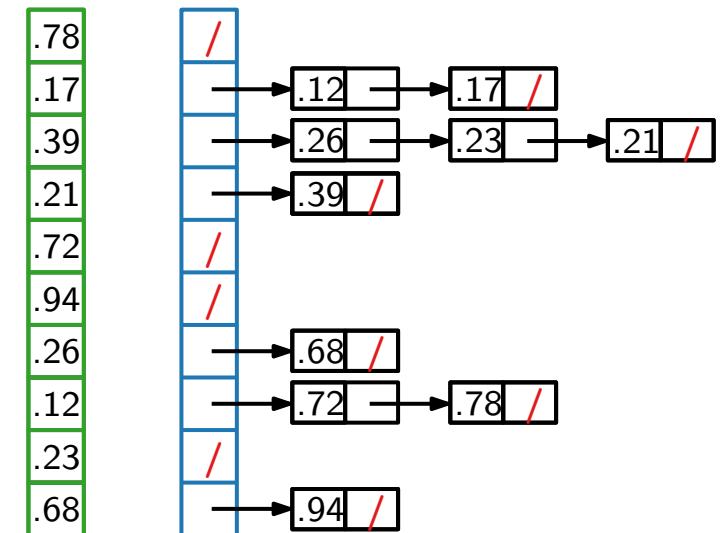
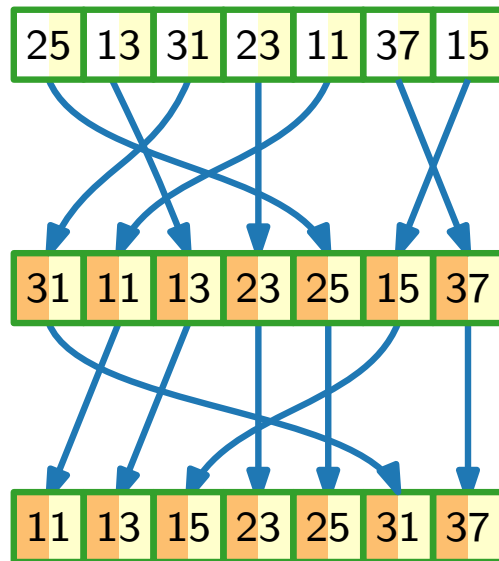


Algorithmen und Datenstrukturen

Vorlesung 9: Sortieren in Linearzeit



Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

vergleichsbasierter Sortieralgorithmus

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)



Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe



Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq/\geq$) eines Vergleichs

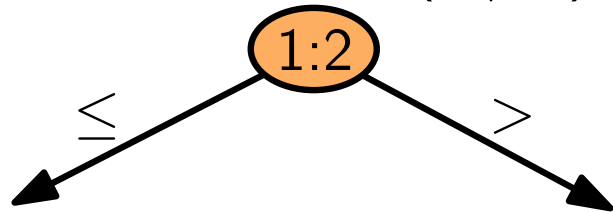


Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs

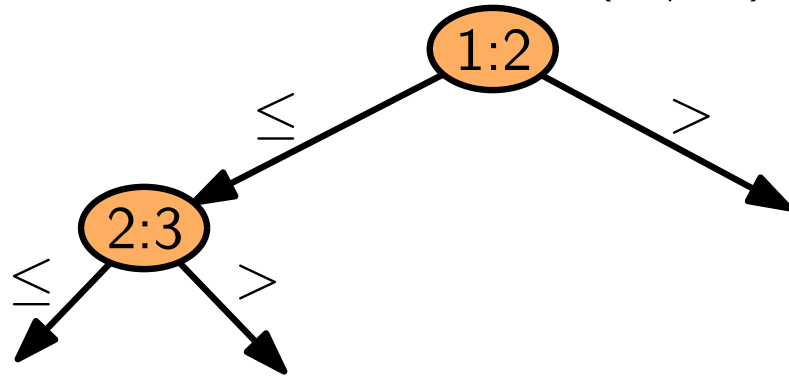


Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs

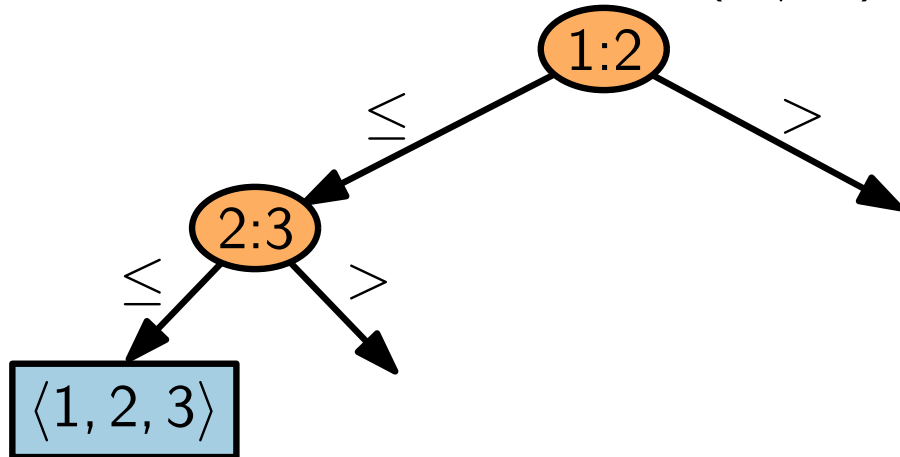


Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs

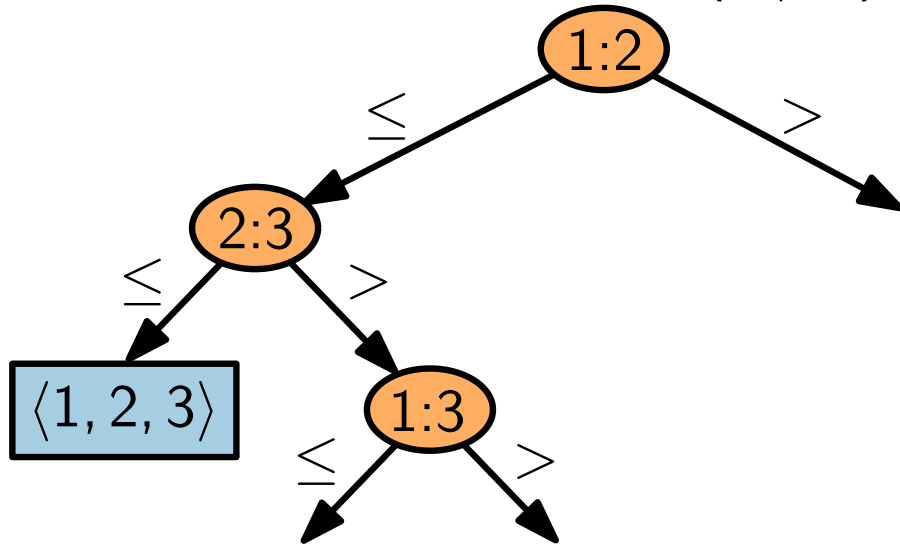


Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs

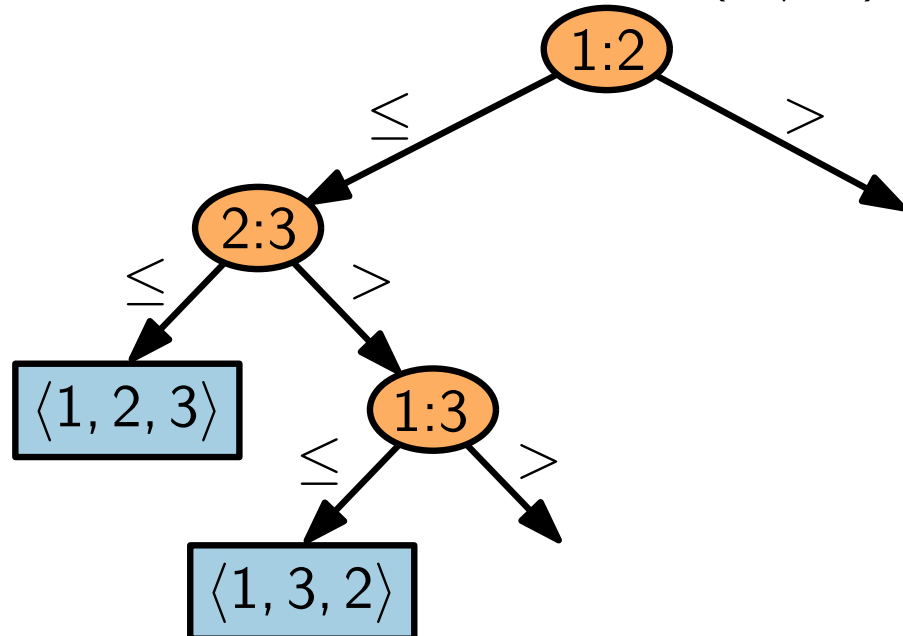


Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs



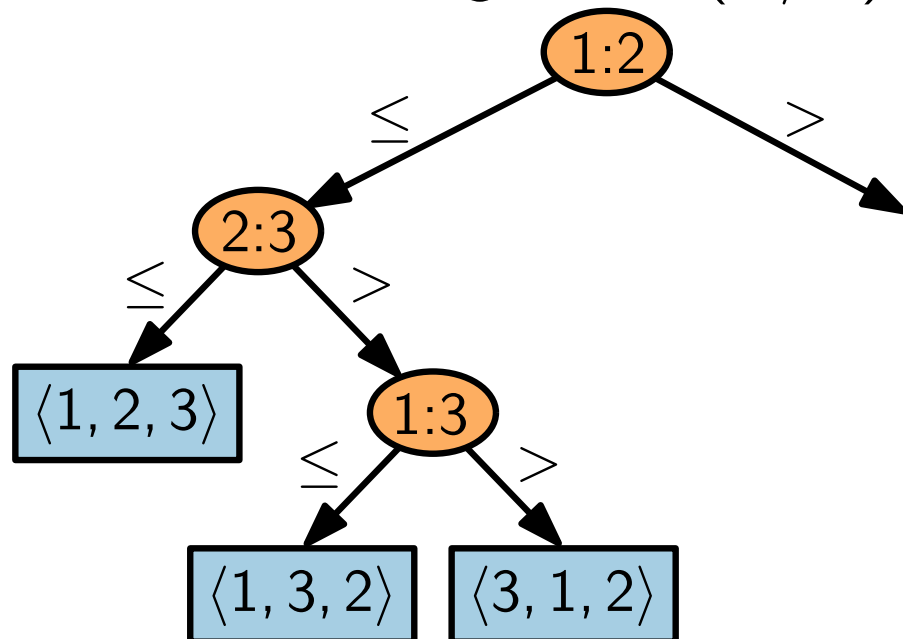
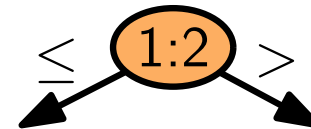
Entscheidungsbaum für INSERTIONSORT und $n = 3$

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs



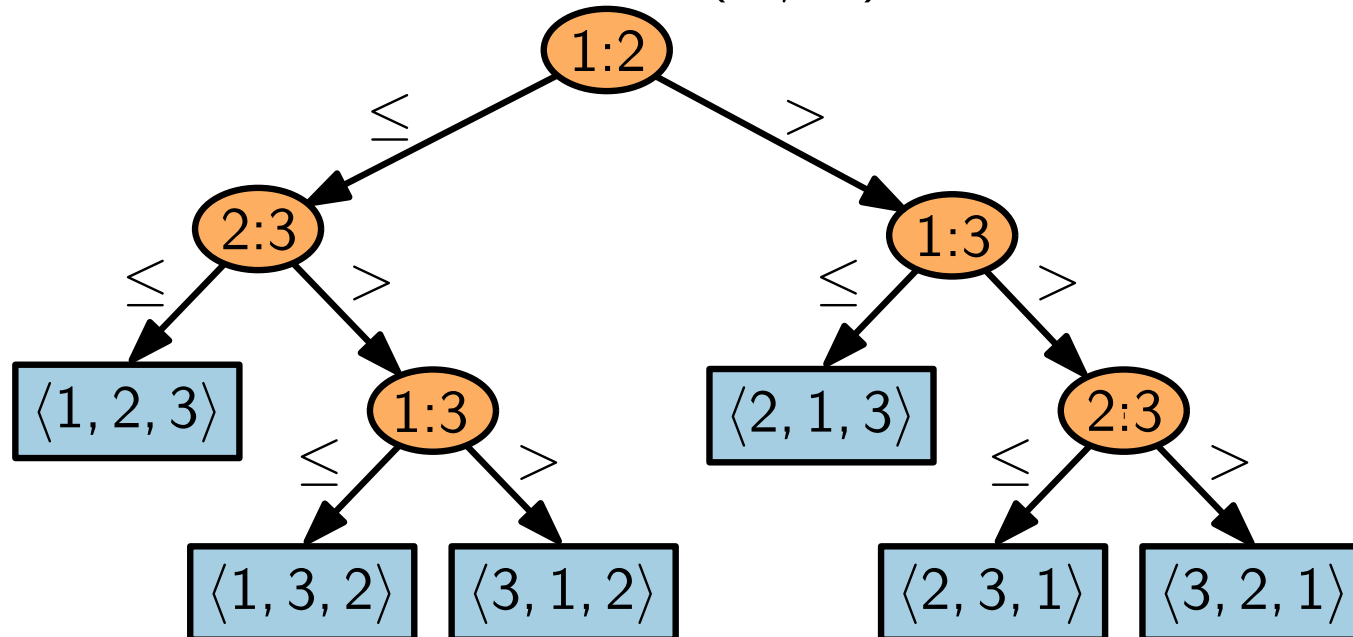
Entscheidungsbaum für INSERTIONSORT und $n = 3$

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs



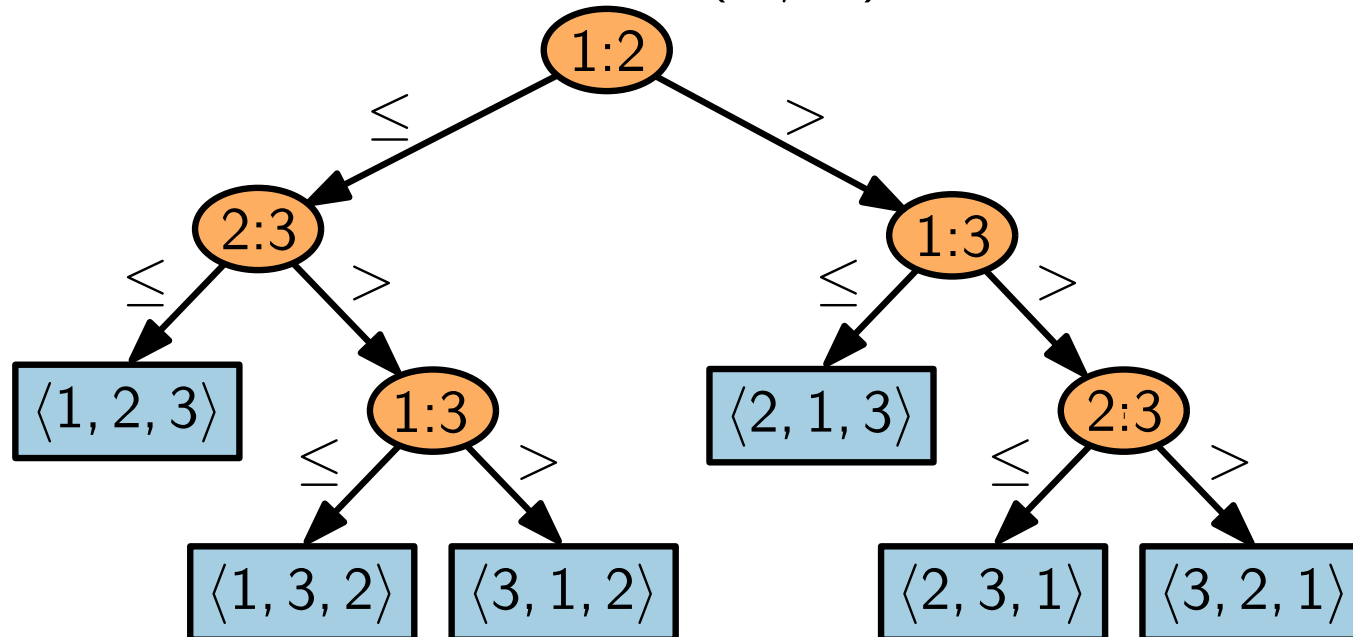
Entscheidungsbaum für INSERTIONSORT und $n = 3$

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs



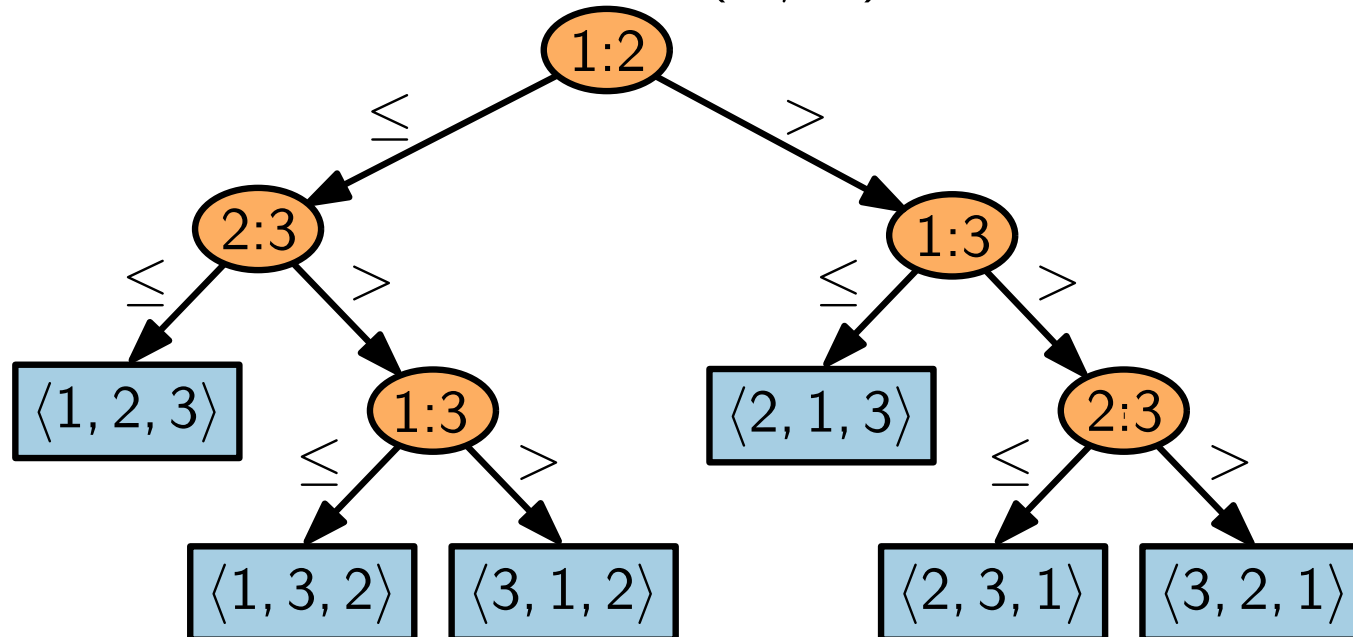
Anz. Vgl. im besten Fall

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortialg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortialgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs



