Laufzeit Bubblesort (1) demogr.

Wie hoch ist die **worst-case** Laufzeit von Bubblesort für n Schlüssel?

- (1) $T(n) = \Theta(\log n)$
- (2) $T(n) = \Theta(n)$
- (3) $T(n) = \Theta(n \log n)$
- (4) $T(n) = \Theta(n^2)$
- (5) Ich kenn nur Bubblegum...

Auflösung: (4) $T(n) = \Theta(n^2)$

Laufzeit Bubblesort (2) demogr.

Wie hoch ist die Laufzeit von Bubblesort auf folgender Eingabe:
$(3, 2, 1, 6, 5, 4, 9, 8, 7, \ldots, n, n-1, n-2)$

- (1) $T(n) = \Theta(\log n)$
- (2) $T(n) = \Theta(n)$
- (3) $T(n) = \Theta(n \log n)$
- (4) $T(n) = \Theta(n^2)$
- (5) Hä?

Auflösung: (2) $T(n) = \Theta(n)$

Laufzeit Bubblesort (3) demogr.

Wie hoch ist die **erwartete** Laufzeit von Bubblesort für n Schlüssel?

- (1) $T(n) = \Theta(\log n)$
- (2) $T(n) = \Theta(n)$
- (3) $T(n) = \Theta(n \log n)$
- (4) $T(n) = \Theta(n^2)$
- (5) $T(n) = \Theta(\text{erwartete Laufzeit von Bubblesort})$

Auflösung: (4) $T(n) = \Theta(n^2)$

Swaps multiple choice

Welche Sortieralgorithmen haben eine Best-Case Laufzeit von $O(n)$?

- (1) Bubblesort
- (2) Selectionsort
- (3) Insertionsort
- (4) Quicksort

Auflösung: (1) & (3)

Quicksort (1) demogr.

Welche Laufzeit hat **Quicksort** auf der Eingabe $(3, 2, 1, 6, 5, 4, 9, 8, 7 \ldots, n, n-1, n-2)$, wenn ein Gegner jeweils das Pivotelement bestimmen darf?

- (1) $\Theta(\log n)$
- (2) $\Theta(n)$
- (3) $\Theta(n \log n)$
- (4) $\Theta(n^2)$
- (5) $\Theta(2^n)$

Auflösung: (4) $\Theta(n^2)$

Quicksort (2) demogr.

Welche Laufzeit hat **Quicksort** auf der Eingabe $(3, 2, 1, 6, 5, 4, 9, 8, 7 \ldots, n, n-1, n-2)$, wenn als Pivotelement jeweils der Median des Teilproblems genommen wird.

- (1) $\Theta(\log n)$
- (2) $\Theta(n)$
- (3) $\Theta(n \log n)$
- (4) $\Theta(n^2)$
- (5) $\Theta(2^n)$

Auflösung: (3) $\Theta(n \log n)$.

Quicksort (3) demogr.

Wenn Quicksort auf einer Eingabe der Größe n eine Laufzeit von $\Theta(n^2)$ zeigt, dann werden auch $\Theta(n^2)$ Element-Vertauschungen beim Partitionieren durchgeführt.

- (1) Stimmt.
- (2) Stimmt nicht.
- (3) Keine Ahnung.

Auflösung: (2) Stimmt nicht.

Bsp.: vorsortierte Eingabe und Pivot immer kleinstes/linkes Element, dann nur linear viele Vertauschungen (Pivot hin/zurück).

Auswahlproblem demogr.

Betrachten Sie den Algorithmus für das **Auswahlproblem**, bei dem wir das s-größte Element in einem Array mit n Elementen bestimmen.

Wie hoch ist die erwartete Laufzeit wenn das Pivotelement immer **uniform zufällig** gewählt wird?

- (1) $\Theta(n/\log n)$
- (2) $\Theta(n)$
- (3) $\Theta(n \log n)$
- (4) $\Theta(n^2)$
- (5) ist mir nicht mehr klar.

Auflösung: (2) $\Theta(n)$

Auswahlproblem (2) demogr.

Betrachten Sie den Algorithmus für das **Auswahlproblem**, bei dem wir das s-größte Element in einem Array mit n Elementen bestimmen.

Wie hoch ist die Laufzeit wenn **ein Gegner** jeweils das Pivotelement bestimmen kann?

- (1) $\Theta(n)$
- (2) $\Theta(n \log_s n)$
- (3) $\Theta(n \log(sn))$
- (4) $\Theta(ns)$
- (5) $\Theta(n^2)$

Auflösung: (5) $\Theta(n^2)$

Auswahlproblem (3) demogr.

Wir möchten nun die Menge der s größten Elemente in beliebiger Reihenfolge ausgeben. Welche Laufzeit ist möglich?

- (1) $\Theta(s)$
- (2) $\Theta(n)$
- (3) $\Theta(n \cdot \log s)$
- (4) $\Theta(ns)$
- (5) $\Theta(n^2)$

Auflösung: (2) $\Theta(n)$

Mergesort demogr.

Betrachten Sie **3-Teile-Mergesort**:

1. Teile das Array der Größe n in drei möglichst gleich große Teile
2. Sortiere jeden der Teile rekursiv mit 3-Teile-Mergesort
3. Mische die drei sortierten Teile zusammen.

