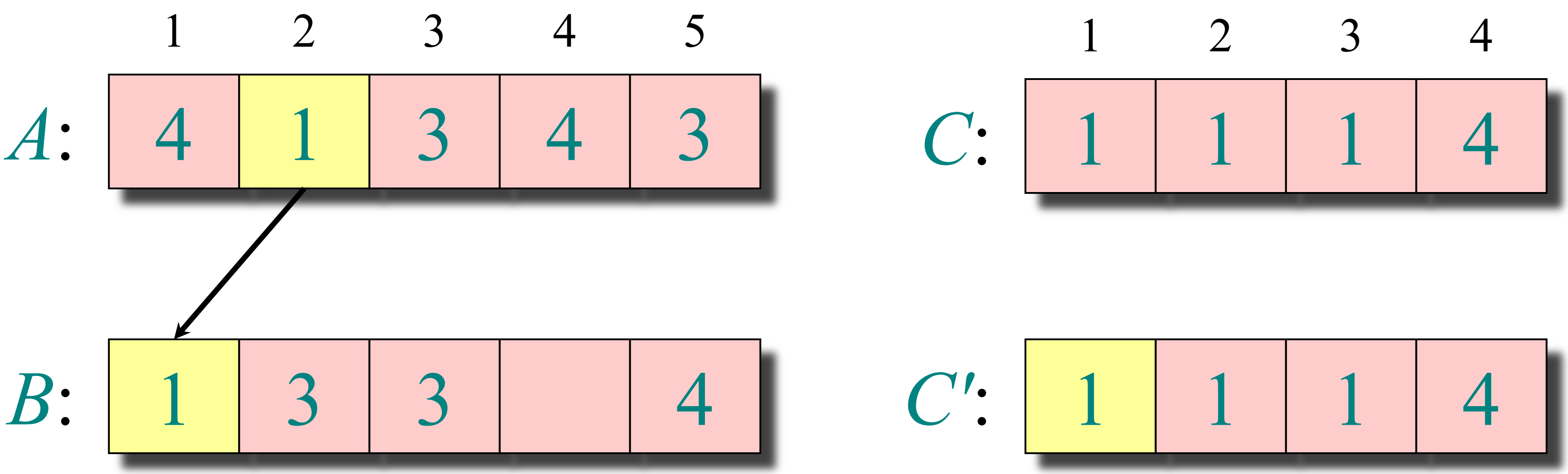


4. for $j \leftarrow n$ **downto** 1

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

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

Schleife 4: Ordne neu

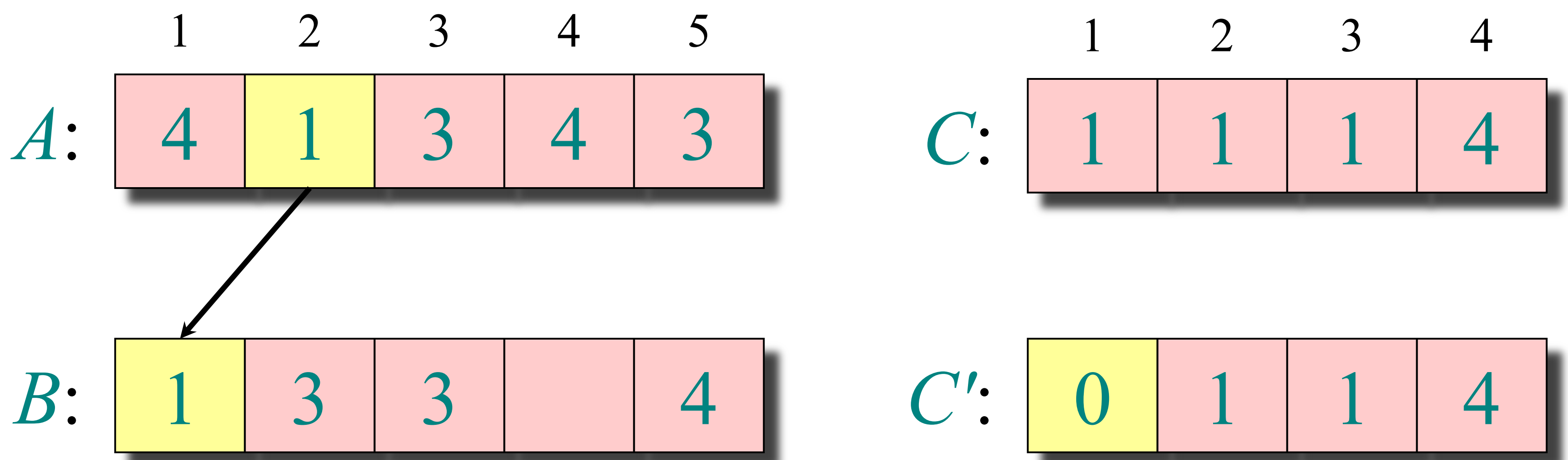


4. for $j \leftarrow n$ **downto** 1

do $B[C[A[j]]] \leftarrow \mathrm{A}[j]$

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

Schleife 4: Ordne neu

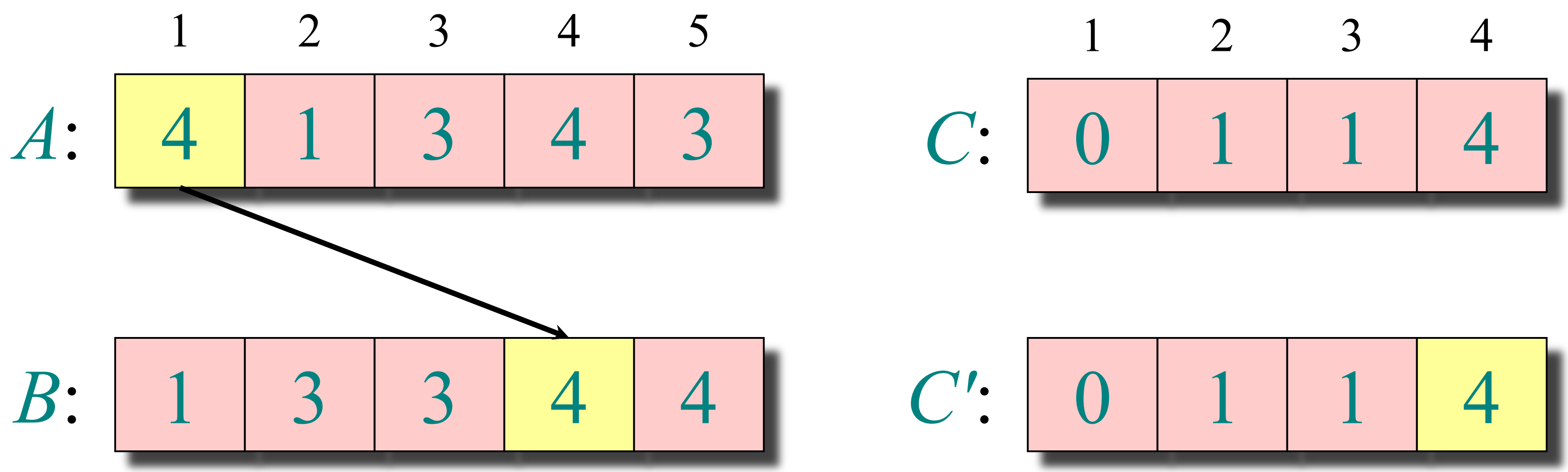


4. for $j \leftarrow n$ **downto** 1

do $B[C[A[j]]] \leftarrow \mathrm{A}[j]$

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

Schleife 4: Ordne neu

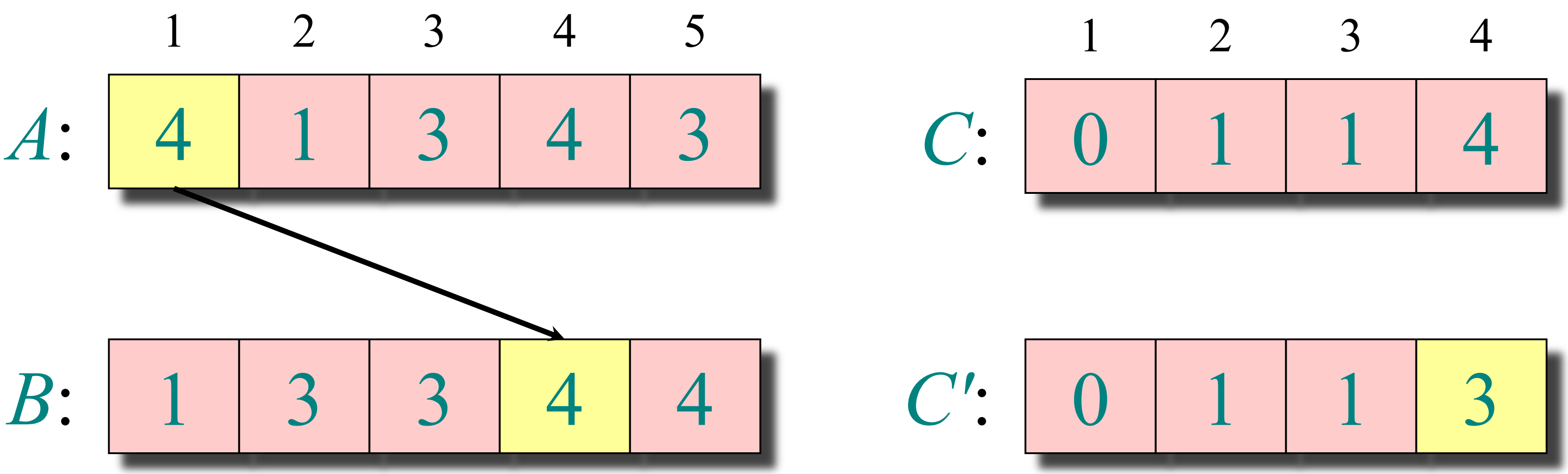


4. for $j \leftarrow n$ **downto** 1

do $B[C[A[j]]] \leftarrow \mathrm{A}[j]$