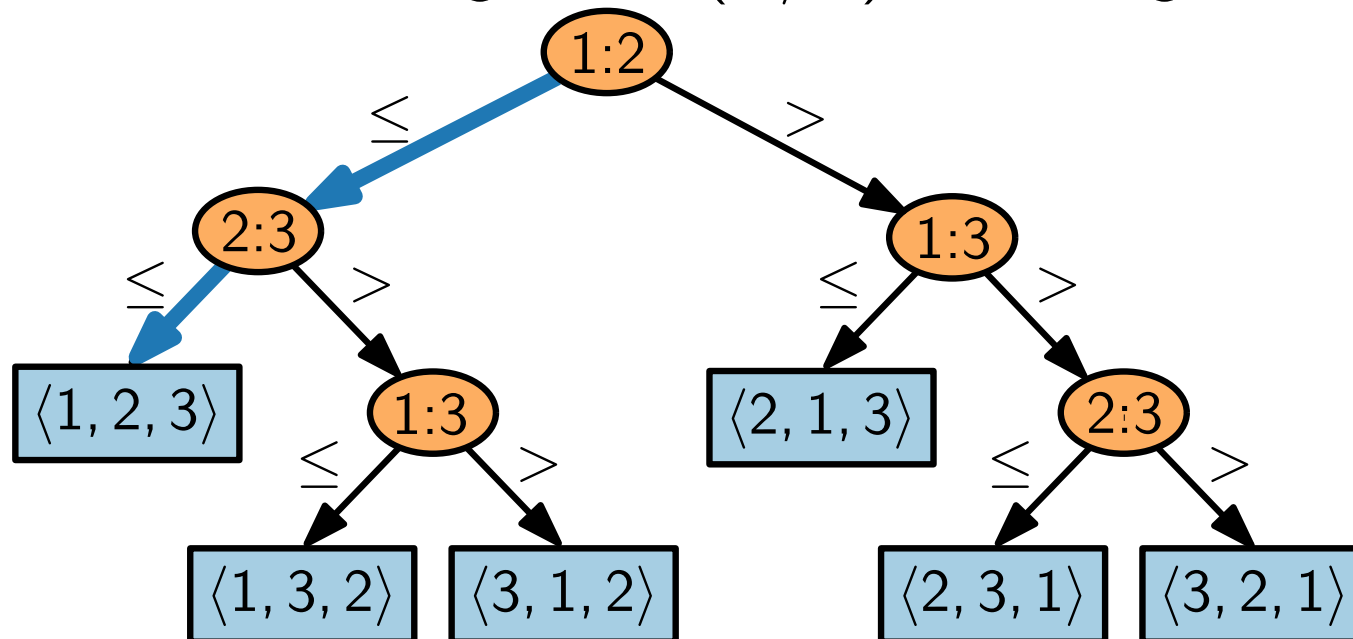
Anz. Vgl. im besten Fall
= Länge eines kürzesten
Wurzel-Blatt-Pfads

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortialg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs



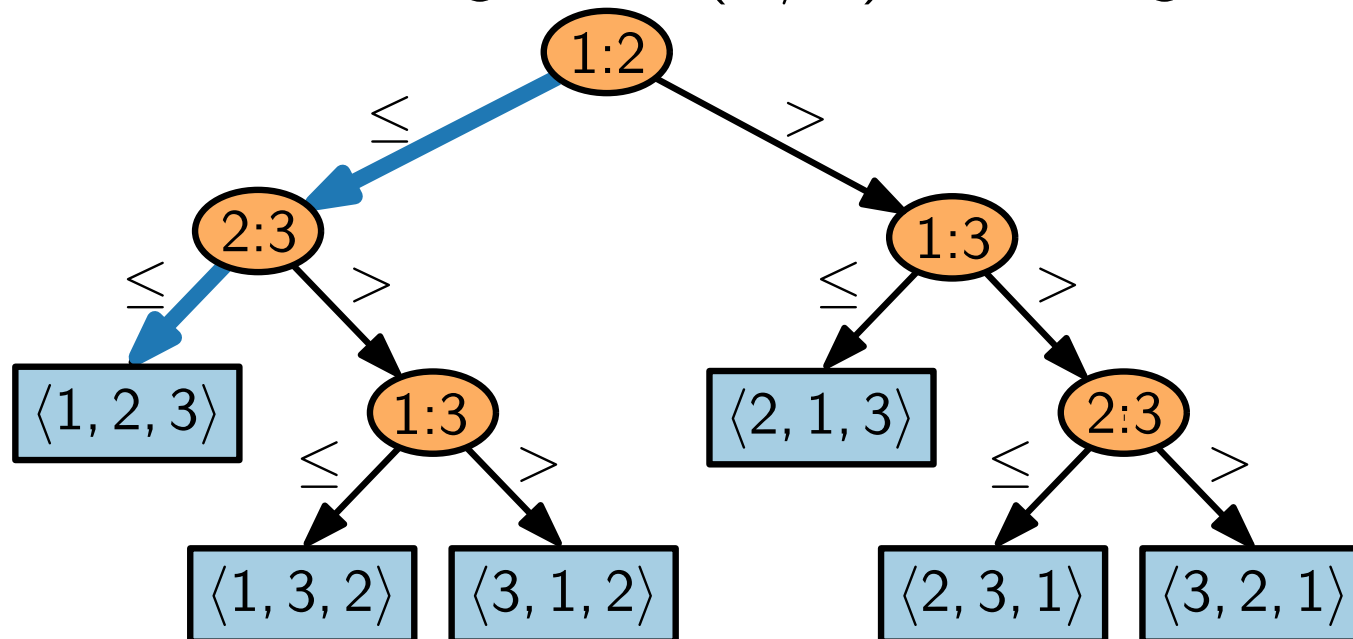
Anz. Vgl. im besten Fall
= Länge eines kürzesten
Wurzel-Blatt-Pfads

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortialg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortialgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs



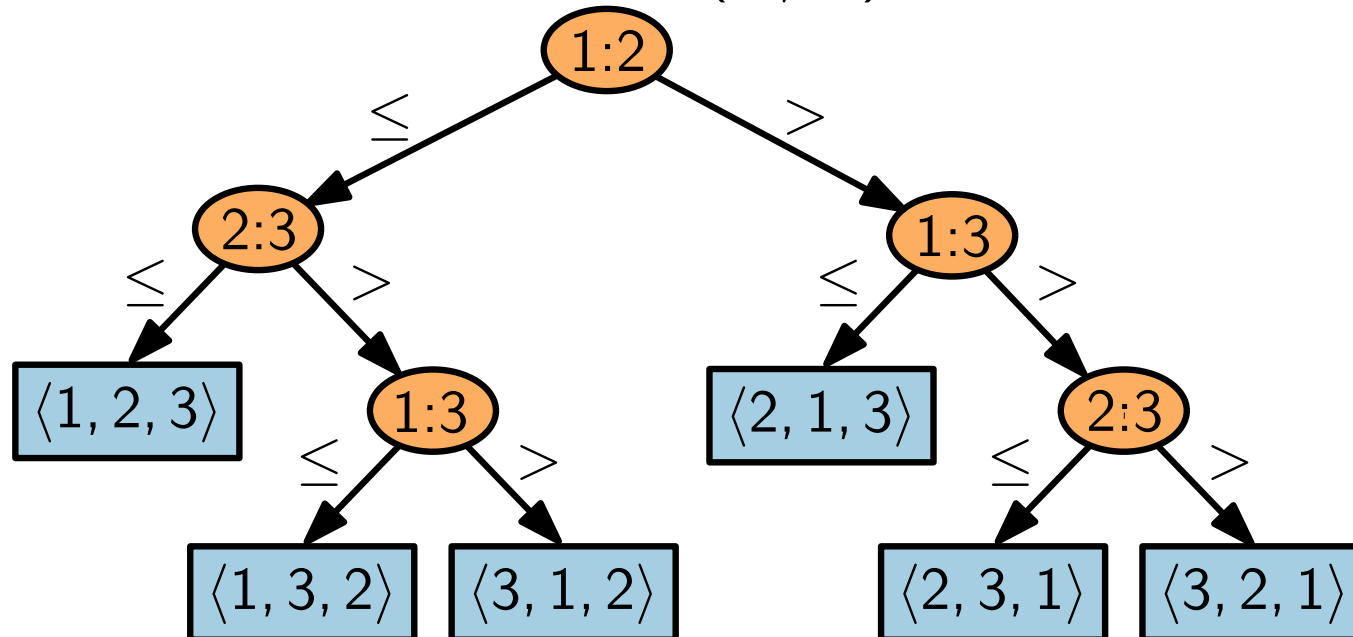
Anz. Vgl. im besten Fall
= Länge eines kürzesten
Wurzel-Blatt-Pfads
= hier 2

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs



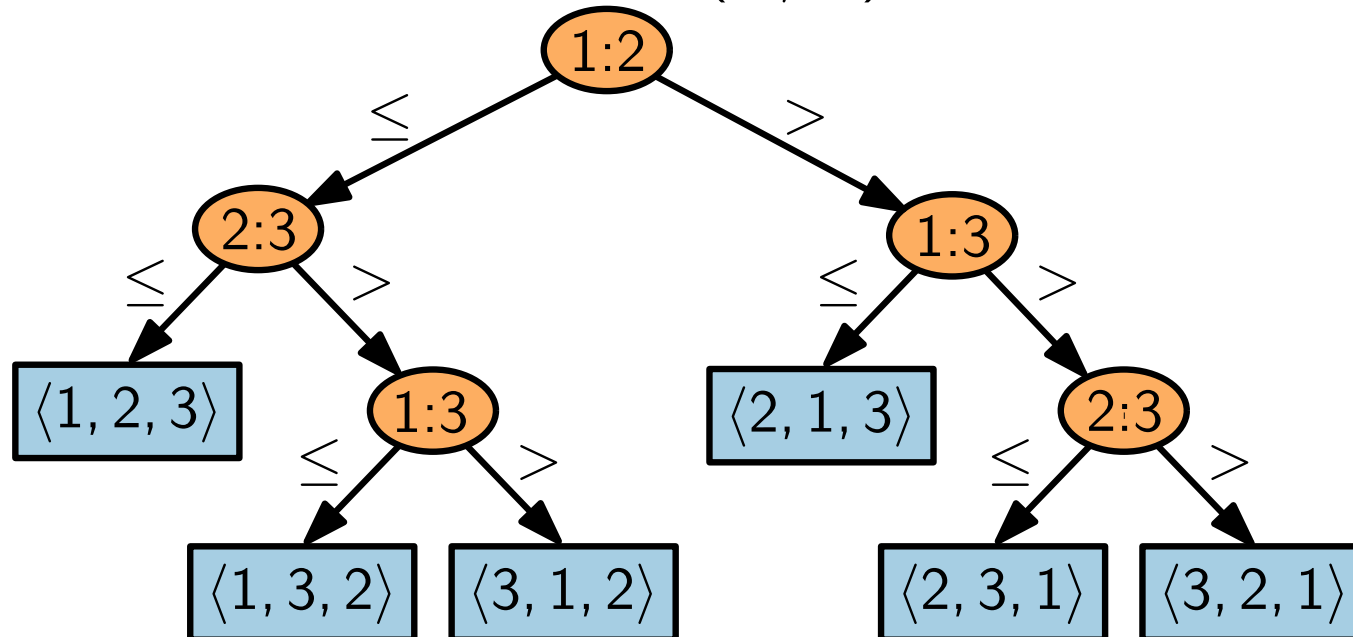
Anz. Vgl. im schlechtesten Fall

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs



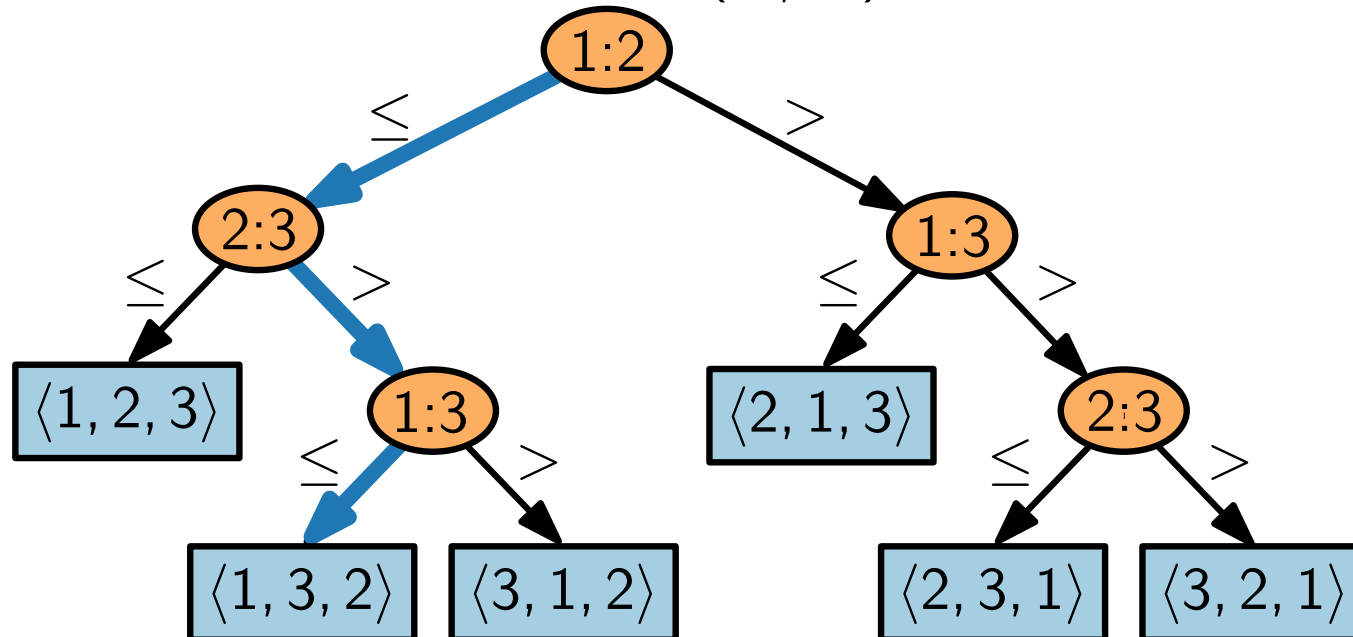
Anz. Vgl. im schlechtesten Fall
= Länge eines **längsten**
Wurzel-Blatt-Pfads

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs



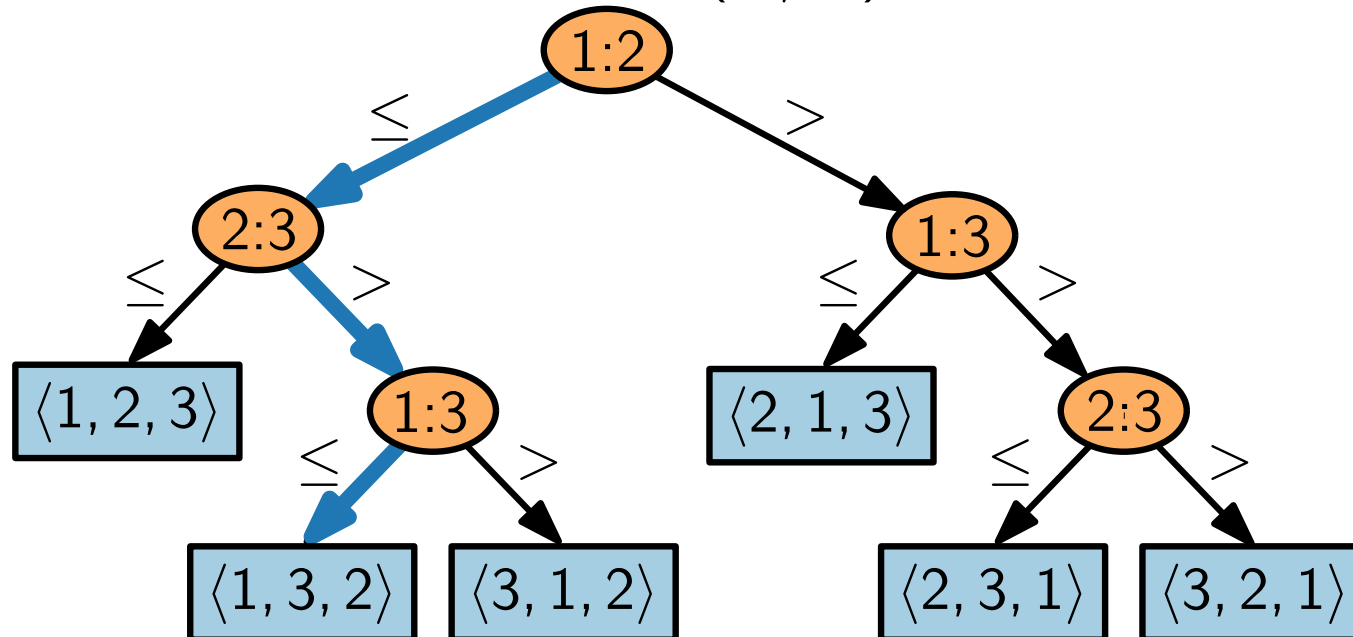
Anz. Vgl. im schlechtesten Fall
= Länge eines **längsten**
Wurzel-Blatt-Pfads

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortialg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs



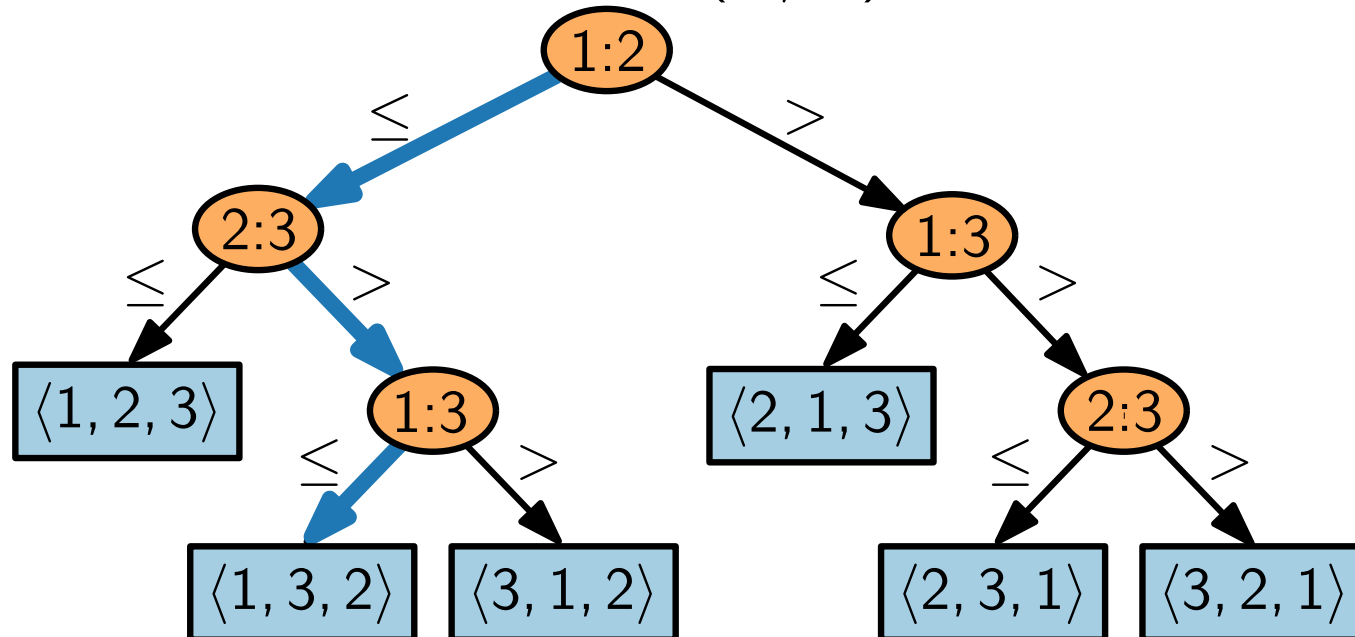
Anz. Vgl. im schlechtesten Fall
 = Länge eines **längsten**
 Wurzel-Blatt-Pfads
 =: Höhe des Baums

Sortieren durch Vergleichen

Eingabefolge $\langle a_1, a_2, \dots, a_n \rangle$ $\xrightarrow[\text{Schlüsselvergleiche}]{\text{Sortieralg.}}$ Ausgabe: sortierte Eingabe

Für festes n ist ein **vergleichsbasierter** Sortieralgorithmus charakterisiert durch seinen **Entscheidungsbaum**:

- innere Knoten = Vergleiche (o.B.d.A. immer $\leq$, z.B. „ $a_1 \leq a_2$?“)
- Blätter = sortierte Permutationen der Eingabe
- Kanten = Ergebnisse ($\leq / >$) eines Vergleichs



Anz. Vgl. im schlechtesten Fall
 = Länge eines **längsten**
 Wurzel-Blatt-Pfads
 =: Höhe des Baums
 = hier 3

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
- eine Zahl n von verschiedenen Objekten, die man sortieren soll,

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

$$\text{Anz. Blätter} = \text{Anz. Permutationen von } n \text{ Obj.} =$$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

$$\text{Anz. Blätter} = \text{Anz. Permutationen von } n \text{ Obj.} = n!$$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

Höhe Entscheidungsbaum $\geq$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

Höhe Entscheidungsbaum $\geq \log_2 n!$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

Höhe Entscheidungsbaum $\geq \log_2 n! = \log_2 \prod_{i=1}^n i$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\text{Höhe Entscheidungsbaum} \geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i$$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\begin{aligned} \text{Höhe Entscheidungsbaum} &\geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i \\ &\geq \int_1^n \log_2 x \, dx \end{aligned}$$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\begin{aligned} \text{Höhe Entscheidungsbaum} &\geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i \\ &\geq \int_1^n \log_2 x \, dx = \frac{1}{\ln 2} \int_1^n \ln x \, dx \end{aligned}$$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\begin{aligned} \text{Höhe Entscheidungsbaum} &\geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i \\ &\geq \int_1^n \log_2 x \, dx = \frac{1}{\ln 2} \int_1^n \ln x \, dx = \frac{1}{\ln 2} \int_1^n 1 \cdot \ln x \, dx \end{aligned}$$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\text{Höhe Entscheidungsbaum} \geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i$$

$$\geq \int_1^n \log_2 x \, dx = \frac{1}{\ln 2} \int_1^n \ln x \, dx = \frac{1}{\ln 2} \int_1^n 1 \cdot \ln x \, dx$$

=

Partielle Integration. $\int_a^b u'v = [uv]_a^b - \int_a^b uv'$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\text{Höhe Entscheidungsbaum} \geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i$$

$$\geq \int_1^n \log_2 x \, dx = \frac{1}{\ln 2} \int_1^n \ln x \, dx = \frac{1}{\ln 2} \int_1^n 1 \cdot \ln x \, dx$$

=

Partielle Integration. $\int_a^b u'v = [uv]_a^b - \int_a^b uv'$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\text{Höhe Entscheidungsbaum} \geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i$$

$$\geq \int_1^n \log_2 x \, dx = \frac{1}{\ln 2} \int_1^n \ln x \, dx = \frac{1}{\ln 2} \int_1^n 1 \cdot \ln x \, dx$$

=

Partielle Integration. $\int_a^b u' v = [uv]_a^b - \int_a^b uv' \, dx$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\text{Höhe Entscheidungsbaum} \geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i$$

$$\geq \int_1^n \log_2 x \, dx = \frac{1}{\ln 2} \int_1^n \ln x \, dx = \frac{1}{\ln 2} \int_1^n 1 \cdot \ln x \, dx$$

=

Partielle Integration. $\int_a^b u' v = [uv]_a^b - \int_a^b uv' \, dx$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\text{Höhe Entscheidungsbaum} \geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i$$

$$\geq \int_1^n \log_2 x \, dx = \frac{1}{\ln 2} \int_1^n \ln x \, dx = \frac{1}{\ln 2} \int_1^n 1 \cdot \ln x \, dx$$

$$= \frac{1}{\ln 2} \left(\left[\int_1^n 1 \cdot \ln x \, dx \right]_1^n - \int_1^n 1 \cdot \ln x \, dx \right)$$

Partielle Integration. $\int_a^b u'v = [uv]_a^b - \int_a^b uv' \, dx$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\text{Höhe Entscheidungsbaum} \geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i$$

$$\geq \int_1^n \log_2 x \, dx = \frac{1}{\ln 2} \int_1^n \ln x \, dx = \frac{1}{\ln 2} \int_1^n 1 \cdot \ln x \, dx$$

$$= \frac{1}{\ln 2} \left(\left[x \cdot \frac{1}{x} \right]_1^n - \int_1^n x \cdot \left(-\frac{1}{x^2} \right) dx \right)$$

Partielle Integration. $\int_a^b u'v = [uv]_a^b - \int_a^b uv'$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\text{Höhe Entscheidungsbaum} \geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i$$

$$\geq \int_1^n \log_2 x \, dx = \frac{1}{\ln 2} \int_1^n \ln x \, dx = \frac{1}{\ln 2} \int_1^n 1 \cdot \ln x \, dx$$

$$= \frac{1}{\ln 2} \left([x \cdot \ln x]_1^n - \int_1^n x \cdot \frac{1}{x} \, dx \right)$$

Partielle Integration. $\int_a^b u' v = [uv]_a^b - \int_a^b uv' \, dx$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\text{Höhe Entscheidungsbaum} \geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i$$

$$\geq \int_1^n \log_2 x \, dx = \frac{1}{\ln 2} \int_1^n \ln x \, dx = \frac{1}{\ln 2} \int_1^n 1 \cdot \ln x \, dx$$

$$= \frac{1}{\ln 2} \left(\left[x \cdot \ln x \right]_1^n - \int_1^n x \cdot \frac{1}{x} \, dx \right)$$

Partielle Integration. $\int_a^b u' v = [uv]_a^b - \int_a^b uv' \, dx$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\text{Höhe Entscheidungsbaum} \geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i$$

$$\geq \int_1^n \log_2 x \, dx = \frac{1}{\ln 2} \int_1^n \ln x \, dx = \frac{1}{\ln 2} \int_1^n 1 \cdot \ln x \, dx$$

$$= \frac{1}{\ln 2} \left(\left[x \cdot \ln x \right]_1^n - \int_1^n x \cdot \frac{1}{x} \, dx \right)$$

Partielle Integration. $\int_a^b u'v = [uv]_a^b - \int_a^b uv' \quad \underbrace{1}_{1}$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\text{Höhe Entscheidungsbaum} \geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i$$

$$\geq \int_1^n \log_2 x \, dx = \frac{1}{\ln 2} \int_1^n \ln x \, dx = \frac{1}{\ln 2} \int_1^n 1 \cdot \ln x \, dx$$

$$= \frac{1}{\ln 2} \left(\left[x \cdot \ln x \right]_1^n - \int_1^n x \cdot \frac{1}{x} \, dx \right) = \frac{(n \ln n - 0) - (n - 1)}{\ln 2}$$

Partielle Integration. $\int_a^b u' v = [uv]_a^b - \int_a^b uv' \quad \underbrace{\hspace{1.5cm}}_1$

Eine untere Schranke

Frage: Wie viele Vergleiche braucht **jeder** vergleichsbasierte Sortieralgorithmus im schlechtesten Fall um n **verschiedene** Objekte zu sortieren?

M.a.W. Gegeben

- ein beliebiger vergleichsbasierter Sortieralgorithmus,
 - eine Zahl n von verschiedenen Objekten, die man sortieren soll,
- welche Höhe hat der Entscheidungsbaum **mindestens**?

Beob.: Die Höhe ist eine Funktion der Blätteranzahl.

Anz. Blätter = Anz. Permutationen von n Obj. = $n!$

Höhe Binärbaum mit B Blättern $\geq \lceil \log_2 B \rceil$

$$\text{Höhe Entscheidungsbaum} \geq \log_2 n! = \log_2 \prod_{i=1}^n i = \sum_{i=1}^n \log_2 i$$

$$\geq \int_1^n \log_2 x \, dx = \frac{1}{\ln 2} \int_1^n \ln x \, dx = \frac{1}{\ln 2} \int_1^n 1 \cdot \ln x \, dx$$

$$= \frac{1}{\ln 2} \left(\left[x \cdot \ln x \right]_1^n - \int_1^n x \cdot \frac{1}{x} \, dx \right) = \frac{(n \ln n - 0) - (n-1)}{\ln 2} \in \Omega(n \log n)$$

Partielle Integration. $\int_a^b u' v = [uv]_a^b - \int_a^b uv' \quad \underbrace{\hspace{1cm}}_1$

Resultat

Satz. Jeder vergleichsbasierte Sortieralg. benötigt im schlechtesten Fall $\Omega(n \log n)$ Vergleiche um n Objekte zu sortieren.

Resultat

Satz. Jeder vergleichsbasierte Sortieralg. benötigt im schlechtesten Fall $\Omega(n \log n)$ Vergleiche um n Objekte zu sortieren.

Korollar. MERGESORT und HEAPSORT sind **asymptotisch worst-case optimale** vergleichsbasierte Sortieralgorithmen.

Wir durchbrechen die Schallmauer



[Ensign John Gay, U.S. Navy, Public domain, via Wikimedia Commons]

Wir durchbrechen die Schallmauer

■ SPAGHETTISORT



[JFVelasquez Floro, CC0, via Wikimedia Commons]



[Ensign John Gay, U.S. Navy, Public domain, via Wikimedia Commons]

Wir durchbrechen die Schallmauer

■ SPAGHETTISORT



[JFVelasquez Floro, CC0, via Wikimedia Commons]

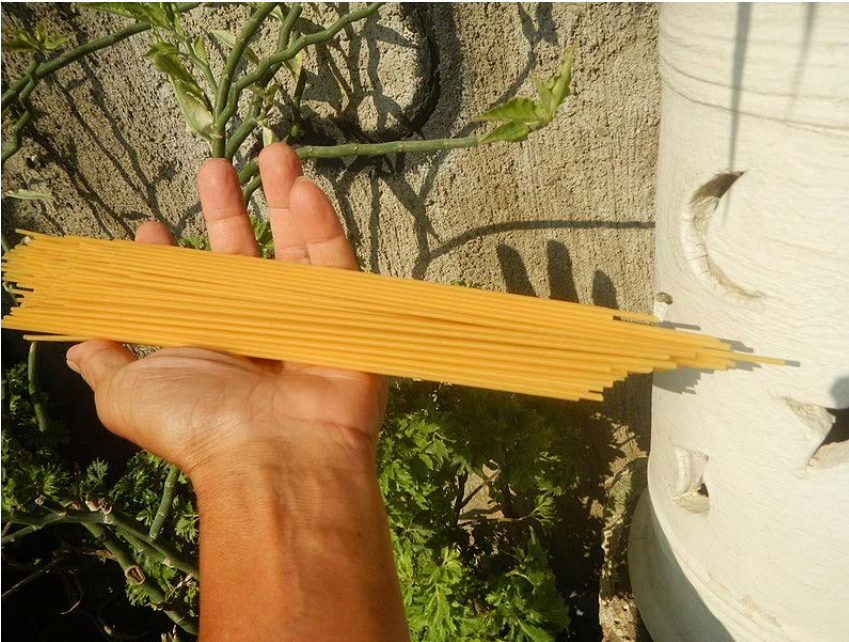


[Ensign John Gay, U.S. Navy, Public domain, via Wikimedia Commons]



Wir durchbrechen die Schallmauer

(■ SPAGHETTISORT sortiert Spaghetti nach Länge)



[JFVelasquez Floro, CC0, via Wikimedia Commons]

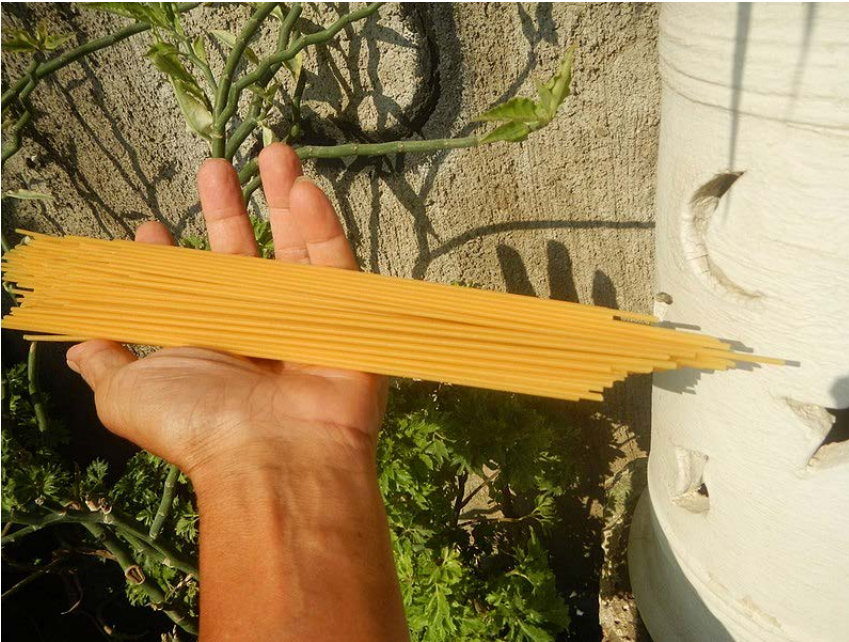


[Ensign John Gay, U.S. Navy, Public domain, via Wikimedia Commons]

Wir durchbrechen die Schallmauer

(■ SPAGHETTISORT sortiert Spaghetti nach Länge)

■ COUNTINGSORT



[JFVelasquez Floro, CC0, via Wikimedia Commons]

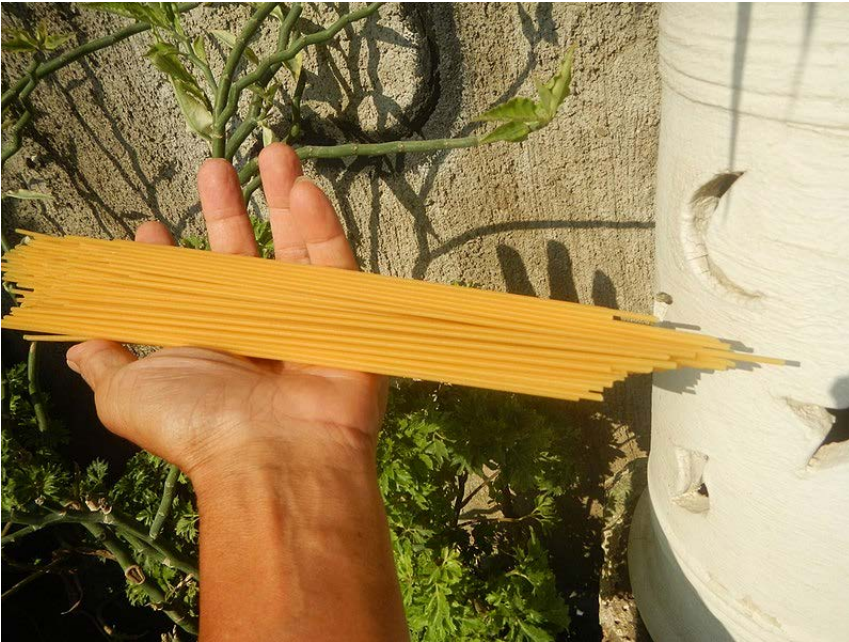


[Ensign John Gay, U.S. Navy, Public domain, via Wikimedia Commons]

Wir durchbrechen die Schallmauer

(■ SPAGHETTISORT sortiert Spaghetti nach Länge)

■ COUNTINGSORT sortiert Zahlen in $\{0, \dots, k\}$



[JFVelasquez Floro, CC0, via Wikimedia Commons]



[Ensign John Gay, U.S. Navy, Public domain, via Wikimedia Commons]

Wir durchbrechen die Schallmauer

(■ **SPAGHETTISORT** sortiert Spaghetti nach Länge)