Wie hoch ist die worst-case Laufzeit von 3-Teile-Mergesort?

- (1) $\Theta(n \log n)$
- (2) $\Theta(n \cdot \sqrt[3]{n})$
- (3) $\Theta(n^{3/2} \cdot \log n)$
- (4) $\Theta(n^2)$
- (5) $\Theta(n^3)$

Auflösung: (1) $\Theta(n \log n)$

Externspeicher-Sortieren demogr.

Sei $B = M^{1/3}$ und $n = M^{1/2}$. Wieviele Mergephasen braucht dann das verbesserte Externspeicher-Sortieren?

- (1) $\Theta(1)$ Phasen
- (2) $\Theta(\log \frac{M}{B})$ Phasen
- (3) $\Theta(\log n)$ Phasen
- (4) $\Theta(n)$ Phasen
- (5) ist mir nicht mehr klar.

Auflösung: (1) $\Theta(1)$ Phasen

Externspeicher - Umsortieren demogr.

Sei in Z[1..n] eine Permutation der Zahlen $1, 2, \ldots, n$ abgelegt.
Wieviel I/O produziert folgendes Codestück im Worst-Case

```
int[1..n] X,Y,Z;
for i=1 to n do X[i]:=Y[Z[i]];
```

- (1) $\Theta(1)$ I/Os.
- (2) $\Theta(n/B)$ I/Os.
- (3) $\Theta(\text{sort}(n))$ I/Os.
- (4) $\Theta(n)$ I/Os.
- (5) $\Theta(n \log n)$ I/Os.
- (6) Θhne meinen Anwalt sag ich gar nichts!

Auflösung: (4), mit ein wenig Sortieren geht es aber besser:

Externspeicher - Umsortieren

```
int[1..n] X,Y,Z;
for i=1 to n do X[i]:=Y[Z[i]];
```

Wende folgendes Vorgehen an:

```
SCAN Z:       (Z[1]=17,1),   (Z[2]=5,2),    ...
SORT(1st):    (Z[73]=1,73), (Z[12]=2,12),  ...
par. SCAN :   (Y[1],73),     (Y[2],12),     ...
SORT(2nd):    (Y[Z[1]],1),   (Y[Z[2]],2),   ...
par. SCAN :  X[1]=Y[Z[1]], X[2]=Y[Z[2]],  ...
```

Das braucht nur $O(\mathrm{sort}(n))$ I/Os, unabhängig von Permutation in Z[].

Radixsort (1) demogr.

Wie sollte die Basis b gewählt werden, wenn n Zahlen aus dem Bereich $\{0, \ldots, n^2 - 1\}$ sortiert werden sollen?

- (1) $b = 2$.
- (2) $b = 10$.
- (3) $b = \sqrt{n}$.
- (4) $b = n$.
- (5) $b = n^2$.
- (6) ist egal.

Auflösung: (4) $b = n$

Radixsort (2) demogr.

In welcher optimalen worst-case Zeit kann man n ganze Zahlen aus dem Bereich $\{1, \ldots, n^{1.5}\}$ sortieren?

- (1) $T(n) = \Theta(n^{1/1.5})$
- (2) $T(n) = \Theta(n)$
- (3) $T(n) = \Theta(n \log n)$
- (4) $T(n) = \Theta(n^{1.5})$
- (5) $T(n) = \Theta(n^2)$

Auflösung: (2) $T(n) = \Theta(n)$

Radix vs. Merge demogr.

Es seien n Zahlen aus dem Bereich $\{0, \ldots, n^x\}$ zu sortieren. Bei welchem Parameter x ist Mergesort in etwa genauso schnell oder schneller als Radixsort?

- (1) $x = \Omega(1)$.
- (2) $x = \Omega(\log n)$.
- (3) $x = \Omega(\sqrt{n})$.
- (4) $x = \Omega(n)$.
- (5) niemals.

Auflösung: (2) $x = \Omega(\log n)$.

Verlassen des Hörsaals demogr.

Wie verfahren Sie, wenn Sie H V möglichst geräuschlos verlassen möchten?

- (1) Zügig und selbstbewusst nach vorne rausgehen.
- (2) Heimlich hinten rausschleichen.
- (3) Rückwärts durch die Hecke.
- (4) Ich würde doch niemals vorzeitig die Vorlesung verlassen!

Auflösung: (4) oder (1).

Paralleles Samplesort demogr.

Wieviele Prozessoren können für das parallele Samplesort höchstens sinnvoll eingesetzt werden?

- (1) $p = 1024$.
- (2) $p = \log n$.
- (3) $p = n^{1/3}$.
- (4) $p = n$.
- (5) beliebig viele.

Auflösung: (3) $p = n^{1/3}$

Paralleles Samplesort (2) demogr.

Welche Laufzeit hat das parallele Samplesort bei $p = \sqrt{n}$ Prozessoren?

- (1) keine Ahnung
- (2) $T(n) = \Theta(\sqrt{n})$
- (3) $T(n) = \Theta(\sqrt{n} \log n)$
- (4) $T(n) = \Theta(n)$
- (5) $T(n) = \Theta(n \log n)$

Auflösung: (5) $T(n) = \Theta(n \log n)$