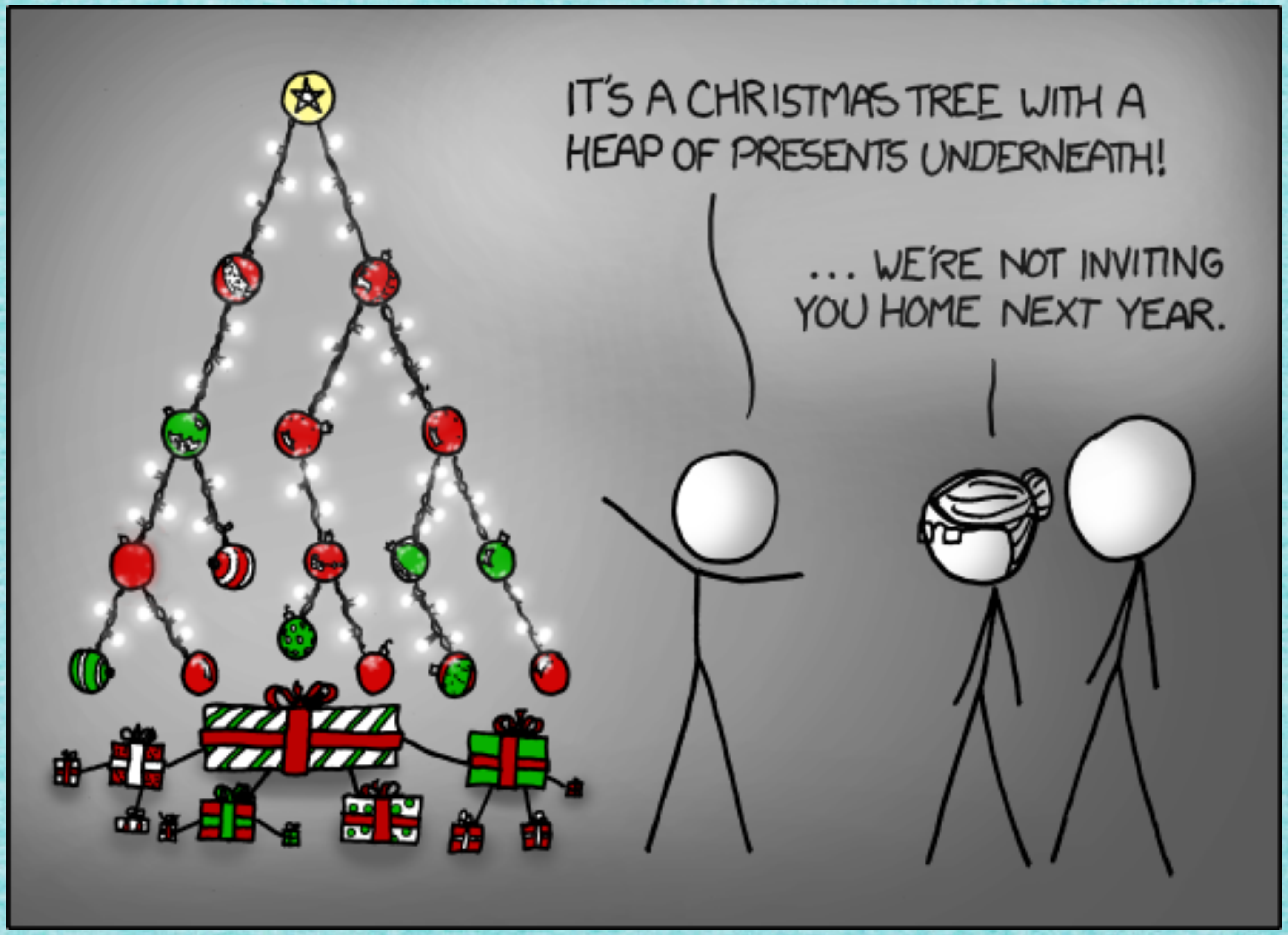
Kapitel 5: Sortieren

Algorithmen und Datenstrukturen
WS 2023/24

Prof. Dr. Sándor Fekete


IT'S A CHRISTMAS TREE WITH A HEAP OF PRESENTS UNDERNEATH!
... WE'RE NOT INVITING YOU HOME NEXT YEAR.

Wer ist eigentlich dieser knut?

Knut hat in Schweden eine fast 1.000 Jahre alte Tradition. Der **13.1.,** der St.-Knuts-Tag, hat seinen Namen von Knut IV., einem König, der von Weihnachten nicht genug bekommen konnte. Also hat er das **Fest auf 20 Tage verlängert.** Seitdem endet es jedes Jahr mit dem Knut-Tag, an dem wir **abschmücken und Platz für Neues schaffen.**

Donald Knuth

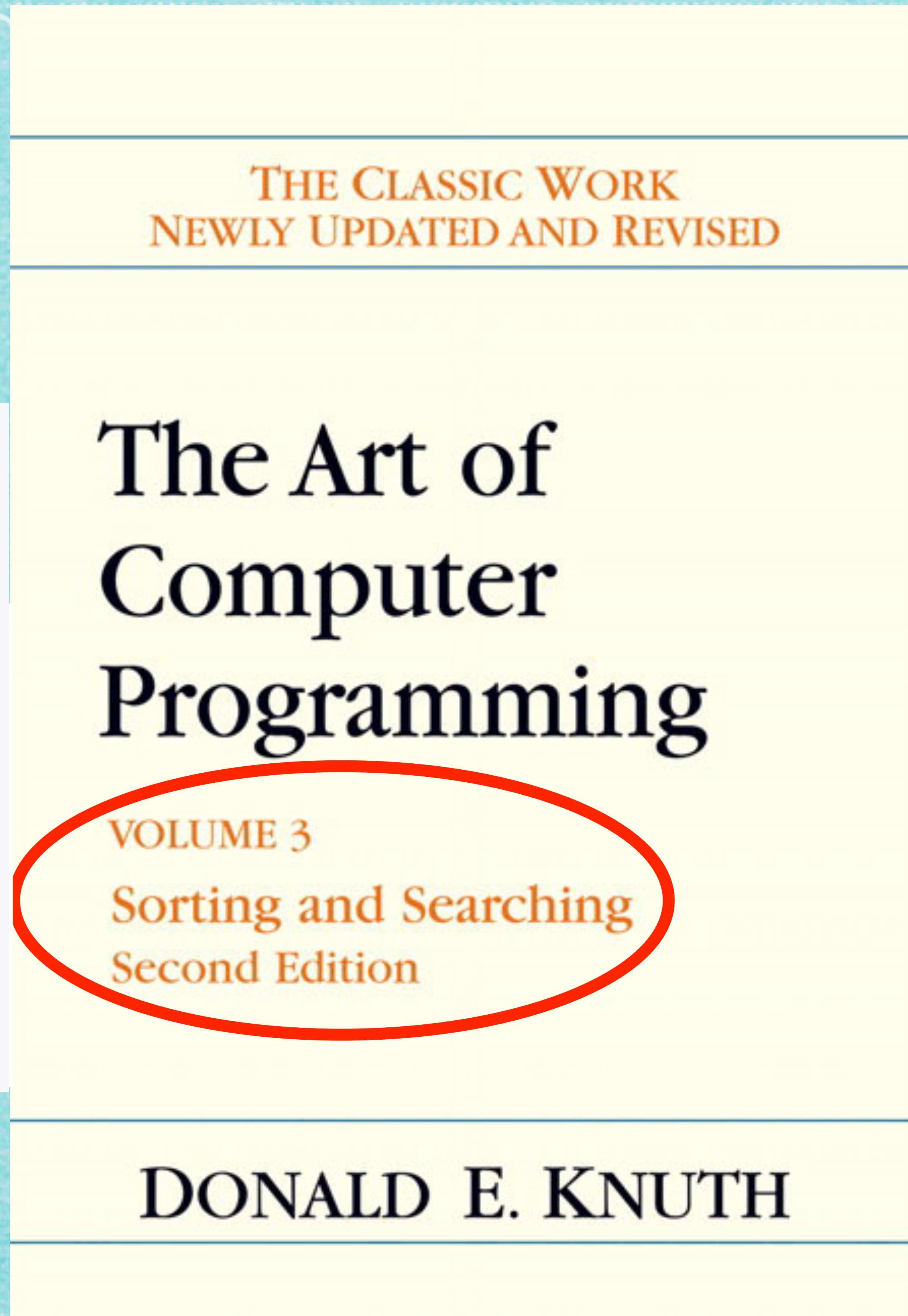
Knuth in 2005

Born Donald Ervin Knuth
January 10, 1938 (age 86)
Milwaukee, Wisconsin, U.S.