■ **COUNTINGSORT** sortiert Zahlen in $\{0, \dots, k\}$

■ **RADIXSORT**



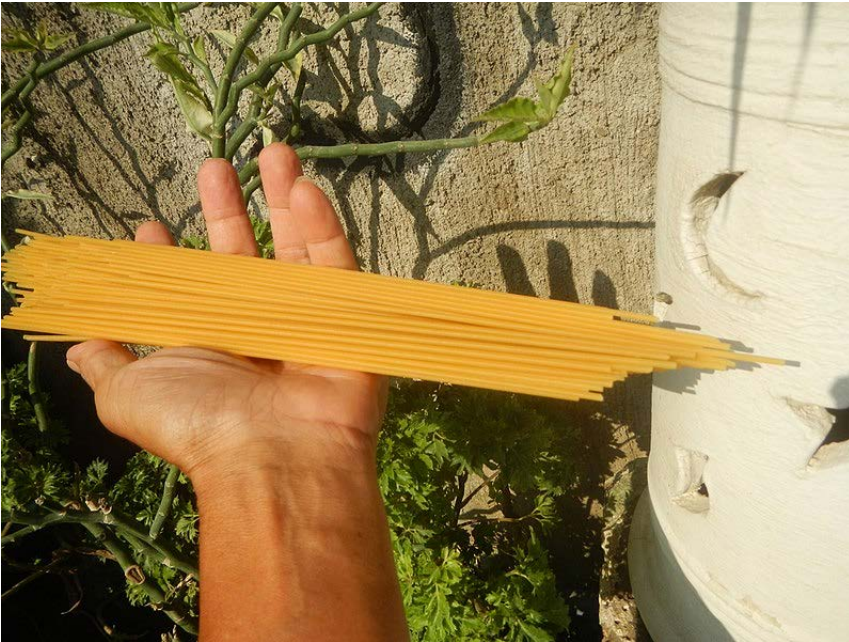
[JFVelasquez Floro, CC0, via Wikimedia Commons]



[Ensign John Gay, U.S. Navy, Public domain, via Wikimedia Commons]

Wir durchbrechen die Schallmauer

- (■ **SPAGHETTISORT** sortiert Spaghetti nach Länge)
- **COUNTINGSORT** sortiert Zahlen in $\{0, \dots, k\}$
- **RADIXSORT** sortiert s -stellige b -adische Zahlen



[JFVelasquez Floro, CC0, via Wikimedia Commons]



[Ensign John Gay, U.S. Navy, Public domain, via Wikimedia Commons]

Wir durchbrechen die Schallmauer

(■ **SPAGHETTISORT** sortiert Spaghetti nach Länge)

■ **COUNTINGSORT** sortiert Zahlen in $\{0, \dots, k\}$

■ **RADIXSORT** sortiert s -stellige b -adische Zahlen

■ **BUCKETSORT**



[JFVelasquez Floro, CC0, via Wikimedia Commons]



[Ensign John Gay, U.S. Navy, Public domain, via Wikimedia Commons]

Wir durchbrechen die Schallmauer

(■ **SPAGHETTISORT** sortiert Spaghetti nach Länge)

■ **COUNTINGSORT** sortiert Zahlen in $\{0, \dots, k\}$

■ **RADIXSORT** sortiert s -stellige b -adische Zahlen

■ **BUCKETSORT** sortiert gleichverteilte zufällige Zahlen



[JFVelasquez Floro, CC0, via Wikimedia Commons]



[Ensign John Gay, U.S. Navy, Public domain, via Wikimedia Commons]

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
2) benutze diese Information um x im Ausgabefeld
direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
2) benutze diese Information um x im Ausgabefeld
direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8
A	3	0	4	1	3	4	1	4

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A, zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8			0	1	2	3	4
A	3	0	4	1	3	4	1	4	⇒	C	0	0	0	0	0

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8		0	1	2	3	4	
A	3	0	4	1	3	4	1	4	⇒	C	0	0	0	1	0

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A, zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8			0	1	2	3	4
A	3	0	4	1	3	4	1	4	⇒	C	0	0	0	1	0

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8		0	1	2	3	4	
A	3	0	4	1	3	4	1	4	⇒	C	1	0	0	1	1

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8			0	1	2	3	4
A	3	0	4	1	3	4	1	4	⇒	C	1	1	0	1	1

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8		0	1	2	3	4	
A	3	0	4	1	3	4	1	4	⇒	C	1	1	0	1	1

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8			0	1	2	3	4
A	3	0	4	1	3	4	1	4	⇒	C	1	1	0	2	1

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8			0	1	2	3	4
A	3	0	4	1	3	4	1	4	⇒	C	1	1	0	2	1

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8			0	1	2	3	4
A	3	0	4	1	3	4	1	4	⇒	C	1	1	0	2	1

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8			0	1	2	3	4
A	3	0	4	1	3	4	1	4	⇒	C	1	1	0	2	1

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8			0	1	2	3	4
A	3	0	4	1	3	4	1	4	⇒	C	1	1	0	2	2

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8			0	1	2	3	4
A	3	0	4	1	3	4	1	4	⇒	C	1	1	0	2	2

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8			0	1	2	3	4
A	3	0	4	1	3	4	1	4	⇒	C	1	1	0	2	2

COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A, zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8		0	1	2	3	4	
A	3	0	4	1	3	4	1	4	⇒	C	1	2	0	2	3

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|---|---------------|-----|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | | 0 | 1 | 2 | 3 | 4 | |
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 | $\Rightarrow$ | C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)

COUNTINGSORT

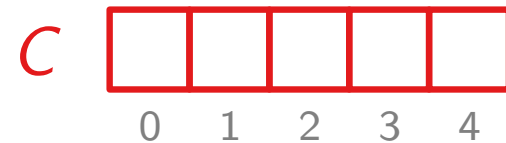
Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)



COUNTINGSORT

Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8
A	3	0	4	1	3	4	1	4

$\Rightarrow$

	0	1	2	3	4
C	1	2	0	2	3

↓

1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)

	0	1	2	3	4
C					

↓

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

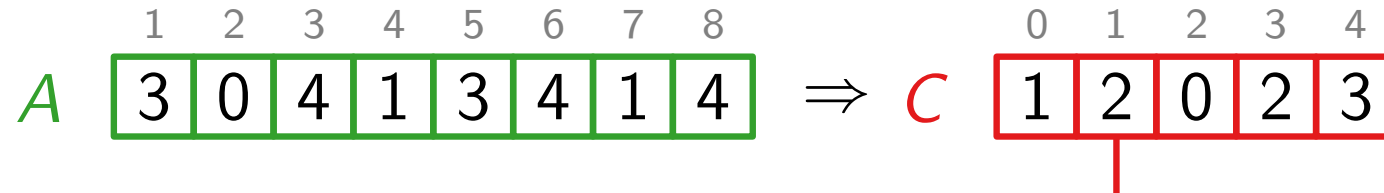
- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- ↓
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | | | | |

COUNTINGSORT

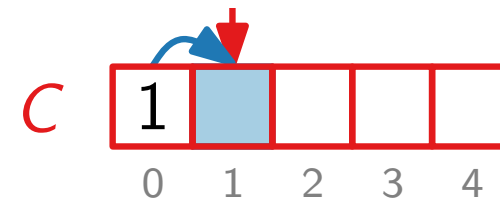
Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)

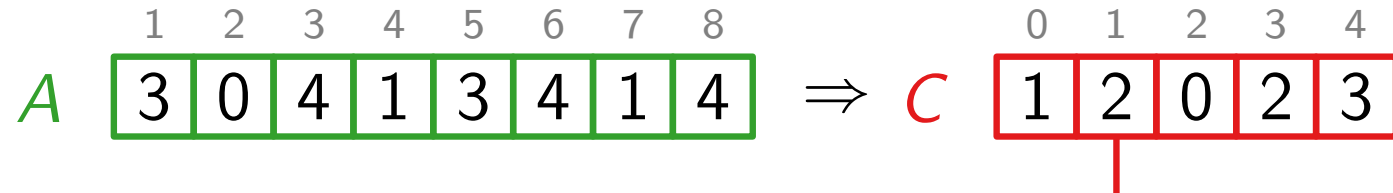


COUNTINGSORT

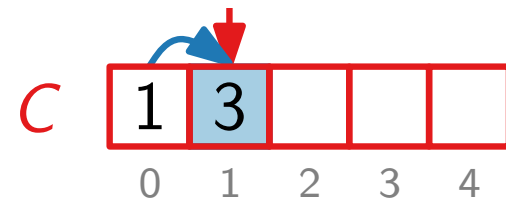
- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)



COUNTINGSORT

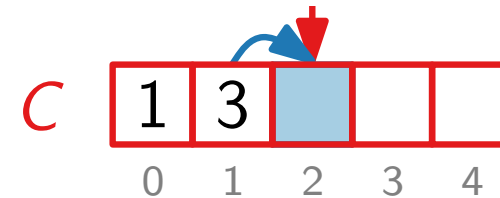
Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)



COUNTINGSORT

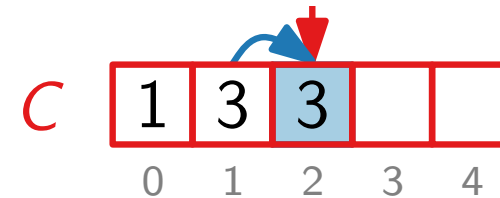
Idee: 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 2) benutze diese Information um x im Ausgabefeld
 direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)



COUNTINGSORT

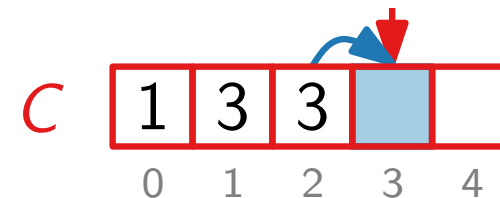
- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)

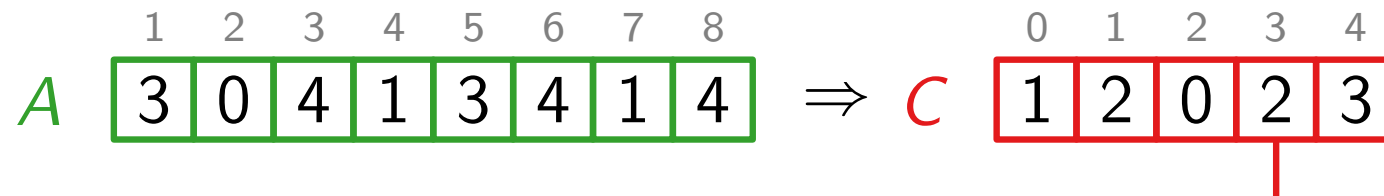


COUNTINGSORT

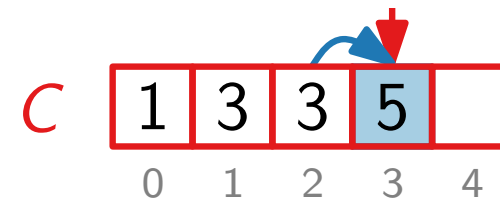
- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



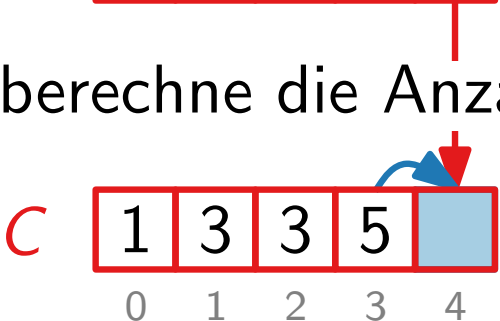
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)



COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 3 | 3 | 5 | |
- 

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 3 | 3 | 5 | 8 |
-

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8
A	3	0	4	1	3	4	1	4

 $\Rightarrow$

	0	1	2	3	4
C	1	2	0	2	3

- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)

	0	1	2	3	4
C	1	3	3	5	8

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|---|---------------|-----|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | | 0 | 1 | 2 | 3 | 4 | |
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 | $\Rightarrow$ | C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | | | | | |
|-----|---|---|---|---|---|
| C | 1 | 3 | 3 | 5 | 8 |
| | 0 | 1 | 2 | 3 | 4 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

COUNTINGSORT

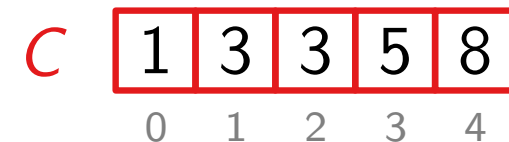
- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

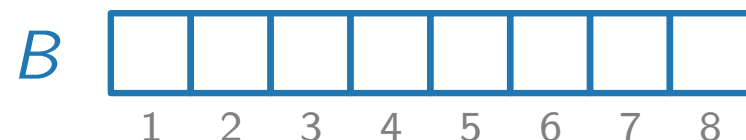
- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)



- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B



Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

B

1	2	3	4	5	6	7	8

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

B

1	2	3	4	5	6	7	8

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

B

1	2	3	4	5	6	7	8

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

B

1	2	3	4	5	6	7	8

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

C

1	3	3	5	8
0	1	2	3	4

B

							4
1	2	3	4	5	6	7	8

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Diagram illustrating the sequence C and its indices. The sequence is $C = [1, 3, 3, 5, 8]$ with indices $0, 1, 2, 3, 4$. The element 8 at index 4 is highlighted with a red box and a red arrow, indicating it is the current element being processed.

B

							4
1	2	3	4	5	6	7	8

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Diagram illustrating the array C with elements $1, 3, 3, 5, 7$. The element 7 at index 4 is highlighted in blue, and a red arrow points to it. The indices $0, 1, 2, 3, 4$ are shown below the array, with index 4 highlighted in an orange circle.

B

							4
1	2	3	4	5	6	7	8

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

B

							4
1	2	3	4	5	6	7	8

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

B

							4
1	2	3	4	5	6	7	8

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

B

							4
1	2	3	4	5	6	7	8

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 3 | 3 | 5 | 7 |
- 0 1 2 3 4
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | | 1 | | | | | 4 |
- 1 2 3 4 5 6 7 8

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 3 | 3 | 5 | 7 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | | 1 | | | | | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 3 | 5 | 7 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | | 1 | | | | | 4 |

COUNTINGSORT

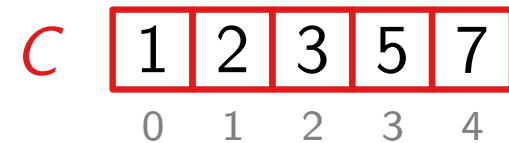
- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

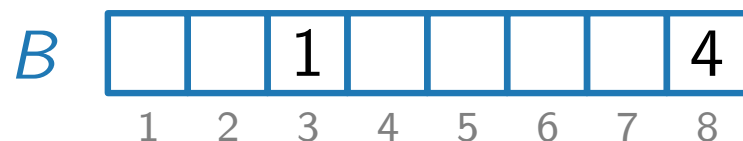
- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)



- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B



COUNTINGSORT

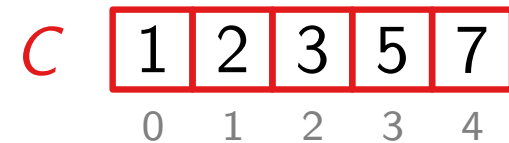
- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

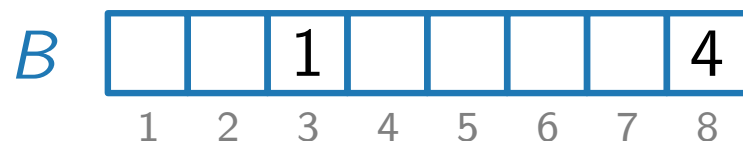
- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)



- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B



COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|---|---------------|-----|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | | 0 | 1 | 2 | 3 | 4 | |
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 | $\Rightarrow$ | C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | | | | | |
|-----|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 |
| C | 1 | 2 | 3 | 5 | 7 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | | | | | | | | |
|-----|---|---|---|---|---|---|---|---|
| B | | | 1 | | | | | 4 |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
-

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|---|---------------|-----|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | | 0 | 1 | 2 | 3 | 4 | |
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 | $\Rightarrow$ | C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | | | | | |
|-----|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 |
| C | 1 | 2 | 3 | 5 | 7 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | | | | | | | | |
|-----|---|---|---|---|---|---|---|---|
| B | | | 1 | | | | 4 | 4 |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
-

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 3 | 5 | 7 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | | 1 | | | | 4 | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 3 | 5 | 6 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | | 1 | | | | 4 | 4 |

COUNTINGSORT

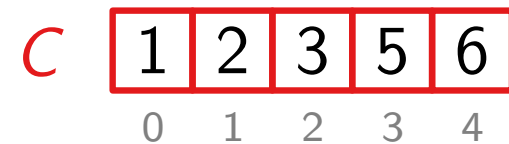
- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

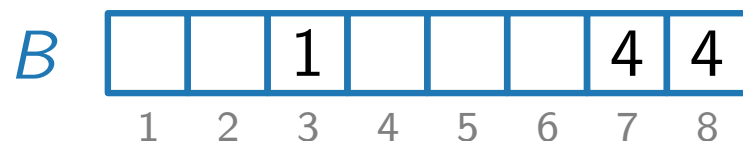
- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)



- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B



COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 3 | 5 | 6 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | | 1 | | | | 4 | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 3 | 5 | 6 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | | 1 | | 3 | | 4 | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 3 | 5 | 6 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | | 1 | | 3 | | 4 | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 3 | 4 | 6 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | | 1 | | 3 | | 4 | 4 |

COUNTINGSORT

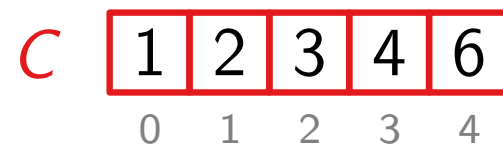
- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

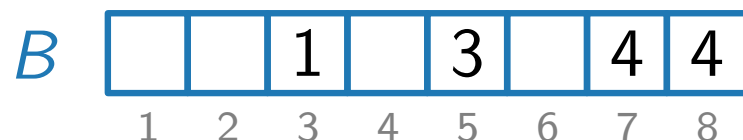
Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)



2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B



COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 3 | 4 | 6 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | | 1 | | 3 | | 4 | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 3 | 4 | 6 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | 1 | 1 | | 3 | | 4 | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|---|---------------|-----|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | | 0 | 1 | 2 | 3 | 4 | |
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 | $\Rightarrow$ | C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | | | | | |
|-----|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 |
| C | 1 | 2 | 3 | 4 | 6 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | | | | | | | | |
|-----|---|---|---|---|---|---|---|---|
| B | | 1 | 1 | | 3 | | 4 | 4 |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 1 | 3 | 4 | 6 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | 1 | 1 | | 3 | | 4 | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 1 | 3 | 4 | 6 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | 1 | 1 | | 3 | | 4 | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

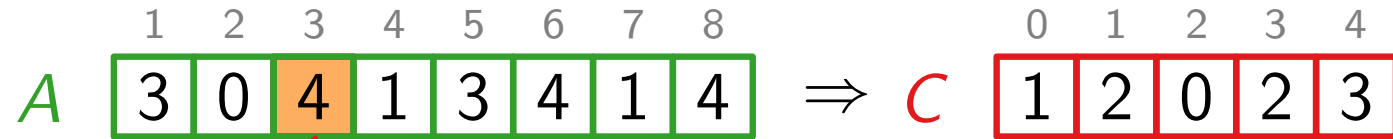
- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 1 | 3 | 4 | 6 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | 1 | 1 | | 3 | 4 | 4 | 4 |