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

Schleife 4: Ordne neu



4. for $j \leftarrow n$ **downto** 1

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

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

Analyse

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

$\Theta(n)$ — 2. **for** $j \leftarrow 1$ **to** n
    do $C[A[j]] \leftarrow C[A[j]] + 1$

$\Theta(k)$ — 3. **for** $i \leftarrow 2$ **to** k
    do $C[i] \leftarrow C[i] + C[i-1]$

$\Theta(n)$ — 4. **for** $j \leftarrow n$ **downto** 1
    do $B[C[A[j]]] \leftarrow A[j]$
    $C[A[j]] \leftarrow C[A[j]] - 1$

$\Theta(n + k)$

Laufzeit: Wodurch wird sie reduziert?

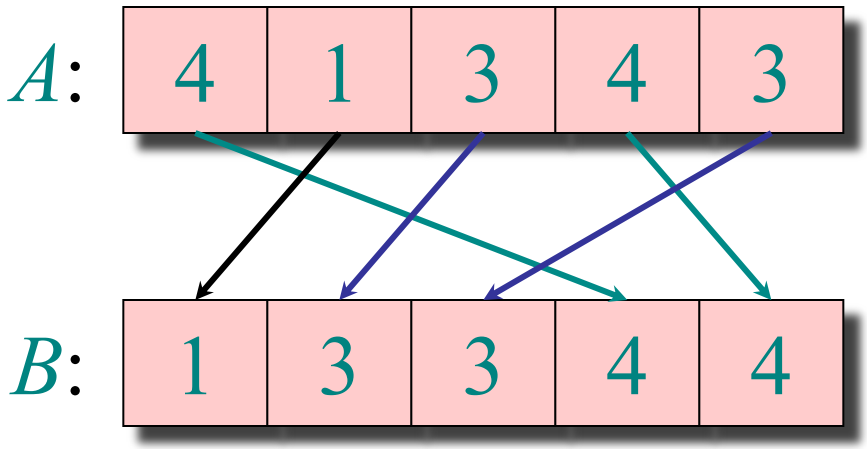
- Falls $k = O(n)$, dann braucht Counting-Sort $\Theta(n)$ viele Schritte.
- Aber theoretisch braucht doch Sortierung $\Omega(n \log n)$ viele Schritte!
- Gibt es ein Problem mit der Theorie?

Antwort:

- ***Sortieren durch Vergleichen*** liegt in $\Omega(n \log n)$
- Counting-Sort macht keine Vergleiche
- Counting-Sort verteilt einfach

Stabiles Sortieren

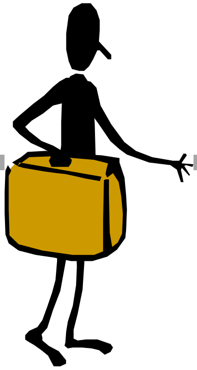
Counting-Sort ist stabil: die Eingabeordnung für gleiche Schlüssel bleibt bestehen



Warum ist das wichtig?

Welche andere Algorithmen haben diese Eigenschaft?

Zusammenfassung



- Bisher behandelt:
  - Sortieren durch Vergleichen (vorige Sitzungen)
  - Sortieren durch Verteilen (lineares Sortieren)
    - Counting Sort