Donald Knuth

Knuth in 2005

Born	Donald Ervin Knuth January 10, 1938 (age 86) Milwaukee, Wisconsin, U.S.



5.0 Aufgabenstellung

Sortieren von Objekten

Gegeben: n Objekte unterschiedlicher Größe

23 17 13 19 33 28 15

Gesucht: Eine Sortierung nach Größe

Sortieren von Objekten

Gegeben: n Objekte unterschiedlicher Größe

13 15 17 19 23 28 33

Gesucht: Eine Sortierung nach Größe

Sortieren von Objekten

Gegeben: n Objekte unterschiedlicher Größe

23 17 13 19 33 28 15

Gesucht: Eine Sortierung nach Größe

Sortieren von Objekten

Gegeben: n Objekte unterschiedlicher Größe

23 17 19 33 28 15

13

Gesucht: Eine Sortierung nach Größe

Sortieren von Objekten

Gegeben: n Objekte unterschiedlicher Größe

23 17 19 33 28

13 15

Gesucht: Eine Sortierung nach Größe

Sortieren von Objekten

Gegeben: n Objekte unterschiedlicher Größe

23 19 33 28

13 15 17

Gesucht: Eine Sortierung nach Größe

Sortieren von Objekten

Gegeben: n Objekte unterschiedlicher Größe

23 33 28

13 15 17 19

Gesucht: Eine Sortierung nach Größe

Sortieren von Objekten

Gegeben: n Objekte unterschiedlicher Größe

33 28

13 15 17 19 23

Gesucht: Eine Sortierung nach Größe

Sortieren von Objekten

Gegeben: n Objekte unterschiedlicher Größe

33

13 15 17 19 23 28

Gesucht: Eine Sortierung nach Größe

Sortieren von Objekten

Gegeben: n Objekte unterschiedlicher Größe

13 15 17 19 23 28 33

Gesucht: Eine Sortierung nach Größe

Sortieren von Objekten

Gegeben: n Objekte unterschiedlicher Größe

13 15 17 19 23 28 33

Zahl der Vergleiche: $O(n^2)$

„Geht‘s nicht noch etwas schneller?“

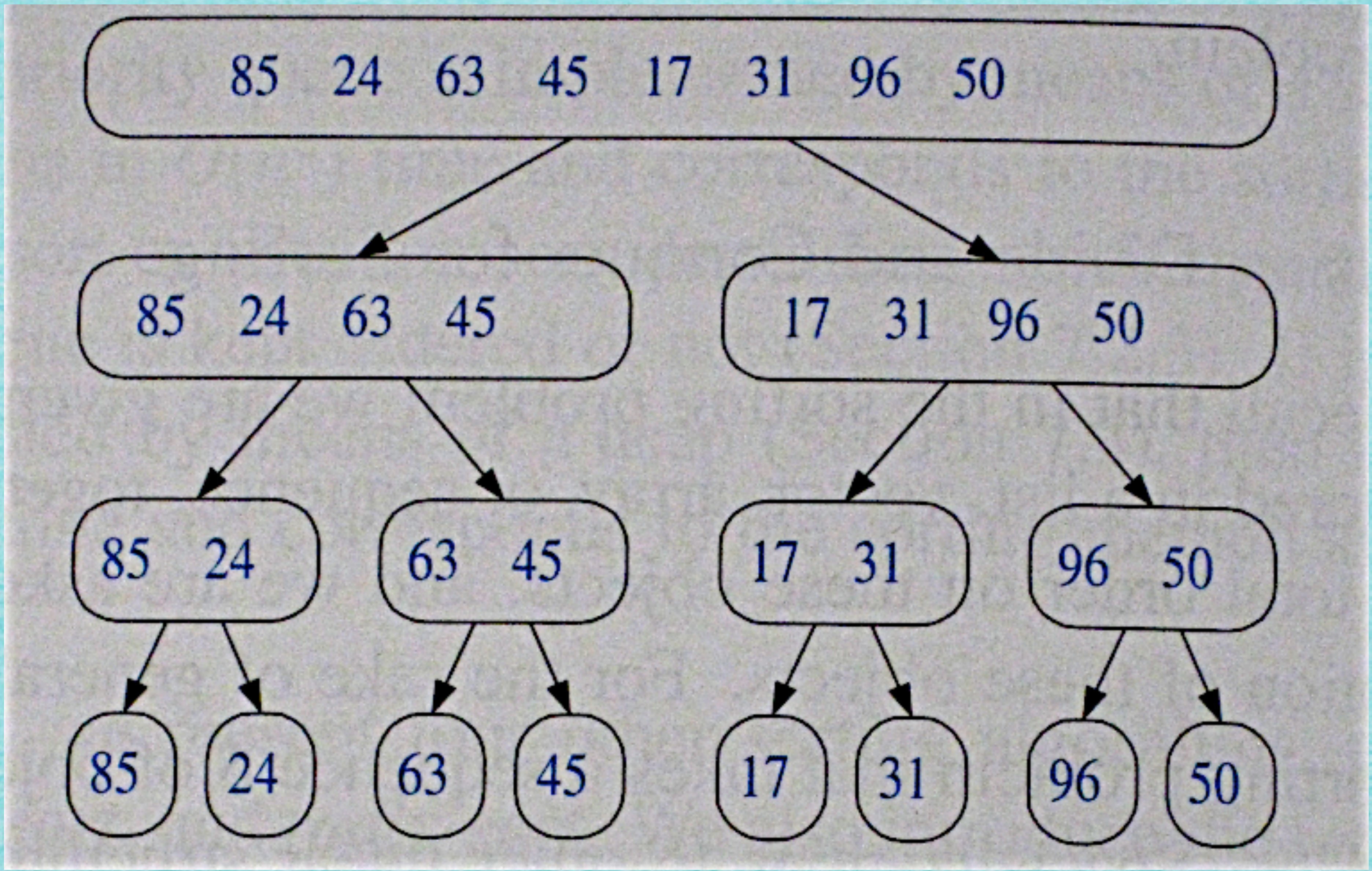
Sortieren von Objekten

Gegeben: n Objekte unterschiedlicher Größe

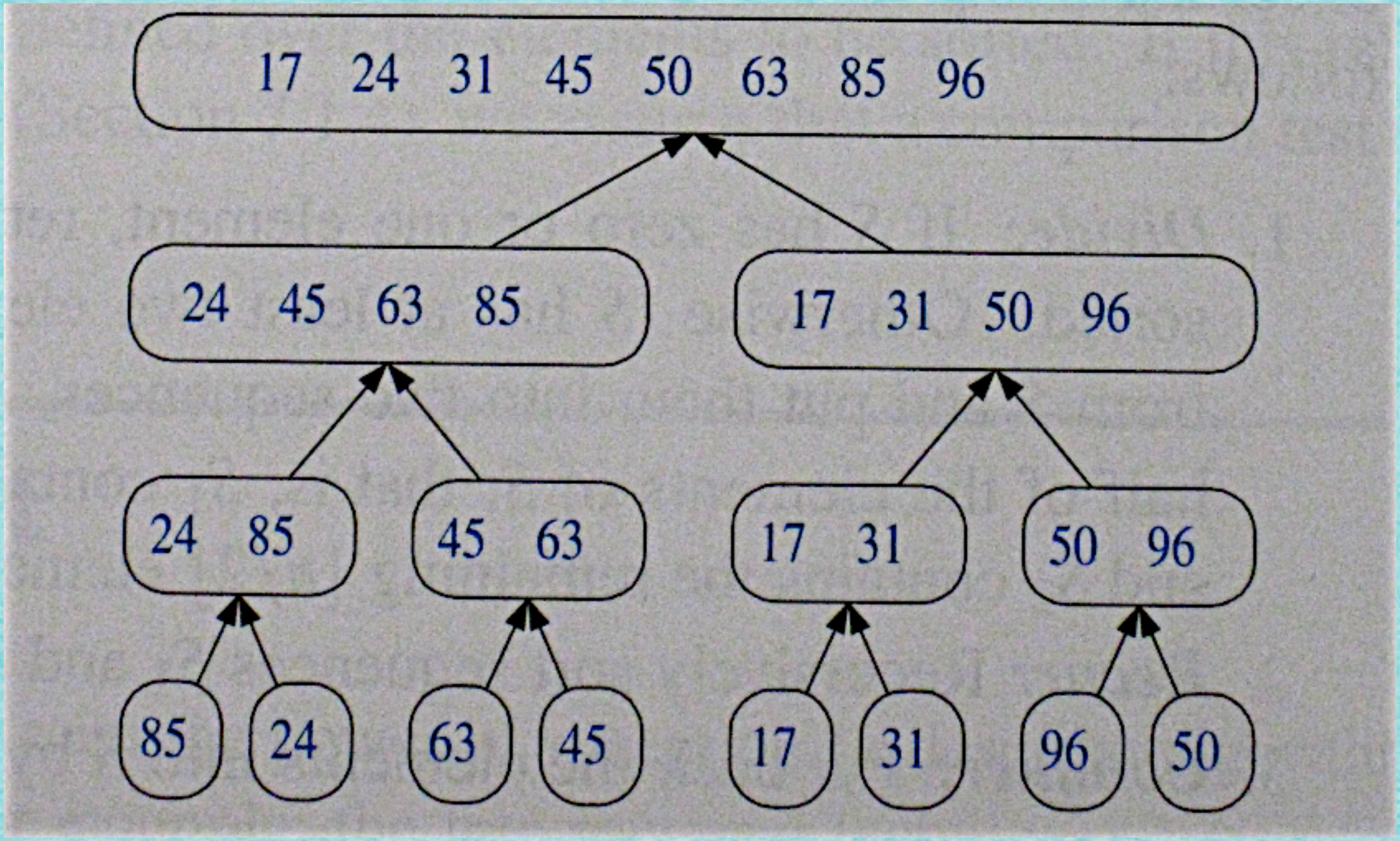
13 15 17 19 23 28 33

Zahl der Vergleiche: $O(n \log n)$

5.1 Mergesort



5.1 Mergesort



Algorithmus 5.1