COUNTINGSORT

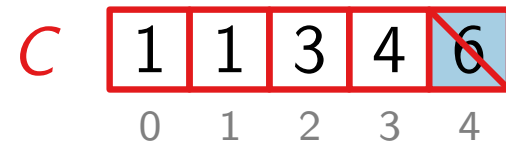
- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

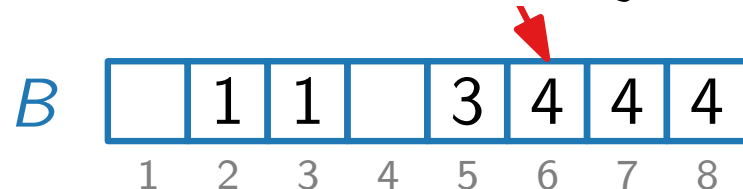
- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)



- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B



COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 1 | 3 | 4 | 5 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | 1 | 1 | | 3 | 4 | 4 | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 1 | 3 | 4 | 5 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | | 1 | 1 | | 3 | 4 | 4 | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 1 | 3 | 4 | 5 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | 0 | 1 | 1 | | 3 | 4 | 4 | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|--------------|---|---|---|---|
| C | 1 | 1 | 3 | 4 | 5 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | 0 | 1 | 1 | | 3 | 4 | 4 | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|---|---------------|-----|---|---|---|---|---|
| A | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | $\Rightarrow$ | C | 0 | 1 | 2 | 3 | 4 |
| | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 | | | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | | | | | |
|-----|---|---|---|---|---|
| C | 0 | 1 | 2 | 3 | 4 |
| | 0 | 1 | 3 | 4 | 5 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | | | | | | | | |
|-----|---|---|---|---|---|---|---|---|
| B | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| | 0 | 1 | 1 | | 3 | 4 | 4 | 4 |

COUNTINGSORT

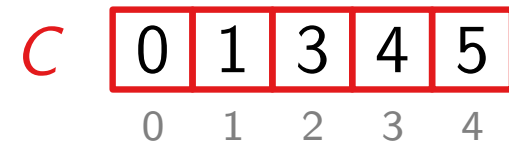
- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

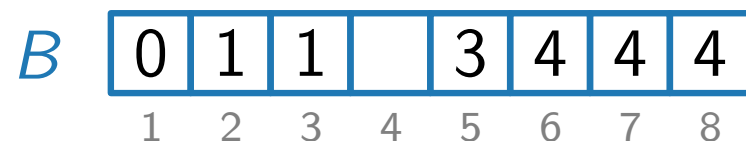
- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)



- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B



COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 0 | 1 | 3 | 4 | 5 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | 0 | 1 | 1 | | 3 | 4 | 4 | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 |
- $\Rightarrow$
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | 0 | 1 | 2 | 3 | 4 |
|-----|---|---|---|---|---|
| C | 0 | 1 | 3 | 4 | 5 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----|---|---|---|---|---|---|---|---|
| B | 0 | 1 | 1 | 3 | 3 | 4 | 4 | 4 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
- | | | | | | | | | | | | | | | | |
|-----|---|---|---|---|---|---|---|---|---------------|-----|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | | 0 | 1 | 2 | 3 | 4 | |
| A | 3 | 0 | 4 | 1 | 3 | 4 | 1 | 4 | $\Rightarrow$ | C | 1 | 2 | 0 | 2 | 3 |
- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
- | | | | | | |
|-----|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 |
| C | 0 | 1 | 3 | 4 | 5 |
| | 0 | 1 | 2 | 3 | 4 |
- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B
- | | | | | | | | | |
|-----|---|---|---|---|---|---|---|---|
| B | 0 | 1 | 1 | 3 | 3 | 4 | 4 | 4 |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8
A	3	0	4	1	3	4	1	4

 $\Rightarrow$

	0	1	2	3	4
C	1	2	0	2	3

- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)

	0	1	2	3	4
C	0	1	3	3	5

- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

	1	2	3	4	5	6	7	8
B	0	1	1	3	3	4	4	4

COUNTINGSORT

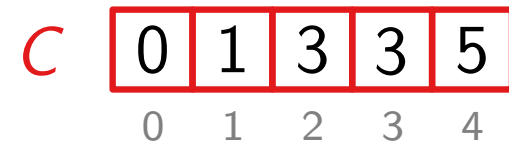
- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

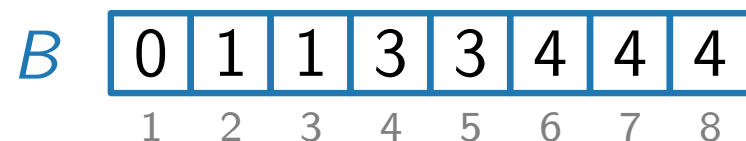
- Beispiel:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



- 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)



- 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B



COUNTINGSORT

- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

Beispiel:

1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)

	1	2	3	4	5	6	7	8
A	3	0	4	1	3	4	1	4

$\Rightarrow$

	0	1	2	3	4
C	1	2	0	2	3

1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)

	0	1	2	3	4
C	0	1	3	3	5

2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

	1	2	3	4	5	6	7	8
B	0	1	1	3	3	4	4	4

COUNTINGSORT

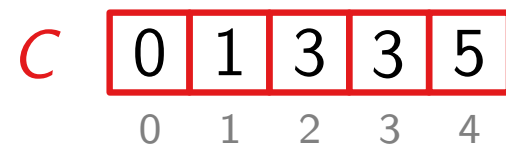
- Idee:**
- 1) für jedes x in der Eingabe: **zähle** die Anzahl der Zahlen $\leq x$
 - 2) benutze diese Information um x im Ausgabefeld direkt an die richtige Position zu **schreiben**

Variablen: A Eingabefeld | C Rechenfeld
 B Ausgabefeld | k begrenzt das **Universum:** $\{0, \dots, k\}$

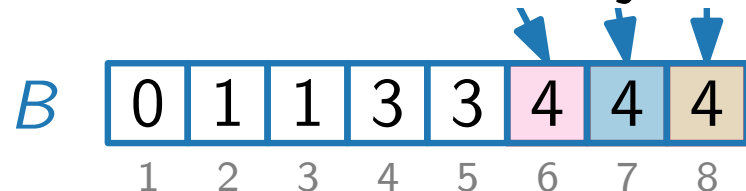
Beispiel: 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)



1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)



2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B



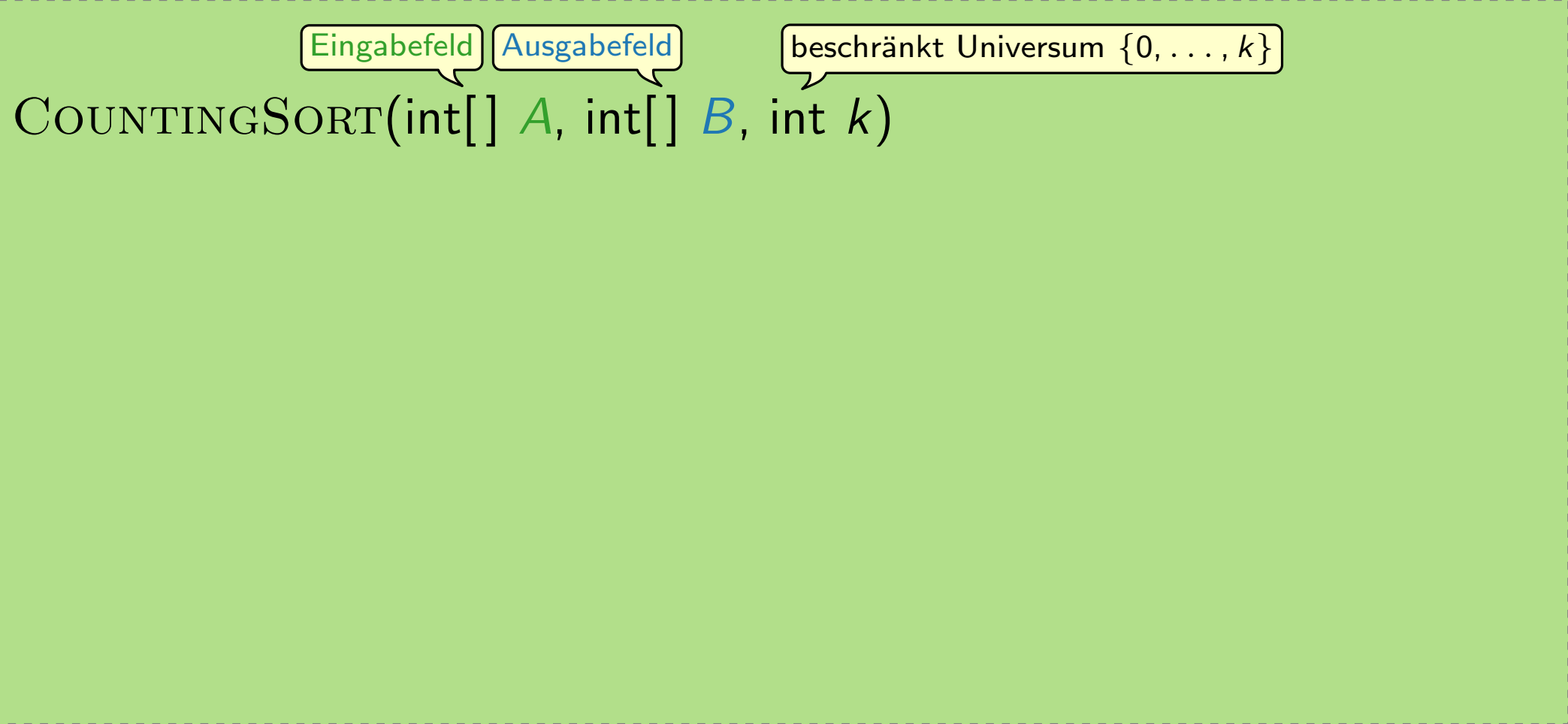
COUNTINGSORT ist **stabil!**

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

COUNTINGSORT

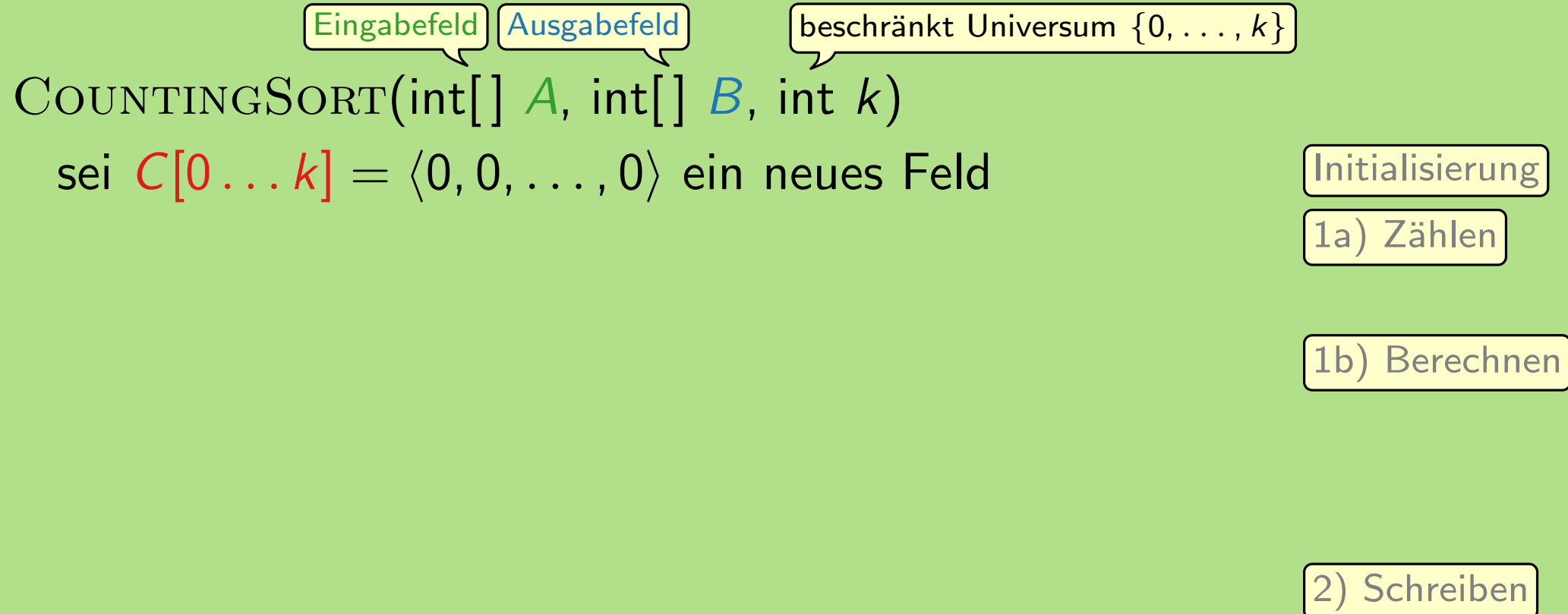
- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

The diagram shows the function signature 'COUNTINGSORT(int[] A, int[] B, int k)' on a light green background. Above the parameter 'A' is a green callout box labeled 'Eingabefeld'. Above the parameter 'B' is a blue callout box labeled 'Ausgabefeld'. Above the parameter 'k' is a yellow callout box labeled 'beschränkt Universum {0, ..., k}'.

```
COUNTINGSORT(int[] A, int[] B, int k)
```

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B



COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld Ausgabefeld beschränkt Universum $\{0, \dots, k\}$
 COUNTINGSORT(int[] A , int[] B , int k)
 sei $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$ ein neues Feld Initialisierung
for $j = 1$ **to** $A.length$ **do** 1a) Zählen
 // $C[i]$ enthält jetzt die Anzahl der Elemente gleich i in A 1b) Berechnen
2) Schreiben

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld

Ausgabefeld

beschränkt Universum $\{0, \dots, k\}$

```

COUNTINGSORT(int[]  $A$ , int[]  $B$ , int  $k$ )
    sei  $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$  ein neues Feld
    for  $j = 1$  to  $A.length$  do
        //  $C[i]$  enthält jetzt die Anzahl der Elemente gleich  $i$  in  $A$ 
    for  $i = 1$  to  $k$  do
        //  $C[i]$  enthält jetzt die Anzahl der Elemente  $\leq i$  in  $A$ 

```

Initialisierung

1a) Zählen

1b) Berechnen

2) Schreiben

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld
Ausgabefeld
beschränkt Universum $\{0, \dots, k\}$

```

COUNTSORT(int[]  $A$ , int[]  $B$ , int  $k$ )
    sei  $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$  ein neues Feld
    for  $j = 1$  to  $A.length$  do
        //  $C[i]$  enthält jetzt die Anzahl der Elemente gleich  $i$  in  $A$ 
    for  $i = 1$  to  $k$  do
        //  $C[i]$  enthält jetzt die Anzahl der Elemente  $\leq i$  in  $A$ 
    for  $j = A.length$  downto 1 do
        [ ]
  
```

Initialisierung

1a) Zählen

1b) Berechnen

2) Schreiben

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld Ausgabefeld beschränkt Universum $\{0, \dots, k\}$
 COUNTINGSORT(int[] A , int[] B , int k)

sei $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$ ein neues Feld

for $j = 1$ **to** $A.length$ **do** []

// $C[i]$ enthält jetzt die Anzahl der Elemente gleich i in A

for $i = 1$ **to** k **do** []

// $C[i]$ enthält jetzt die Anzahl der Elemente $\leq i$ in A

for $j = A.length$ **downto** 1 **do**

[]

Aufgabe.

Füllen Sie die Felder mit Code, der obige Idee umsetzt!

1a) Zählen

1b) Berechnen

2) Schreiben

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld Ausgabefeld beschränkt Universum $\{0, \dots, k\}$
 COUNTINGSORT(int[] A , int[] B , int k)
 sei $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$ ein neues Feld
for $j = 1$ **to** $A.length$ **do** $C[A[j]] = C[A[j]] + 1$
 // $C[i]$ enthält jetzt die Anzahl der Elemente gleich i in A
for $i = 1$ **to** k **do** []
 // $C[i]$ enthält jetzt die Anzahl der Elemente $\leq i$ in A
for $j = A.length$ **downto** 1 **do**
 | []

Aufgabe.

Füllen Sie die Felder mit Code, der obige Idee umsetzt!

1a) Zählen

1b) Berechnen

2) Schreiben

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld Ausgabefeld beschränkt Universum $\{0, \dots, k\}$
 COUNTINGSORT(int[] A , int[] B , int k)

sei $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$ ein neues Feld

for $j = 1$ **to** $A.length$ **do** $C[A[j]] = C[A[j]] + 1$

// $C[i]$ enthält jetzt die Anzahl der Elemente gleich i in A

for $i = 1$ **to** k **do** $C[i] = C[i] + C[i - 1]$

// $C[i]$ enthält jetzt die Anzahl der Elemente $\leq i$ in A

for $j = A.length$ **downto** 1 **do**

Aufgabe.

Füllen Sie die Felder mit Code, der obige Idee umsetzt!

1a) Zählen

1b) Berechnen

2) Schreiben

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld Ausgabefeld beschränkt Universum $\{0, \dots, k\}$
 COUNTINGSORT(int[] A , int[] B , int k)
 sei $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$ ein neues Feld
for $j = 1$ **to** $A.length$ **do** $C[A[j]] = C[A[j]] + 1$
 // $C[i]$ enthält jetzt die Anzahl der Elemente gleich i in A
for $i = 1$ **to** k **do** $C[i] = C[i] + C[i - 1]$
 // $C[i]$ enthält jetzt die Anzahl der Elemente $\leq i$ in A
for $j = A.length$ **downto** 1 **do**
 $B[C[A[j]]] = A[j]$

Aufgabe.

Füllen Sie die Felder mit Code, der obige Idee umsetzt!

1a) Zählen

1b) Berechnen

2) Schreiben

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld
Ausgabefeld
beschränkt Universum $\{0, \dots, k\}$

```

COUNTSORT(int[]  $A$ , int[]  $B$ , int  $k$ )
    sei  $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$  ein neues Feld
    for  $j = 1$  to  $A.length$  do  $C[A[j]] = C[A[j]] + 1$ 
    //  $C[i]$  enthält jetzt die Anzahl der Elemente gleich  $i$  in  $A$ 
    for  $i = 1$  to  $k$  do  $C[i] = C[i] + C[i - 1]$ 
    //  $C[i]$  enthält jetzt die Anzahl der Elemente  $\leq i$  in  $A$ 
    for  $j = A.length$  downto 1 do
         $B[C[A[j]]] = A[j]$ 
         $C[A[j]] = C[A[j]] - 1$ 
  
```

Aufgabe.

Füllen Sie die Felder mit Code, der obige Idee umsetzt!

1a) Zählen

1b) Berechnen

2) Schreiben

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld Ausgabefeld beschränkt Universum $\{0, \dots, k\}$
 COUNTINGSORT(int[] A , int[] B , int k)
 sei $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$ ein neues Feld
for $j = 1$ **to** $A.length$ **do** $C[A[j]] = C[A[j]] + 1$
 // $C[i]$ enthält jetzt die Anzahl der Elemente gleich i in A
for $i = 1$ **to** k **do** $C[i] = C[i] + C[i - 1]$
 // $C[i]$ enthält jetzt die Anzahl der Elemente $\leq i$ in A
for $j = A.length$ **downto** 1 **do**
 $B[C[A[j]]] = A[j]$
 $C[A[j]] = C[A[j]] - 1$

Aufgabe.

Füllen Sie die Felder mit Code, der obige Idee umsetzt!

1a) Zählen

1b) Berechnen

Demo.

<https://algo.uni-trier.de/demos/sort.html>

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld Ausgabefeld beschränkt Universum $\{0, \dots, k\}$
 COUNTINGSORT(int[] A , int[] B , int k)

sei $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$ ein neues Feld

for $j = 1$ **to** $A.length$ **do** $C[A[j]] = C[A[j]] + 1$

// $C[i]$ enthält jetzt die Anzahl der Elemente gleich i in A

for $i = 1$ **to** k **do** $C[i] = C[i] + C[i - 1]$

// $C[i]$ enthält jetzt die Anzahl der Elemente $\leq i$ in A

for $j = A.length$ **downto** 1 **do**

$B[C[A[j]]] = A[j]$
 $C[A[j]] = C[A[j]] - 1$

Initialisierung

1a) Zählen

1b) Berechnen

2) Schreiben

Laufzeit:

$\mathcal{O}(\quad)$

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld Ausgabefeld beschränkt Universum $\{0, \dots, k\}$
 COUNTINGSORT(int[] A , int[] B , int k)
 sei $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$ ein neues Feld Initialisierung
for $j = 1$ **to** $A.length$ **do** $C[A[j]] = C[A[j]] + 1$ 1a) Zählen
 // $C[i]$ enthält jetzt die Anzahl der Elemente gleich i in A
for $i = 1$ **to** k **do** $C[i] = C[i] + C[i - 1]$ 1b) Berechnen
 // $C[i]$ enthält jetzt die Anzahl der Elemente $\leq i$ in A
for $j = A.length$ **downto** 1 **do**
 $B[C[A[j]]] = A[j]$
 $C[A[j]] = C[A[j]] - 1$ 2) Schreiben

Laufzeit:
 $\mathcal{O}(\quad)$

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld Ausgabefeld beschränkt Universum $\{0, \dots, k\}$
 COUNTINGSORT(int[] A , int[] B , int k)

sei $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$ ein neues Feld

for $j = 1$ **to** $A.length$ **do** $C[A[j]] = C[A[j]] + 1$

// $C[i]$ enthält jetzt die Anzahl der Elemente gleich i in A

for $i = 1$ **to** k **do** $C[i] = C[i] + C[i - 1]$

// $C[i]$ enthält jetzt die Anzahl der Elemente $\leq i$ in A

for $j = A.length$ **downto** 1 **do**

$B[C[A[j]]] = A[j]$
 $C[A[j]] = C[A[j]] - 1$

Initialisierung

1a) Zählen

1b) Berechnen

2) Schreiben

Laufzeit:

$\mathcal{O}(k)$

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld
Ausgabefeld
beschränkt Universum $\{0, \dots, k\}$

```

COUNTSORT(int[]  $A$ , int[]  $B$ , int  $k$ )
    sei  $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$  ein neues Feld
    for  $j = 1$  to  $A.length$  do  $C[A[j]] = C[A[j]] + 1$ 
    //  $C[i]$  enthält jetzt die Anzahl der Elemente gleich  $i$  in  $A$ 
    for  $i = 1$  to  $k$  do  $C[i] = C[i] + C[i - 1]$ 
    //  $C[i]$  enthält jetzt die Anzahl der Elemente  $\leq i$  in  $A$ 
    for  $j = A.length$  downto 1 do
         $B[C[A[j]]] = A[j]$ 
         $C[A[j]] = C[A[j]] - 1$ 
  
```

Initialisierung

1a) Zählen

1b) Berechnen

2) Schreiben

Laufzeit:
 $\mathcal{O}(k)$

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld Ausgabefeld beschränkt Universum $\{0, \dots, k\}$
 COUNTINGSORT(int[] A , int[] B , int k)

sei $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$ ein neues Feld

for $j = 1$ **to** $A.length$ **do** $C[A[j]] = C[A[j]] + 1$

// $C[i]$ enthält jetzt die Anzahl der Elemente gleich i in A

for $i = 1$ **to** k **do** $C[i] = C[i] + C[i - 1]$

// $C[i]$ enthält jetzt die Anzahl der Elemente $\leq i$ in A

for $j = A.length$ **downto** 1 **do**

$B[C[A[j]]] = A[j]$
 $C[A[j]] = C[A[j]] - 1$

Initialisierung

1a) Zählen

1b) Berechnen

2) Schreiben

Laufzeit:

$\mathcal{O}(n + k)$

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld Ausgabefeld beschränkt Universum $\{0, \dots, k\}$
 COUNTINGSORT(int[] A , int[] B , int k)

sei $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$ ein neues Feld

for $j = 1$ **to** $A.length$ **do** $C[A[j]] = C[A[j]] + 1$

// $C[i]$ enthält jetzt die Anzahl der Elemente gleich i in A

for $i = 1$ **to** k **do** $C[i] = C[i] + C[i - 1]$

// $C[i]$ enthält jetzt die Anzahl der Elemente $\leq i$ in A

for $j = A.length$ **downto** 1 **do**

$B[C[A[j]]] = A[j]$
 $C[A[j]] = C[A[j]] - 1$

Initialisierung

1a) Zählen

1b) Berechnen

2) Schreiben

Laufzeit:

$\mathcal{O}(n + k)$

COUNTINGSORT

- Plan:**
- 1a) **Zählen:** Für jedes x in A , zähle die Anzahl der Zahlen gleich x (in C)
 - 1b) **Berechnen:** Für jedes x in A , berechne die Anzahl der Zahlen $\leq x$ (in C)
 - 2) **Schreiben:** Schreibe jedes x in A direkt an die richtige Position in B

Eingabefeld
Ausgabefeld
beschränkt Universum $\{0, \dots, k\}$

```

COUNTINGSORT(int[]  $A$ , int[]  $B$ , int  $k$ )
    sei  $C[0 \dots k] = \langle 0, 0, \dots, 0 \rangle$  ein neues Feld
    for  $j = 1$  to  $A.length$  do  $C[A[j]] = C[A[j]] + 1$ 
    //  $C[i]$  enthält jetzt die Anzahl der Elemente gleich  $i$  in  $A$ 
    for  $i = 1$  to  $k$  do  $C[i] = C[i] + C[i - 1]$ 
    //  $C[i]$  enthält jetzt die Anzahl der Elemente  $\leq i$  in  $A$ 
    for  $j = A.length$  downto 1 do
         $B[C[A[j]]] = A[j]$ 
         $C[A[j]] = C[A[j]] - 1$ 
  
```

Initialisierung

1a) Zählen

1b) Berechnen

2) Schreiben

Laufzeit:
 $\mathcal{O}(n + k)$

1. Zwischentest am Do, 13.11., ab 8:15 Uhr

Bitte melden Sie sich **jetzt** in WueCampus für den Test an, wenn Sie das noch nicht getan haben.

(Sonst drucken wir keinen Test für Sie aus. Frist: Mi, 10:00 Uhr.)

Zusammenfassung

- Jedes vergleichsbasierte Sortierverfahren braucht im Worst-Case $\Omega(n \log n)$ Vergleiche.
- COUNTINGSORT sortiert Zahlen in $\{0, \dots, k\}$ (**stabil!**)
Laufzeit für n Zahlen: $\mathcal{O}(n + k)$
- RADIXSORT sortiert s -stellige b -adische Zahlen
- BUCKETSORT sortiert gleichverteilte zufällige Zahlen

Zusammenfassung

- Jedes vergleichsbasierte Sortierverfahren braucht im Worst-Case $\Omega(n \log n)$ Vergleiche.
- COUNTINGSORT sortiert Zahlen in $\{0, \dots, k\}$ (**stabil!**)
Laufzeit für n Zahlen: $\mathcal{O}(n + k)$
- RADIXSORT sortiert s -stellige b -adische Zahlen
- BUCKETSORT sortiert gleichverteilte zufällige Zahlen

Zusammenfassung

■ Jedes vergleichsbasierte Sortierverfahren braucht im Worst-Case $\Omega(n \log n)$ Vergleiche.

■ COUNTINGSORT sortiert Zahlen in $\{0, \dots, k\}$ (**stabil!**)

Laufzeit für n Zahlen: $\mathcal{O}(n + k)$

■ RADIXSORT max. s Stellen sortiert s -stellige b -adische Zahlen

■ BUCKETSORT sortiert gleichverteilte zufällige Zahlen

Zusammenfassung

■ Jedes vergleichsbasierte Sortierverfahren braucht im Worst-Case $\Omega(n \log n)$ Vergleiche.

■ COUNTINGSORT sortiert Zahlen in $\{0, \dots, k\}$ (**stabil!**)

Laufzeit für n Zahlen: $\mathcal{O}(n + k)$

■ RADIXSORT sortiert ^{max. s Stellen} s -stellige ^{b mögliche unterschiedliche Ziffern} b -adische Zahlen

■ BUCKETSORT sortiert gleichverteilte zufällige Zahlen

Zusammenfassung

■ Jedes vergleichsbasierte Sortierverfahren braucht im Worst-Case $\Omega(n \log n)$ Vergleiche.

■ COUNTINGSORT sortiert Zahlen in $\{0, \dots, k\}$ (**stabil!**)

Laufzeit für n Zahlen: $\mathcal{O}(n + k)$

■ RADIXSORT max. s Stellen sortiert s -stellige b mögliche unterschiedliche Ziffern b -adische Zahlen z.B. Dezimalzahl: $b = 10$

■ BUCKETSORT sortiert gleichverteilte zufällige Zahlen

Zusammenfassung

■ Jedes vergleichsbasierte Sortierverfahren braucht im Worst-Case $\Omega(n \log n)$ Vergleiche.

■ COUNTINGSORT sortiert Zahlen in $\{0, \dots, k\}$ (**stabil!**)

Laufzeit für n Zahlen: $\mathcal{O}(n + k)$

■ RADIXSORT max. s Stellen sortiert s -stellige b mögliche unterschiedliche Ziffern b -adische Zahlen

z.B. Dezimalzahl: $b = 10$
Binärzahl: $b = 2$

■ BUCKETSORT sortiert gleichverteilte zufällige Zahlen

Zusammenfassung

■ Jedes vergleichsbasierte Sortierverfahren braucht im Worst-Case $\Omega(n \log n)$ Vergleiche.

■ COUNTINGSORT sortiert Zahlen in $\{0, \dots, k\}$ (**stabil!**)

Laufzeit für n Zahlen: $\mathcal{O}(n + k)$

■ RADIXSORT max. s Stellen sortiert s -stellige b mögliche unterschiedliche Ziffern b -adische Zahlen

z.B. Dezimalzahl: $b = 10$
Binärzahl: $b = 2$
Wörter: $b = 26$

■ BUCKETSORT sortiert gleichverteilte zufällige Zahlen

RADIXSORT

Frage: Gegeben Liste von Menschen mit deren Geburtstagen.
Wie würden Sie die Liste nach Alter sortieren?

RADIXSORT

(Jahr, Monat, Tag)

Frage: Gegeben Liste von Menschen mit deren Geburtstagen.
Wie würden Sie die Liste nach Alter sortieren?

RADIXSORT

(Jahr, Monat, Tag)

Frage: Gegeben Liste von Menschen mit deren Geburtstagen.
Wie würden Sie die Liste nach Alter sortieren?

Drei (?) Lösungen:

RADIXSORT

(Jahr, Monat, Tag)

Frage: Gegeben Liste von Menschen mit deren Geburtstagen.
Wie würden Sie die Liste nach Alter sortieren?

Drei (?) Lösungen:

- Geburtstage in Anzahl Tage seit 1.1.1970 umrechnen, dann vergleichsbasiertes Sortierverfahren verwenden.

RADIXSORT

(Jahr, Monat, Tag)

Frage: Gegeben Liste von Menschen mit deren Geburtstagen.
Wie würden Sie die Liste nach Alter sortieren?

Drei (?) Lösungen:

- Geburtstage in Anzahl Tage seit 1.1.1970 umrechnen, dann vergleichsbasiertes Sortierverfahren verwenden.
- Spezielle Vergleichsroutine schreiben und in vergleichsbasiertes Sortierverfahren einbauen.

RADIXSORT

(Jahr, Monat, Tag)

Frage: Gegeben Liste von Menschen mit deren Geburtstagen.
Wie würden Sie die Liste nach Alter sortieren?

Drei (?) Lösungen:

- Geburtstage in Anzahl Tage seit 1.1.1970 umrechnen, dann vergleichsbasiertes Sortierverfahren verwenden.
- Spezielle Vergleichsroutine schreiben und in vergleichsbasiertes Sortierverfahren einbauen.
- Liste $3\times$ sortieren: je $1\times$ nach Jahr, Monat, Tag.

RADIXSORT

(Jahr, Monat, Tag)

Frage: Gegeben Liste von Menschen mit deren Geburtstagen.
Wie würden Sie die Liste nach Alter sortieren?

Drei (?) Lösungen:

- Geburtstage in Anzahl Tage seit 1.1.1970 umrechnen, dann vergleichsbasiertes Sortierverfahren verwenden.
- Spezielle Vergleichsroutine schreiben und in vergleichsbasiertes Sortierverfahren einbauen.
- Liste $3 \times$ sortieren: je $1 \times$ nach Jahr, Monat, Tag.

RADIXSORT(A, s)

RADIXSORT

(Jahr, Monat, Tag)

Frage: Gegeben Liste von Menschen mit deren Geburtstagen.
Wie würden Sie die Liste nach Alter sortieren?

Drei (?) Lösungen:

- Geburtstage in Anzahl Tage seit 1.1.1970 umrechnen, dann vergleichsbasiertes Sortierverfahren verwenden.
- Spezielle Vergleichsroutine schreiben und in vergleichsbasiertes Sortierverfahren einbauen.
- Liste $3 \times$ sortieren: je $1 \times$ nach Jahr, Monat, Tag.

Aber in welcher Reihenfolge?

Anz. Stellen (hier: 3)

RADIXSORT($\overset{\text{Anz. Stellen (hier: 3)}{\curvearrowright}}{A}, s$)

RADIXSORT

(Jahr, Monat, Tag)

Frage: Gegeben Liste von Menschen mit deren Geburtstagen.
Wie würden Sie die Liste nach Alter sortieren?

Drei (?) Lösungen:

- Geburtstage in Anzahl Tage seit 1.1.1970 umrechnen, dann vergleichsbasiertes Sortierverfahren verwenden.
- Spezielle Vergleichsroutine schreiben und in vergleichsbasiertes Sortierverfahren einbauen.
- Liste $3 \times$ sortieren: je $1 \times$ nach Jahr, Monat, Tag.

Aber in welcher Reihenfolge?

Anz. Stellen (hier: 3)

RADIXSORT(A, s)

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

RADIXSORT

(Jahr, Monat, Tag)

Frage: Gegeben Liste von Menschen mit deren Geburtstagen.
Wie würden Sie die Liste nach Alter sortieren?

Drei (?) Lösungen:

- Geburtstage in Anzahl Tage seit 1.1.1970 umrechnen, dann vergleichsbasiertes Sortierverfahren verwenden.
- Spezielle Vergleichsroutine schreiben und in vergleichsbasiertes Sortierverfahren einbauen.
- Liste $3 \times$ sortieren: je $1 \times$ nach Jahr, Monat, Tag.

Aber in welcher Reihenfolge?

Anz. Stellen (hier: 3)

RADIXSORT(A, s)

for $i = 1$ **to** s **do**

1 = Index der **niederwertigsten** (!) Stelle

└ sortiere A **stabil** nach der i -ten Stelle

RADIXSORT

(Jahr, Monat, Tag)

Frage: Gegeben Liste von Menschen mit deren Geburtstagen.
Wie würden Sie die Liste nach Alter sortieren?

Drei (?) Lösungen:

- Geburtstage in Anzahl Tage seit 1.1.1970 umrechnen, dann vergleichsbasiertes Sortierverfahren verwenden.
- Spezielle Vergleichsroutine schreiben und in vergleichsbasiertes Sortierverfahren einbauen.
- Liste $3 \times$ sortieren: je $1 \times$ nach Jahr, Monat, Tag.

Aber in welcher Reihenfolge?

RADIXSORT(A, s)

Anz. Stellen (hier: 3)

for $i = 1$ **to** s **do**

1 = Index der **niederwertigsten** (!) Stelle

└ sortiere A **stabil** nach der i -ten Stelle

z.B. mit COUNTINGSORT

RADIXSORT

(Jahr, Monat, Tag)

Frage: Gegeben Liste von Menschen mit deren Geburtstagen.
Wie würden Sie die Liste nach Alter sortieren?

Drei (?) Lösungen:

- Geburtstage in Anzahl Tage seit 1.1.1970 umrechnen, dann vergleichsbasiertes Sortiervverfahren verwenden.
- Spezielle Vergleichsroutine schreiben und in vergleichsbasiertes Sortiervverfahren einbauen.
- Liste $3 \times$ sortieren: je $1 \times$ nach Jahr, Monat, Tag.

Aber in welcher Reihenfolge?

RADIXSORT(A, s)

Anz. Stellen (hier: 3)

for $i = 1$ **to** s **do**

1 = Index der **niederwertigsten** (!) Stelle

└ sortiere A **stabil** nach der i -ten Stelle

z.B. mit COUNTINGSORT



RADIXSORT

(Jahr, Monat, Tag)

Frage: Gegeben Liste von Menschen mit deren Geburtstagen.
Wie würden Sie die Liste nach Alter sortieren?

Drei (?) Lösungen:

- Geburtstage in Anzahl Tage seit 1.1.1970 umrechnen, dann vergleichsbasiertes Sortierverfahren verwenden.
- Spezielle Vergleichsroutine schreiben und in vergleichsbasiertes Sortierverfahren einbauen.
- Liste $3 \times$ sortieren: je $1 \times$ nach Jahr, Monat, Tag. Aber in welcher Reihenfolge?