INPUT: Subarray von A=[1,...,n], der bei Index p beginnt und bei Index r endet, d.h. A[p,...,r]

OUTPUT: Sortierter Subarray

MERGE-SORT(A,p,r)

```
if p < r
   then q ← ⌊(p + r)/2⌋
        MERGE-SORT(A, p, q)
        MERGE-SORT(A, q + 1, r)
        MERGE(A, p, q, r)
```

Subroutine 5.2

<u>INPUT:</u> Zwei sortierte Subarrays von A=[1,...,n], d.h. A[p,...,q] und A[q+1,...,r]

<u>OUTPUT:</u> Sortierter Subarray A[p,...,r]

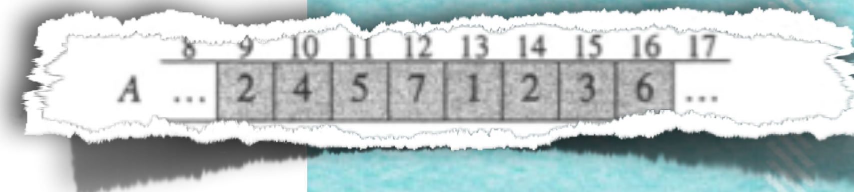
MERGE(A, p, q, r)

Subroutine 5.2

<u>INPUT:</u> Zwei sortierte Subarrays von A=[1,...,n], d.h. A[p,...,q] und A[q+1,...,r]

<u>OUTPUT:</u> Sortierter Subarray A[p,...,r]

```
MERGE(A, p, q, r)
n₁ ← q − p + 1
n₂ ← r − q
erzeuge die Felder L[1 . . n₁ + 1] und R[1 . . n₂ + 1]
for i ← 1 to n₁
        do L[i] ← A[p + i − 1]
for j ← 1 to n₂
        do R[j] ← A[q + j]
```

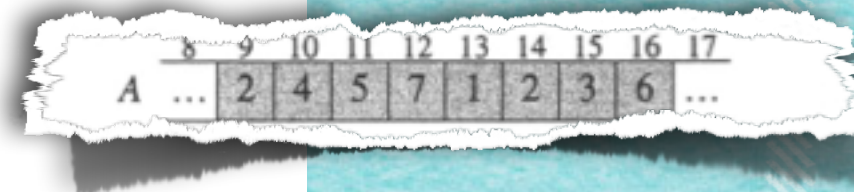


Subroutine 5.2

INPUT: Zwei sortierte Subarrays von A=[1,...,n], d.h. A[p,...,q] und A[q+1,...,r]

OUTPUT: Sortierter Subarray A[p,...,r]

```
MERGE(A, p, q, r)
  n₁ ← q − p + 1
  n₂ ← r − q
  erzeuge die Felder L[1 .. n₁ + 1] und R[1 .. n₂ + 1]
  for i ← 1 to n₁
        do L[i] ← A[p + i − 1]
  for j ← 1 to n₂
        do R[j] ← A[q + j]
  L[n₁ + 1] ← ∞
  R[n₂ + 1] ← ∞
```



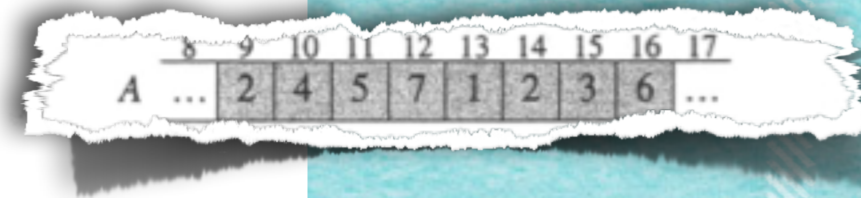
Subroutine 5.2

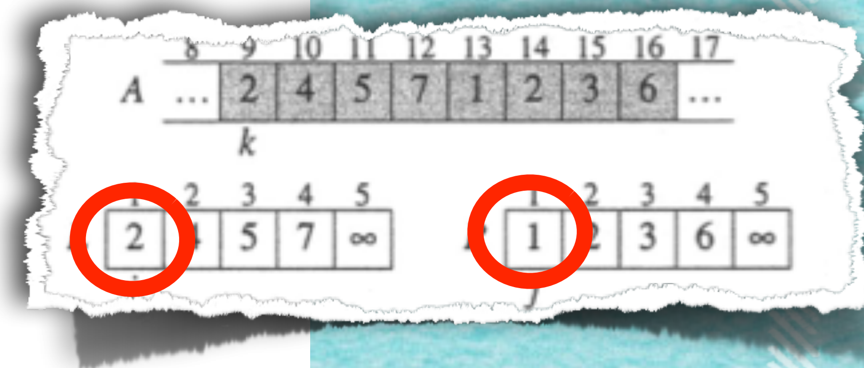
<u>INPUT:</u> Zwei sortierte Subarrays von A=[1,...,n], d.h. A[p,...,q] und A[q+1,...,r]

<u>OUTPUT:</u> Sortierter Subarray A[p,...,r]

MERGE(A, p, q, r)

$n_1 \leftarrow q - p + 1$
$n_2 \leftarrow r - q$
erzeuge die Felder $L[1\,..\,n_1 + 1]$ und $R[1\,..\,n_2 + 1]$
for $i \leftarrow 1$ **to** n_1
    do $L[i] \leftarrow A[p + i - 1]$
for $j \leftarrow 1$ **to** n_2
    do $R[j] \leftarrow A[q + j]$
$L[n_1 + 1] \leftarrow \infty$
$R[n_2 + 1] \leftarrow \infty$
$i \leftarrow 1$
$j \leftarrow 1$