RADIXSORT(A, s)

Anz. Stellen (hier: 3)

for $i = 1$ **to** s **do**

1 = Index der **niederwertigsten** (!) Stelle

└ sortiere A **stabil** nach der i -ten Stelle

z.B. mit COUNTINGSORT

Laufzeit?

Beispiel

Sortiere

A

25	13	31	23	11	37	15
----	----	----	----	----	----	----

RADIXSORT(*A*, *s*)

for *i* = 1 **to** *s* **do**

└ sortiere *A* **stabil** nach der *i*-ten Stelle

Beispiel

Sortiere

A

25	13	31	23	11	37	15
----	----	----	----	----	----	----

Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

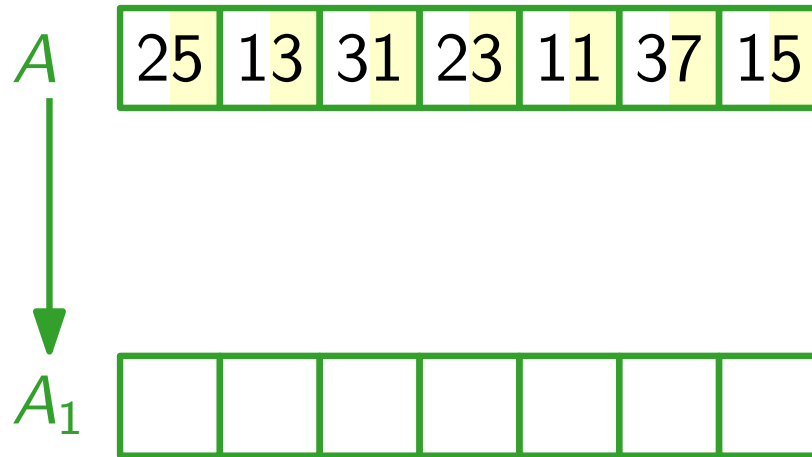
RADIXSORT(*A*, *s*)

for *i* = 1 **to** *s* **do**

└ sortiere *A* **stabil** nach der *i*-ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

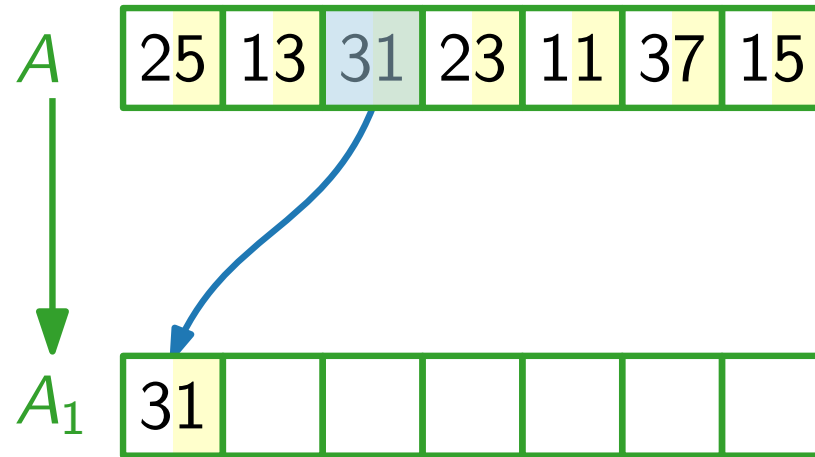
RADIXSORT(A, s)

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

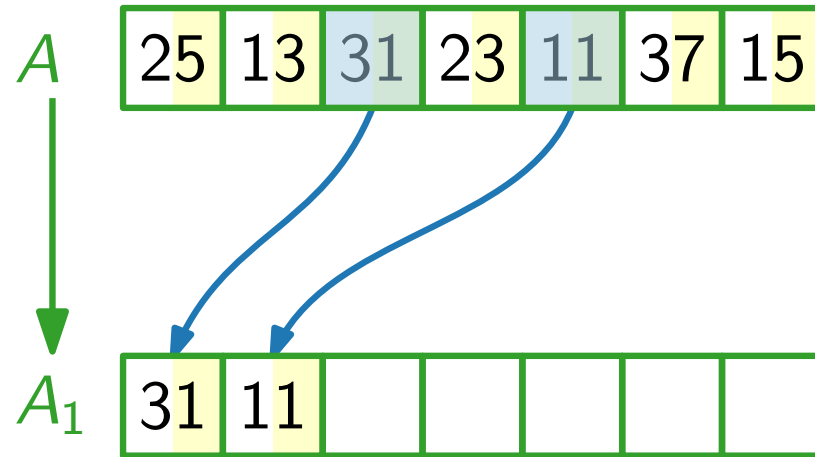
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

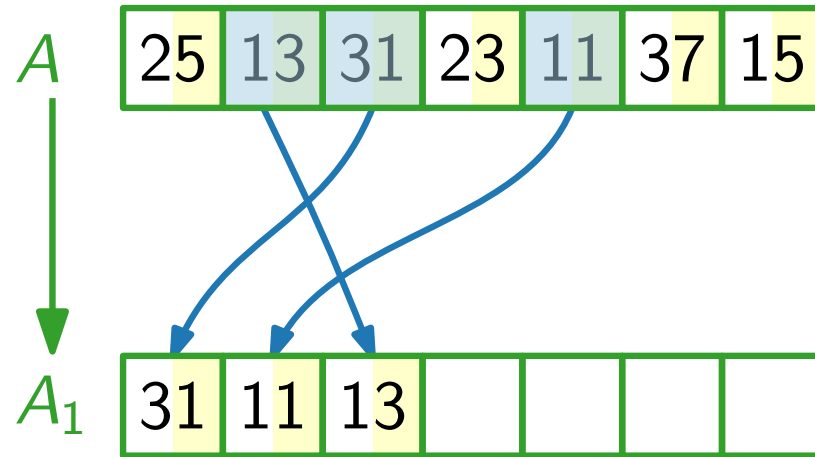
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

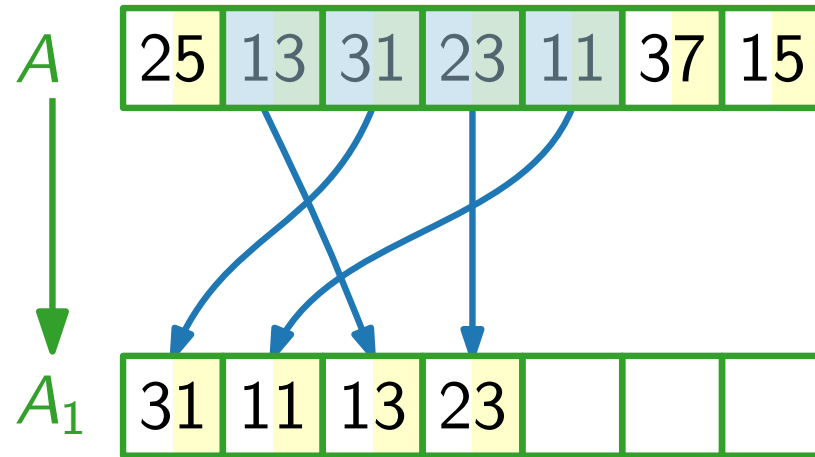
RADIXSORT(A, s)

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

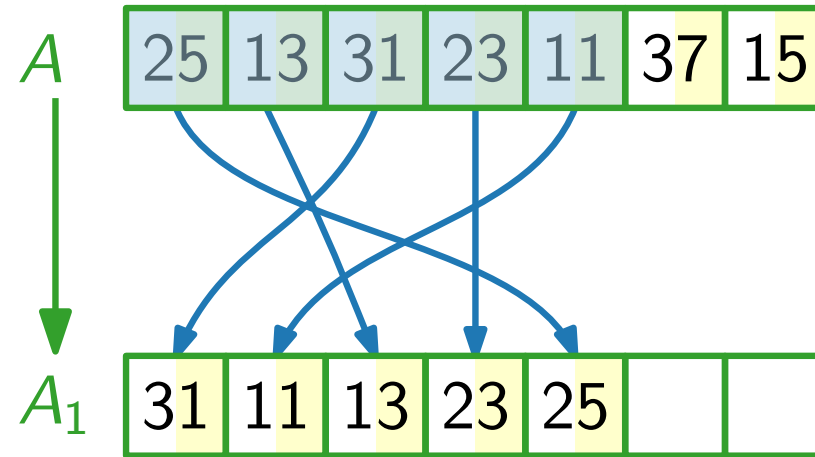
RADIXSORT(A, s)

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

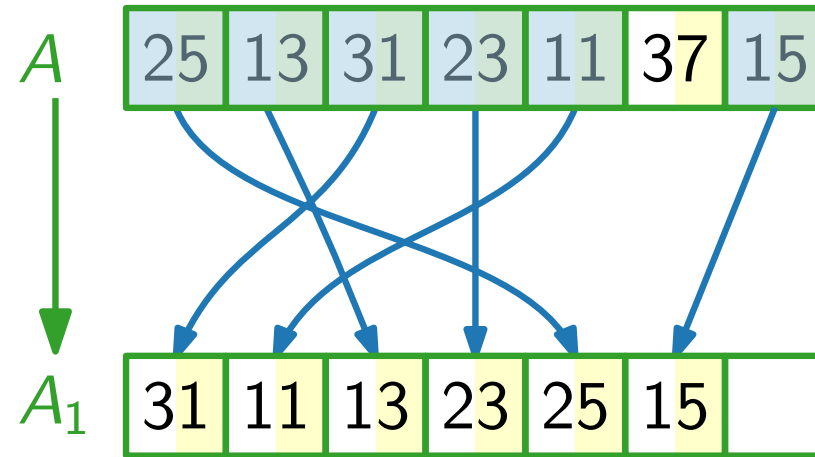
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

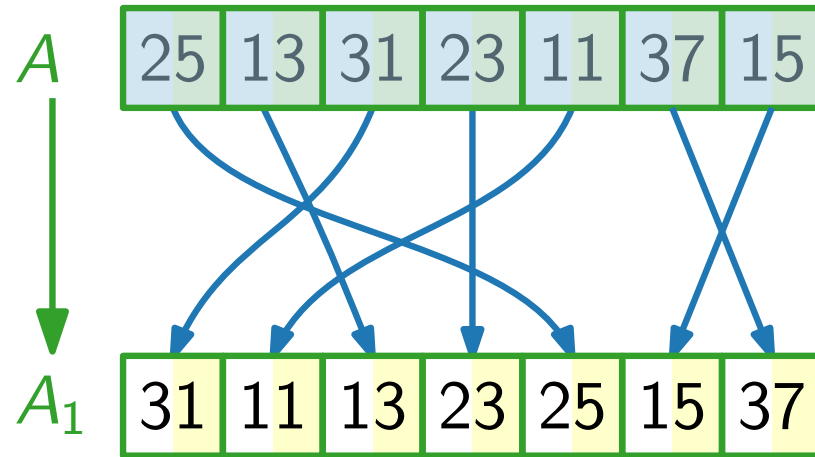
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

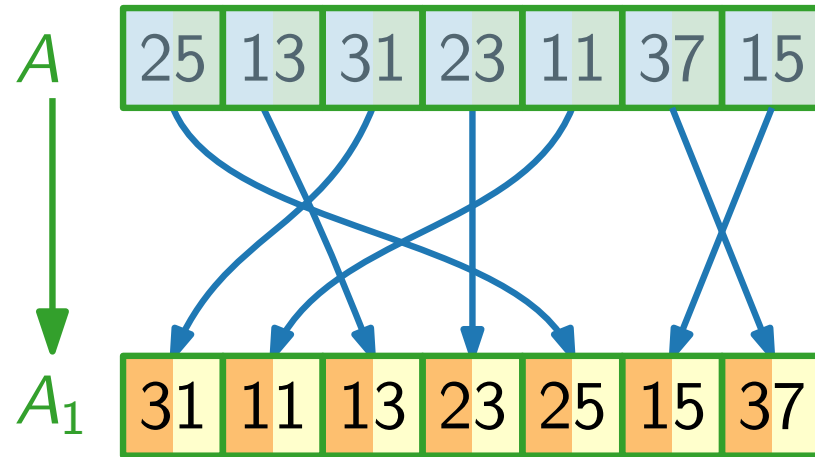
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

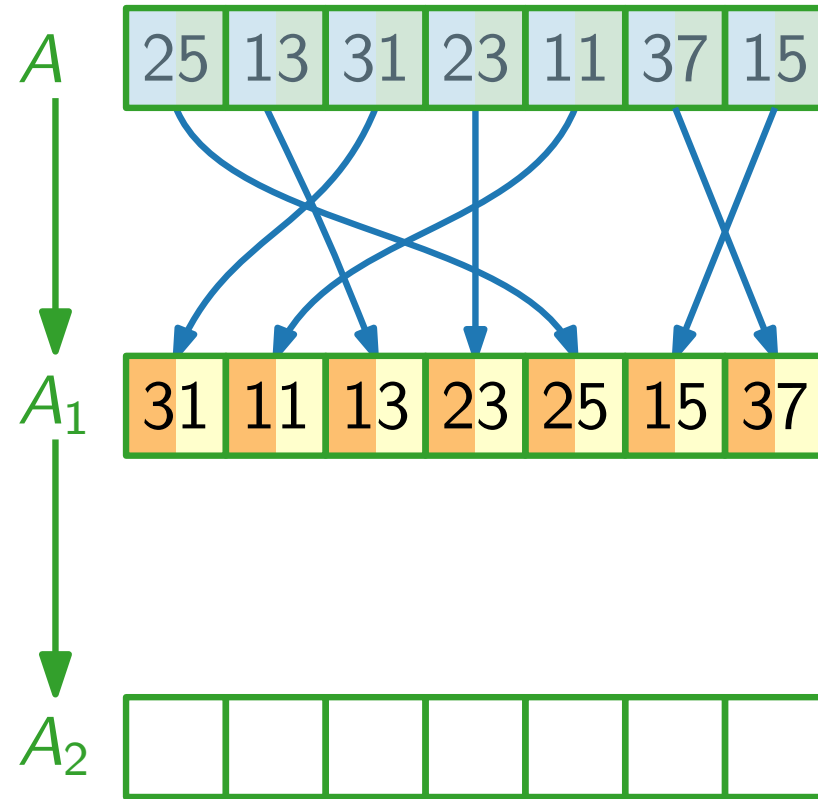
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

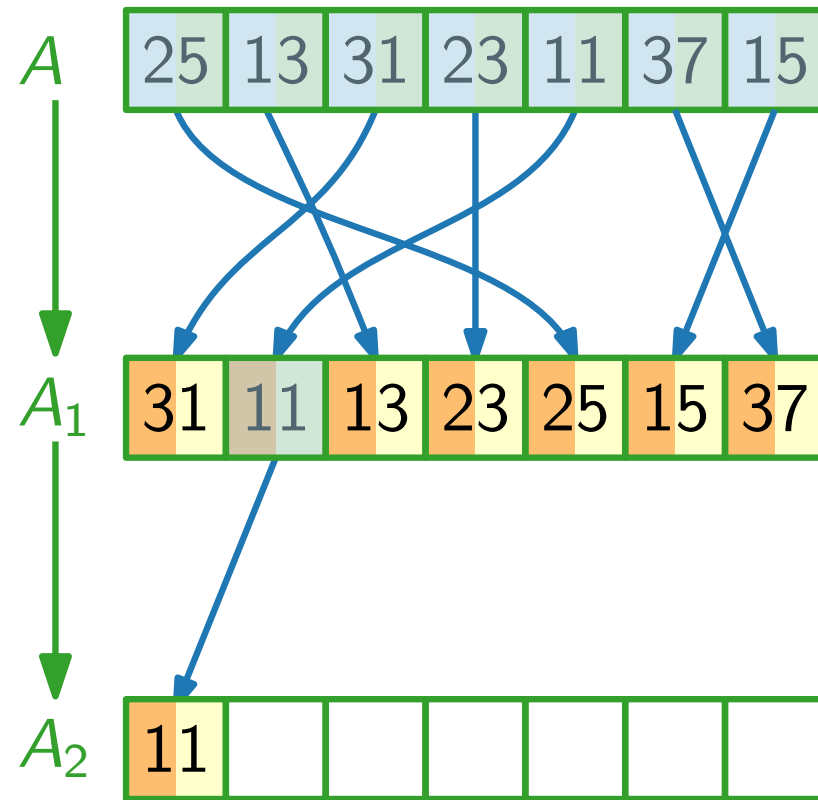
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

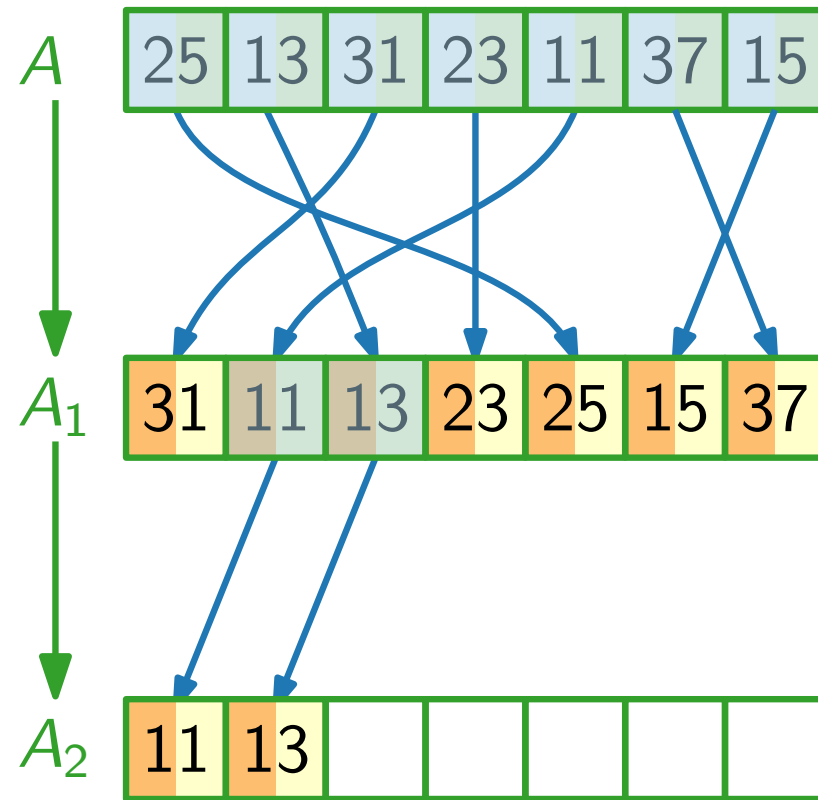
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