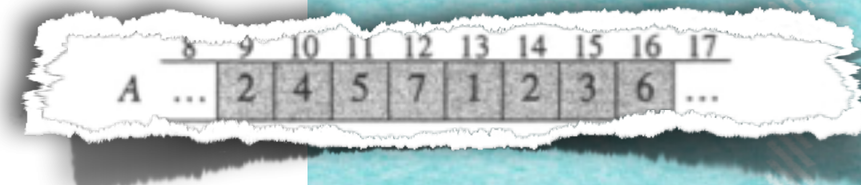


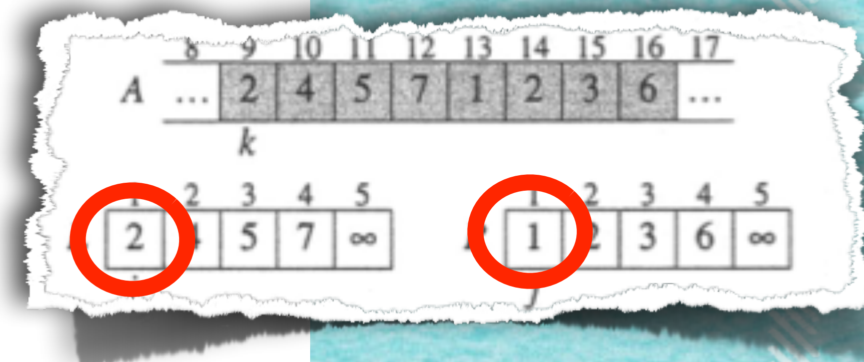
Subroutine 5.2

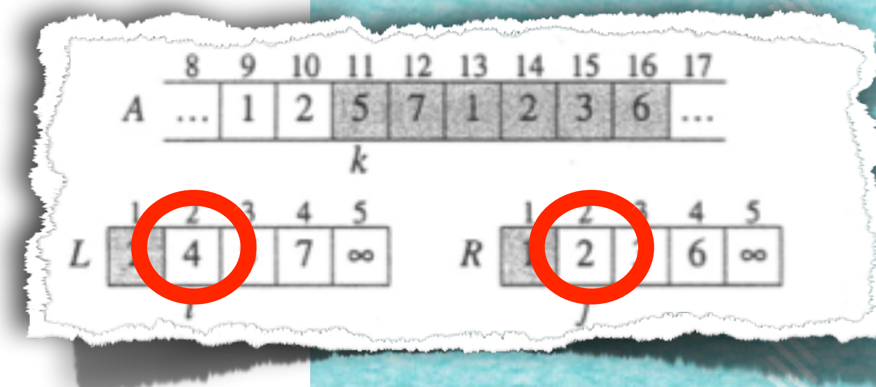
INPUT: Zwei sortierte Subarrays von A=[1,...,n], d.h. A[p,...,q] und A[q+1,...,r]

OUTPUT: Sortierter Subarray A[p,...,r]

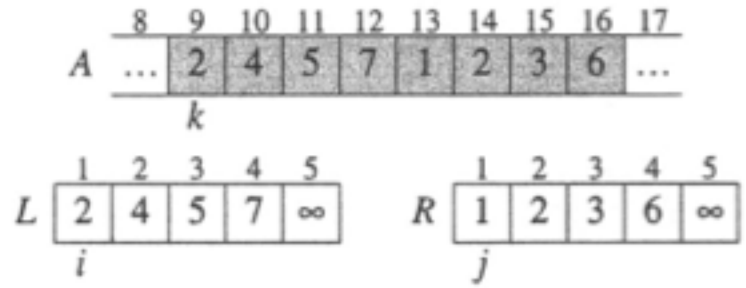
```
MERGE(A, p, q, r)
  n1 ← q − p + 1
  n2 ← r − q
  erzeuge die Felder L[1 .. n1 + 1] und R[1 .. n2 + 1]
  for i ← 1 to n1
        do L[i] ← A[p + i − 1]
  for j ← 1 to n2
        do R[j] ← A[q + j]
  L[n1 + 1] ← ∞
  R[n2 + 1] ← ∞
  i ← 1
  j ← 1
  for k ← p to r
        do if L[i] ≤ R[j]
              then A[k] ← L[i]
                   i ← i + 1
              else A[k] ← R[j]
                   j ← j + 1
```



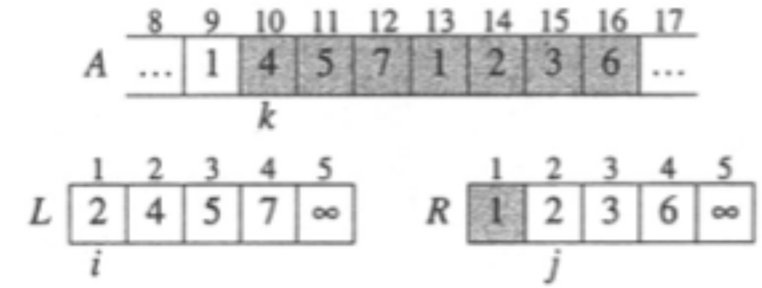




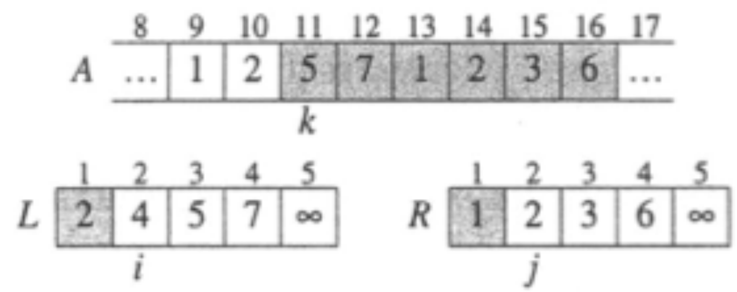
5.1 Mergesort



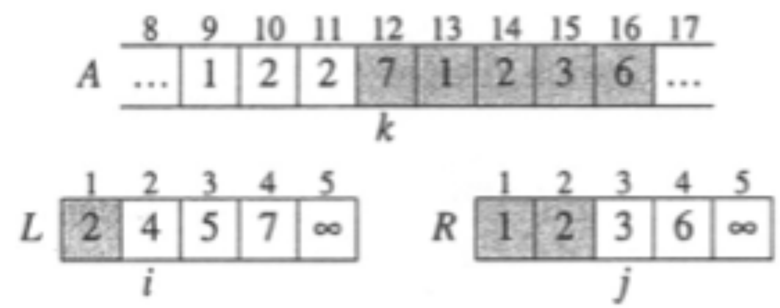
(a)



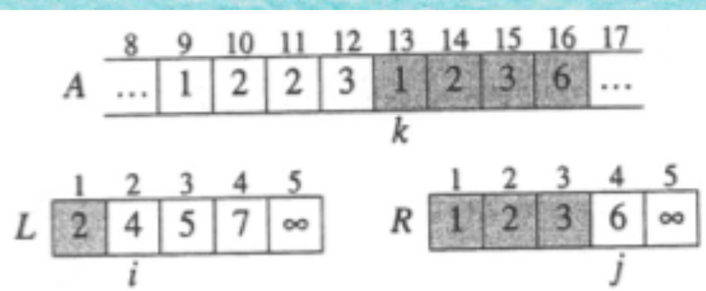
(b)



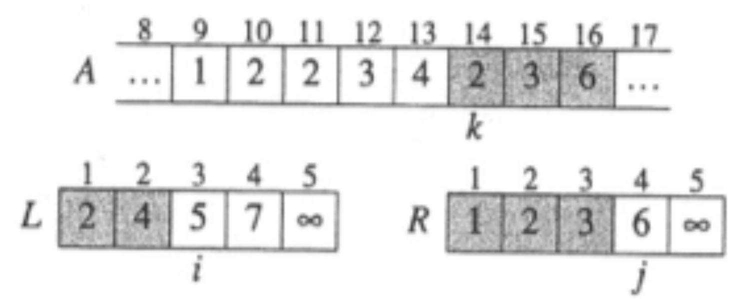
(c)



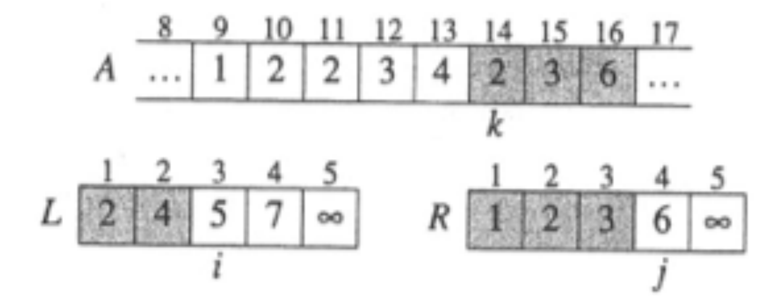
(d)



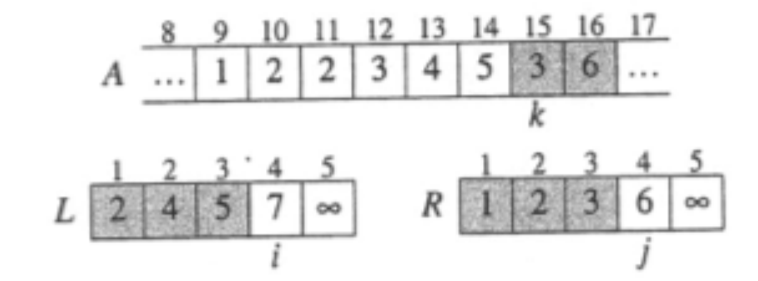
(e)



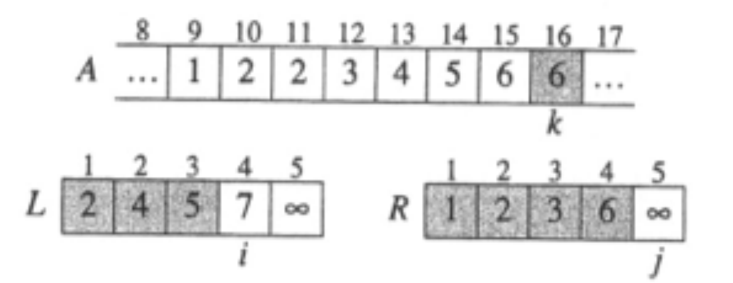
(f)


8 9 10 11 12 13 14 15 16 17
A … 1 2 2 3 4 2 3 6 …
k
1 2 3 4 5
L 2 4 5 7 ∞
i
1 2 3 4 5
R 1 2 3 6 ∞
j

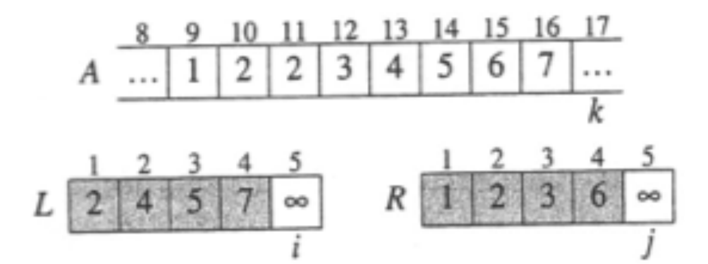
(f)


8 9 10 11 12 13 14 15 16 17
A … 1 2 2 3 4 5 3 6 …
k
1 2 3 4 5
L 2 4 5 7 ∞
i
1 2 3 4 5
R 1 2 3 6 ∞
j

(g)


8 9 10 11 12 13 14 15 16 17
A … 1 2 2 3 4 5 6 6 …
k
1 2 3 4 5
L 2 4 5 7 ∞
i
1 2 3 4 5
R 1 2 3 6 ∞
j

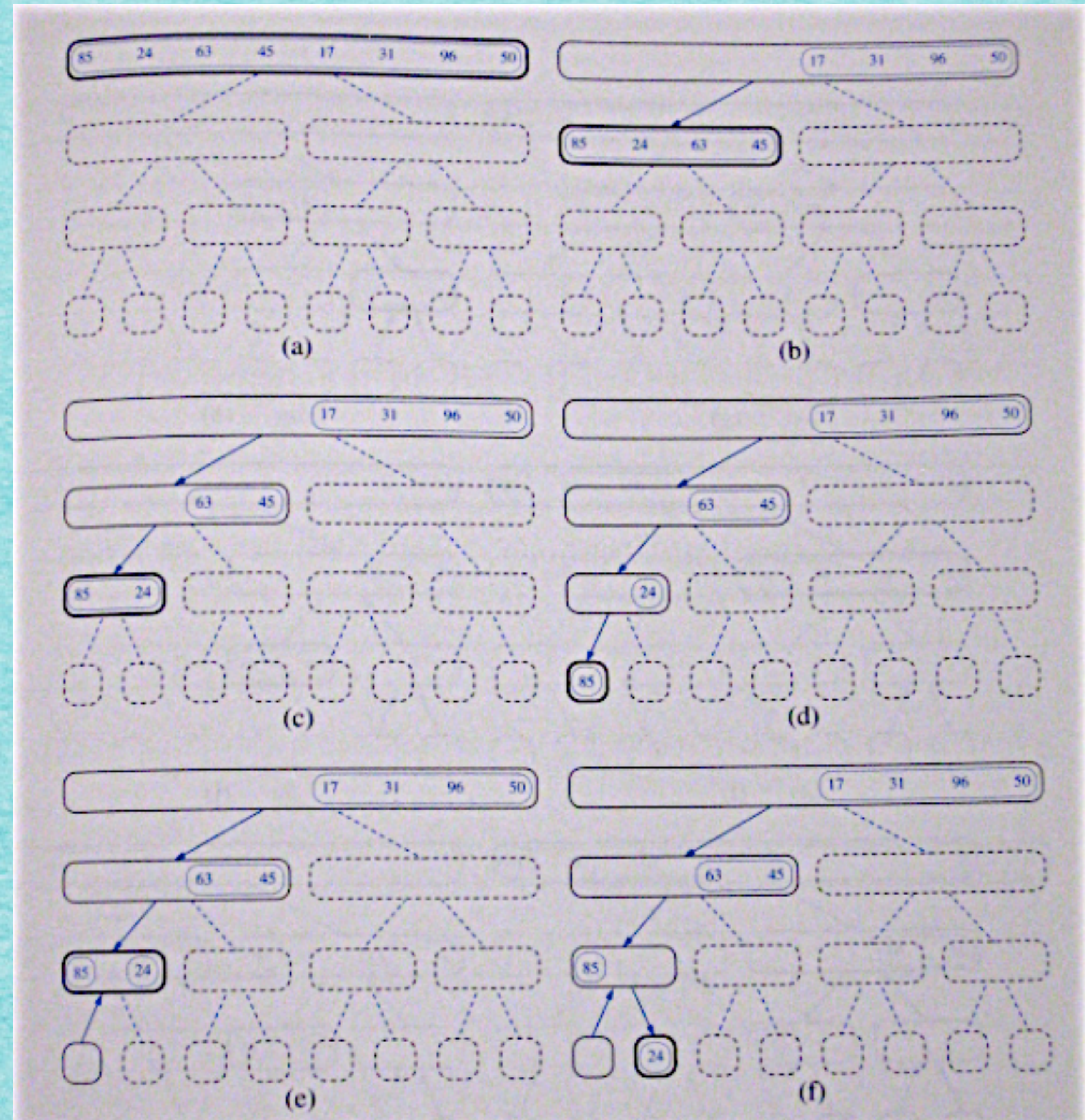
(h)


8 9 10 11 12 13 14 15 16 17
A … 1 2 2 3 4 5 6 7 …
k
1 2 3 4 5
L 2 4 5 7 ∞
i
1 2 3 4 5
R 1 2 3 6 ∞
j

(i)

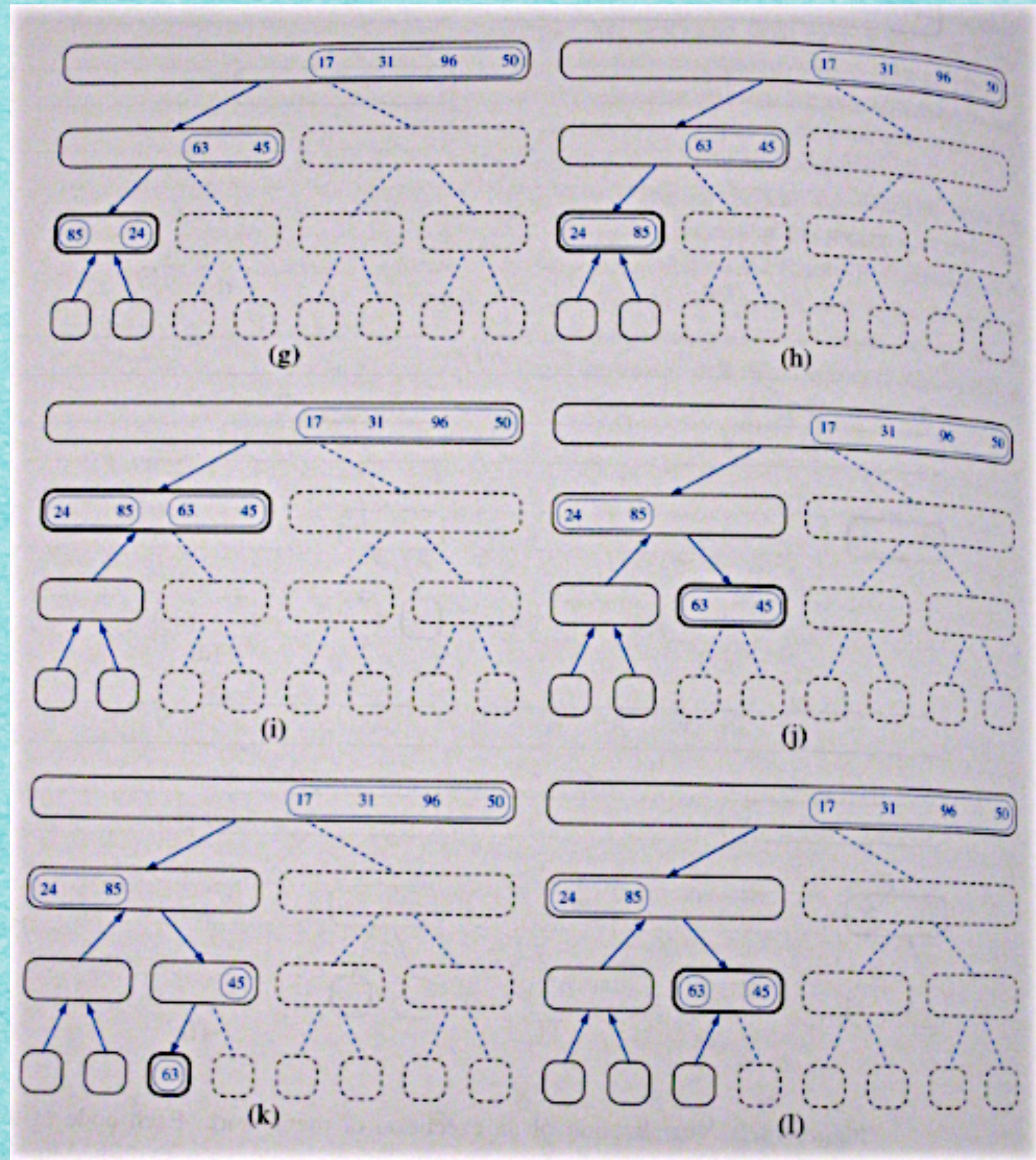
Ablauf makroskopisch

```
if p < r
    then q ← ⌊(p + r)/2⌋
         MERGE-SORT(A, p, q)
         MERGE-SORT(A, q + 1, r)
         MERGE(A, p, q, r)
```



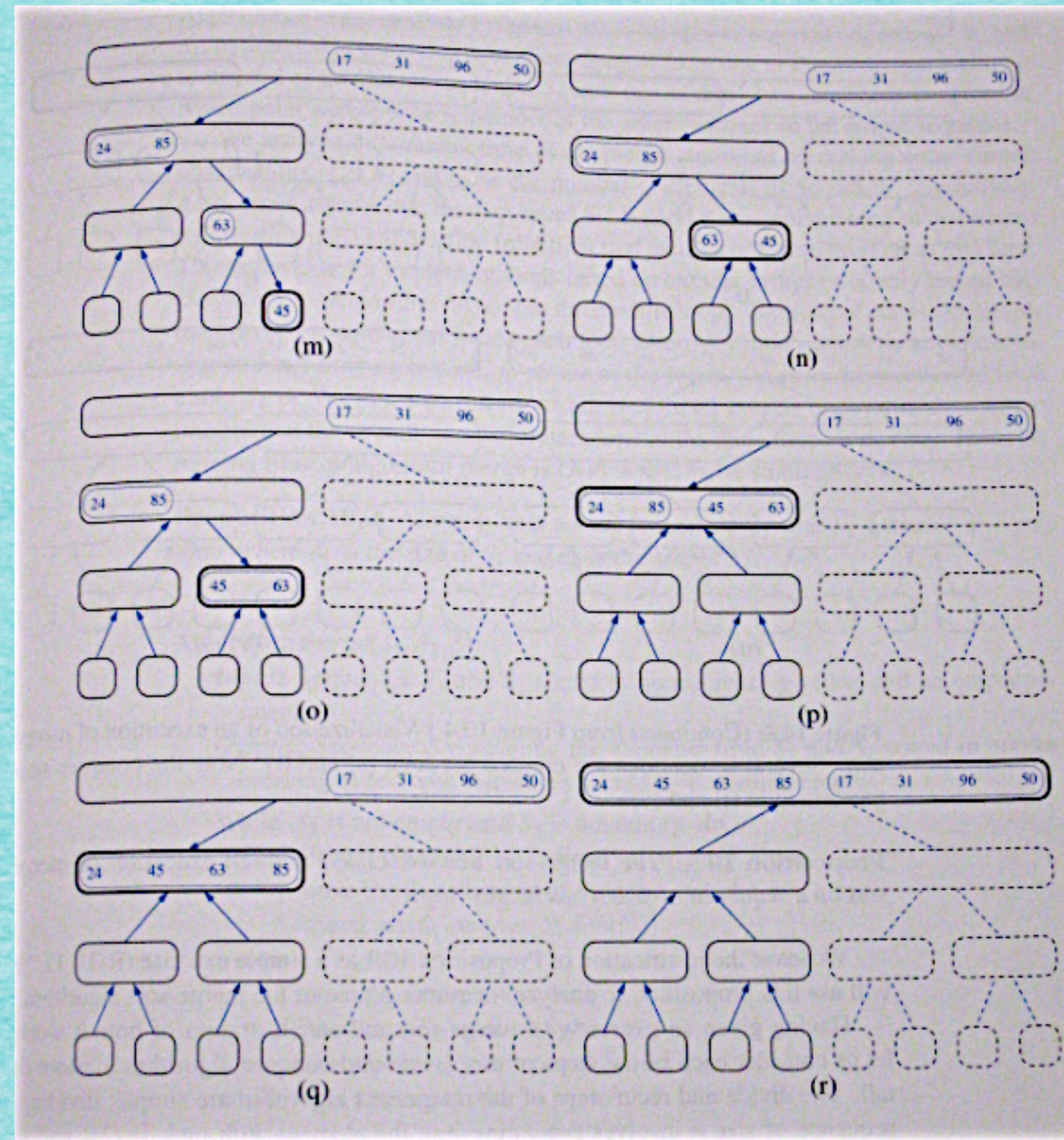
Ablauf makroskopisch

```
if p < r
    then q ← ⌊(p + r)/2⌋
         MERGE-SORT(A, p, q)
         MERGE-SORT(A, q + 1, r)
         MERGE(A, p, q, r)
```



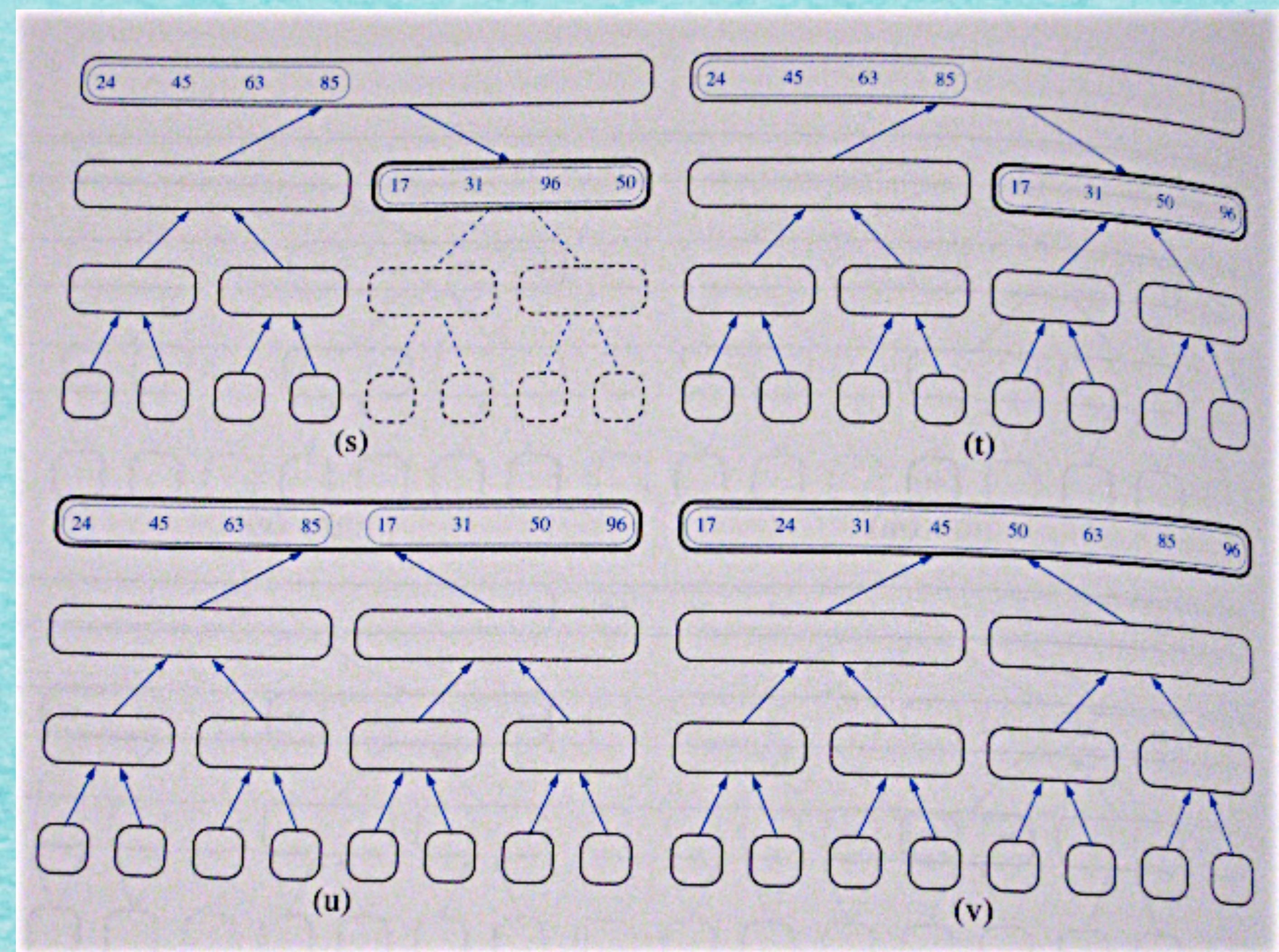
Ablauf makroskopisch

```
if p < r
    then q ← ⌊(p + r)/2⌋
         MERGE-SORT(A, p, q)
         MERGE-SORT(A, q + 1, r)
         MERGE(A, p, q, r)
```



Ablauf makroskopisch

```
if p < r
   then q ← ⌊(p + r)/2⌋
        MERGE-SORT(A, p, q)
        MERGE-SORT(A, q + 1, r)
        MERGE(A, p, q, r)
```

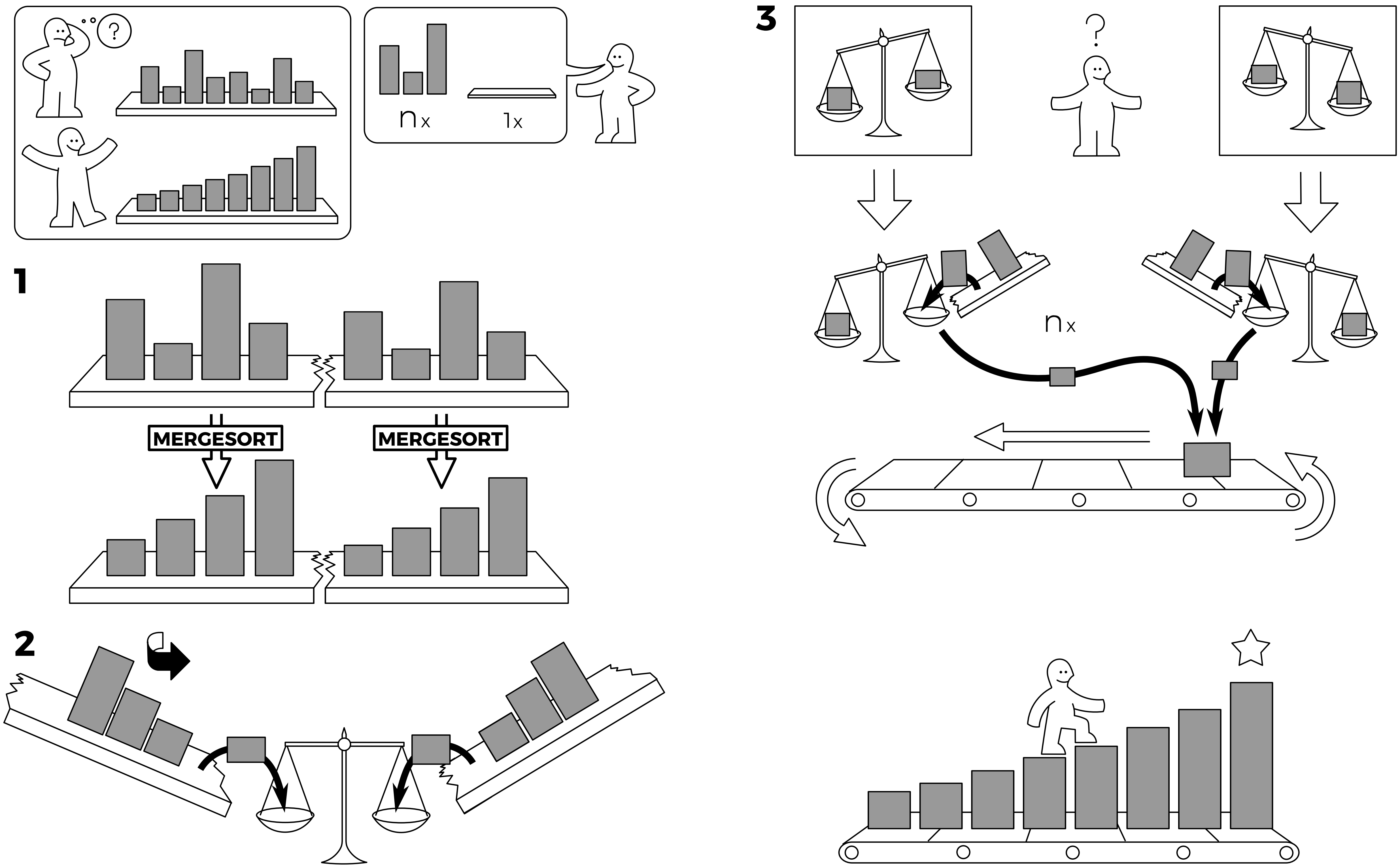


5.1 Mergesort

MERGE SÖRT

idea-instructions.com/merge-sort/

IDEA



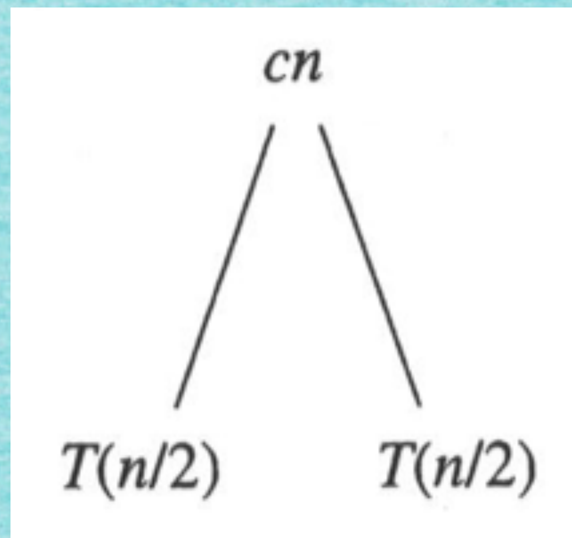
5.1.3 Laufzeit von Mergesort

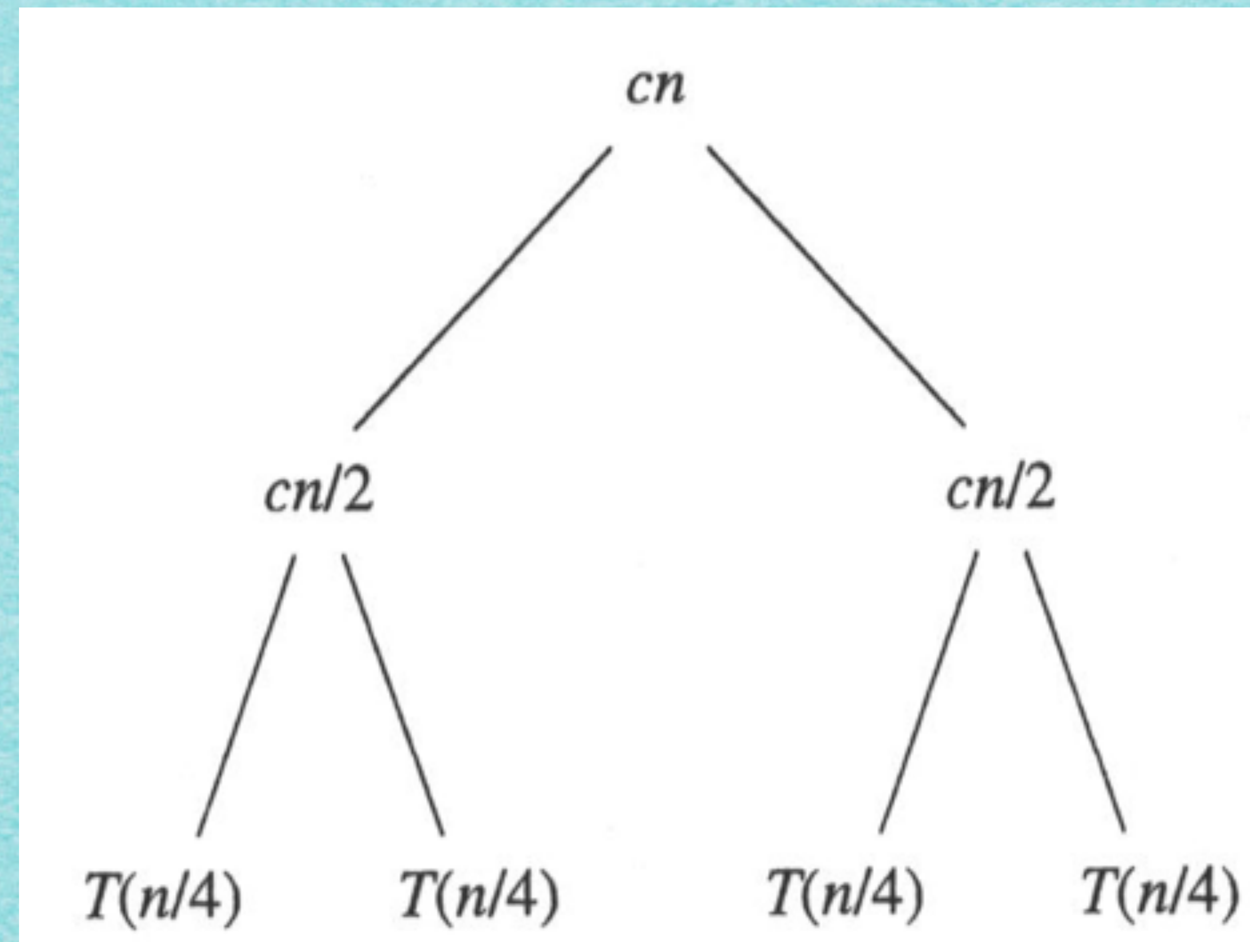
Wie viele Schritte benötigt Merge-Sort für einen Array der Länge n?

$$T(n)$$

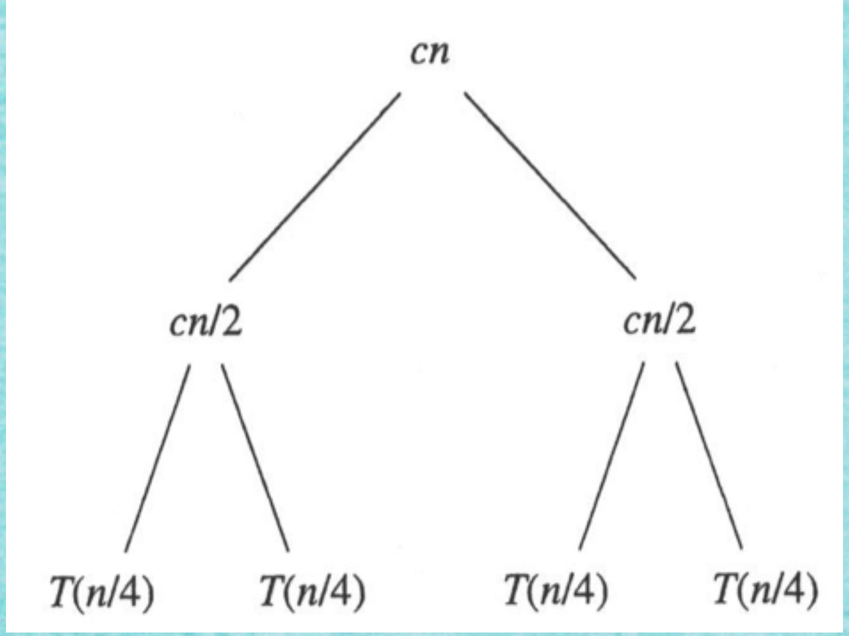
5.1.3 Laufzeit von Mergesort

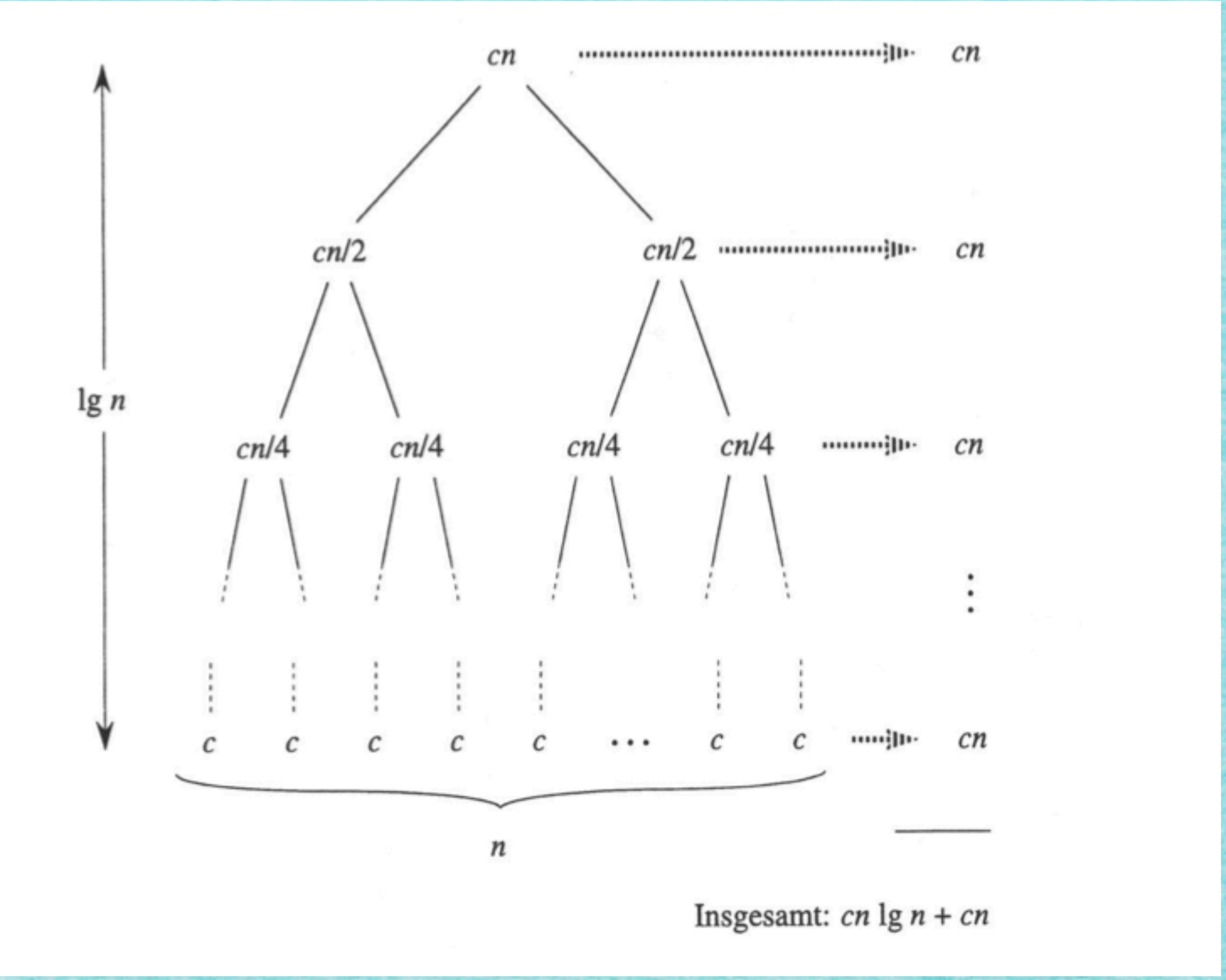
$T(n)$





5.1.3 Laufzeit von Mergesort





<u>Satz 5.3 (Komplexität von Mergesort)</u>
Für einen n-elementigen Array A hat Mergesort eine Laufzeit von O(n log n).

Mehr an der Tafel!

s.fekete@tu-bs.de