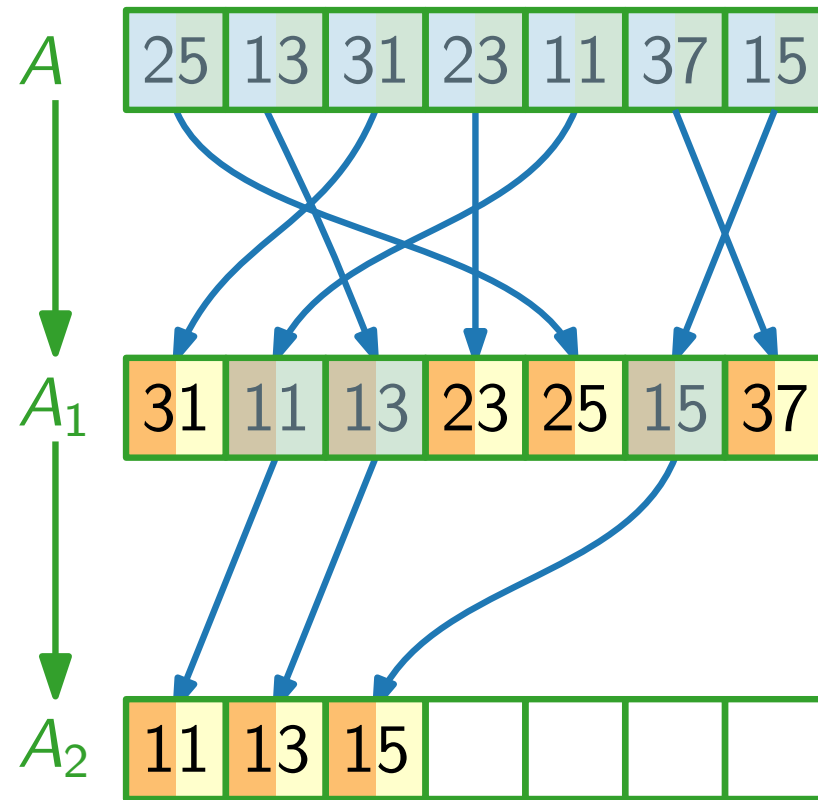
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

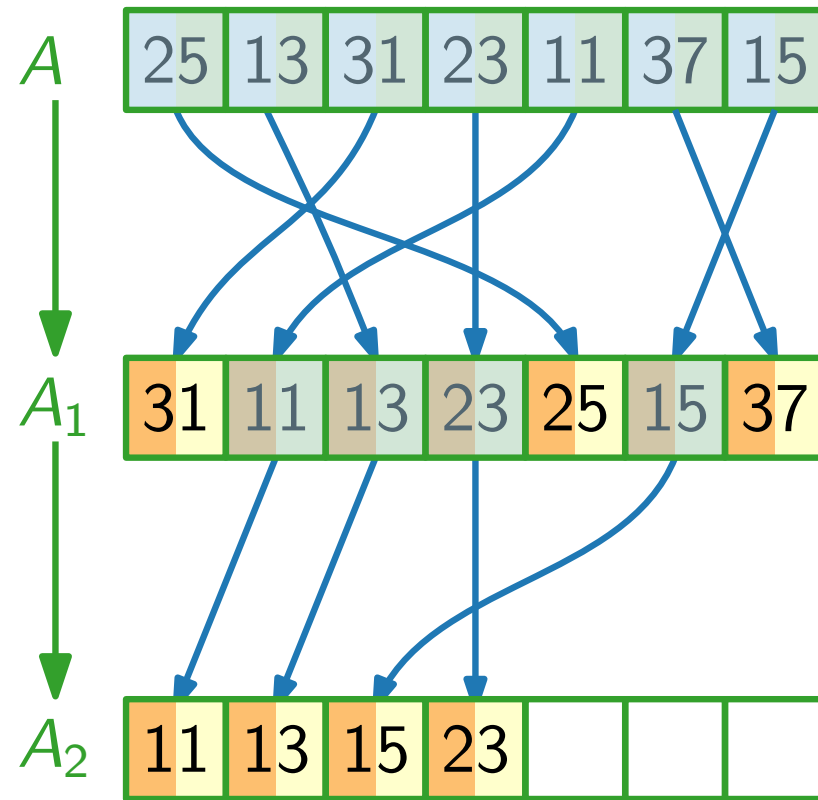
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

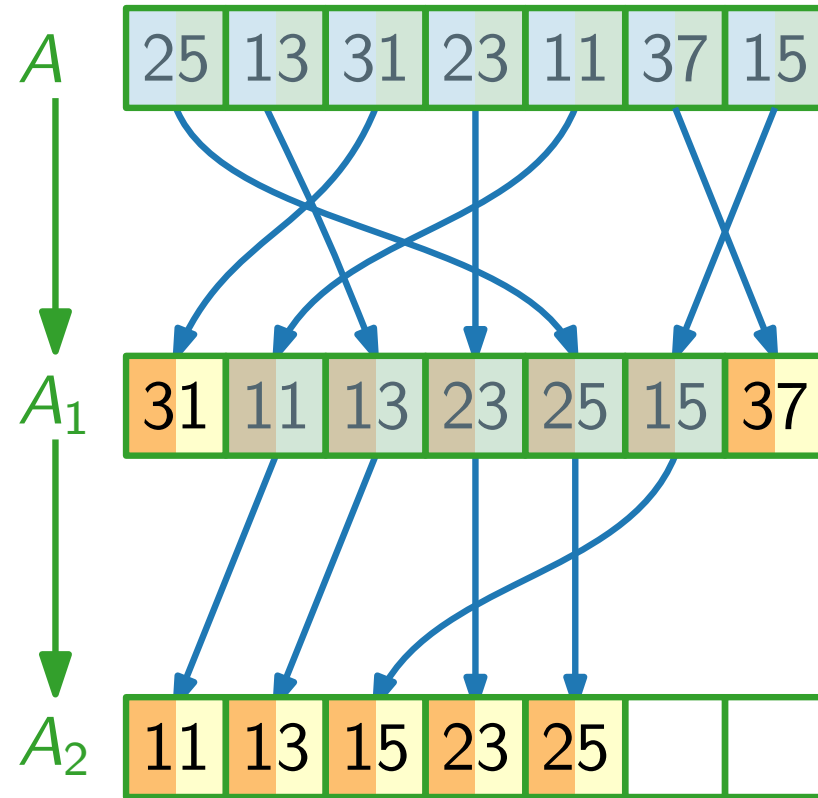
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

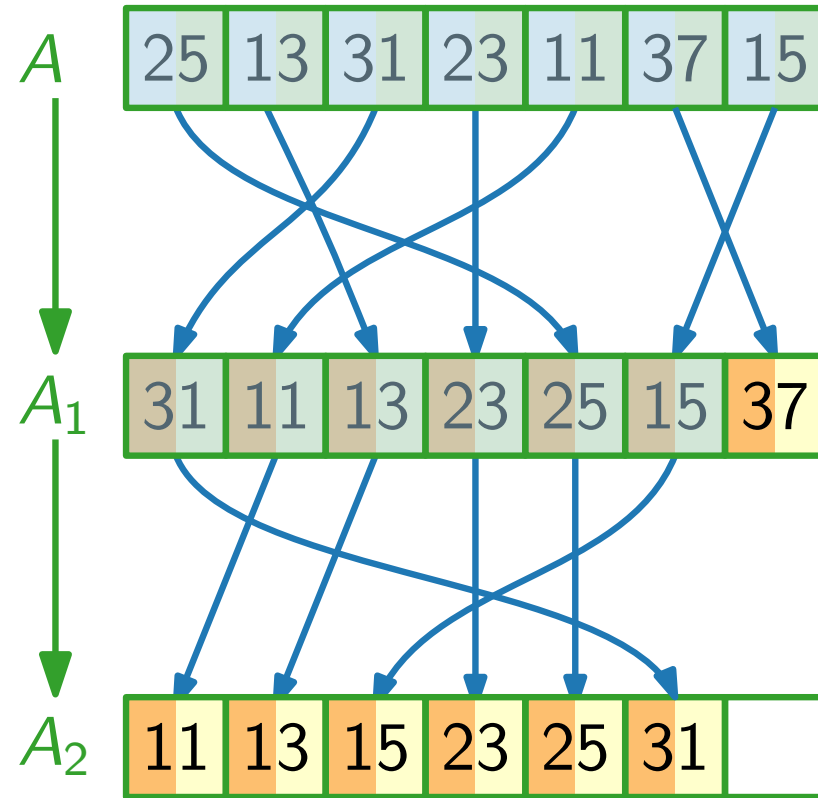
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

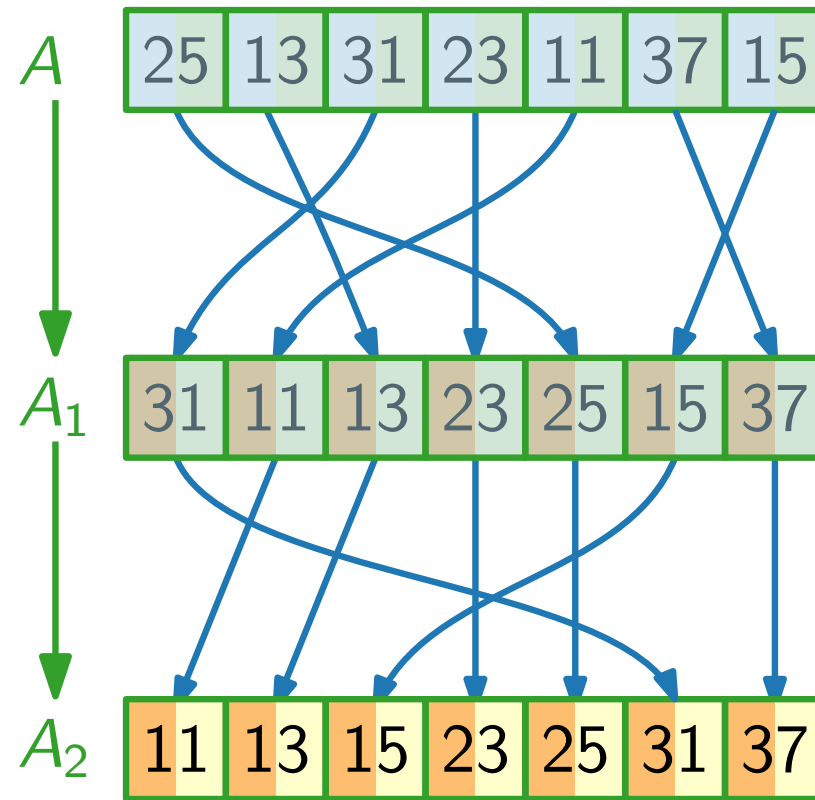
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

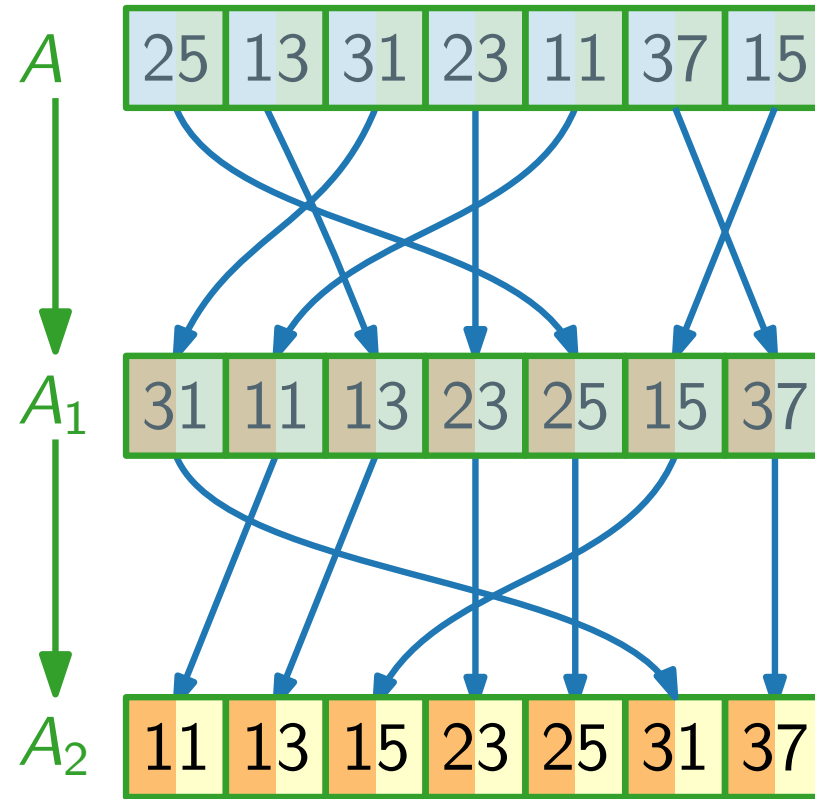
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

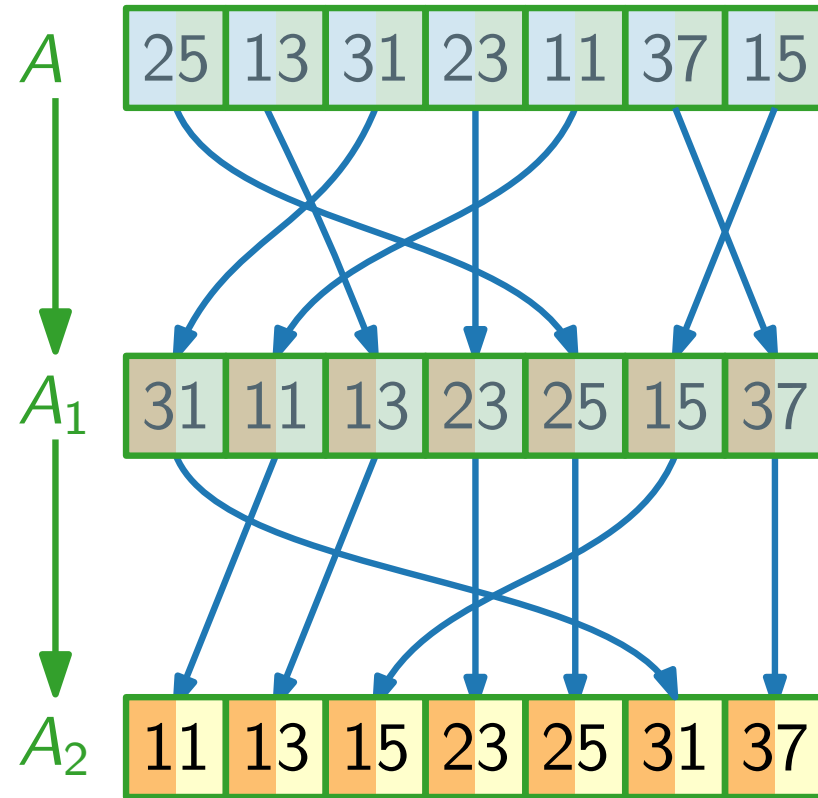
$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Beispiel

Sortiere



Gemäß RADIXSORT erst nach Einern,
dann (stabil) nach Zehnern.

Demo.

<https://algo.uni-trier.de/demos/sort.html>



$\text{RADIXSORT}(A, s)$

for $i = 1$ **to** s **do**

└ sortiere A **stabil** nach der i -ten Stelle

Zusammenfassung

- Jedes vergleichsbasierte Sortierverfahren braucht im schlechtesten Fall $\Omega(n \log n)$ Vergleiche.

- COUNTINGSORT sortiert Zahlen in $\{0, \dots, k\}$ (**stabil!**)

Laufzeit für n Zahlen: $\mathcal{O}(n + k)$

- RADIXSORT

max. s Stellen

b mögliche unterschiedliche Ziffern

sortiert s -stellige b -adische Zahlen

Laufzeit für n Zahlen: $\mathcal{O}(s \cdot (n + b))$

z.B. Dezimalzahl: $b = 10$
Binärzahl: $b = 2$
Wörter: $b = 26$

- BUCKETSORT sortiert gleichverteilte zufällige Zahlen

Zusammenfassung

- Jedes vergleichsbasierte Sortierverfahren braucht im schlechtesten Fall $\Omega(n \log n)$ Vergleiche.

- COUNTINGSORT sortiert Zahlen in $\{0, \dots, k\}$ (**stabil!**)

Laufzeit für n Zahlen: $\mathcal{O}(n + k)$

- RADIXSORT

max. s Stellen

b mögliche unterschiedliche Ziffern

sortiert s -stellige b -adische Zahlen

Laufzeit für n Zahlen: $\mathcal{O}(s \cdot (n + b))$

z.B. Dezimalzahl: $b = 10$
 Binärzahl: $b = 2$
 Wörter: $b = 26$

- BUCKETSORT sortiert gleichverteilte zufällige Zahlen

BUCKETSORT

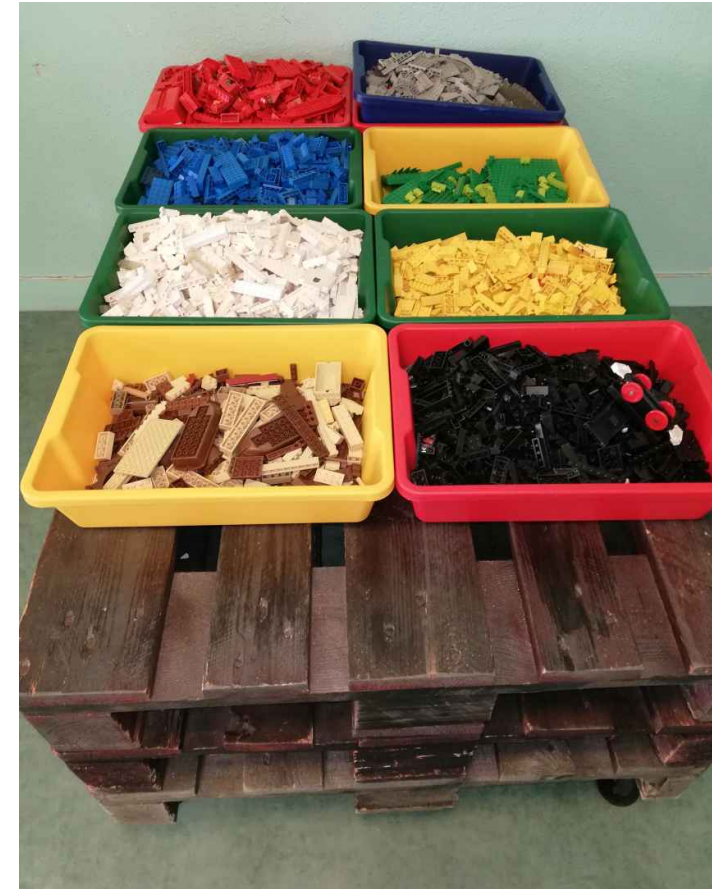


photo by JulienFou on reddit

BUCKETSORT

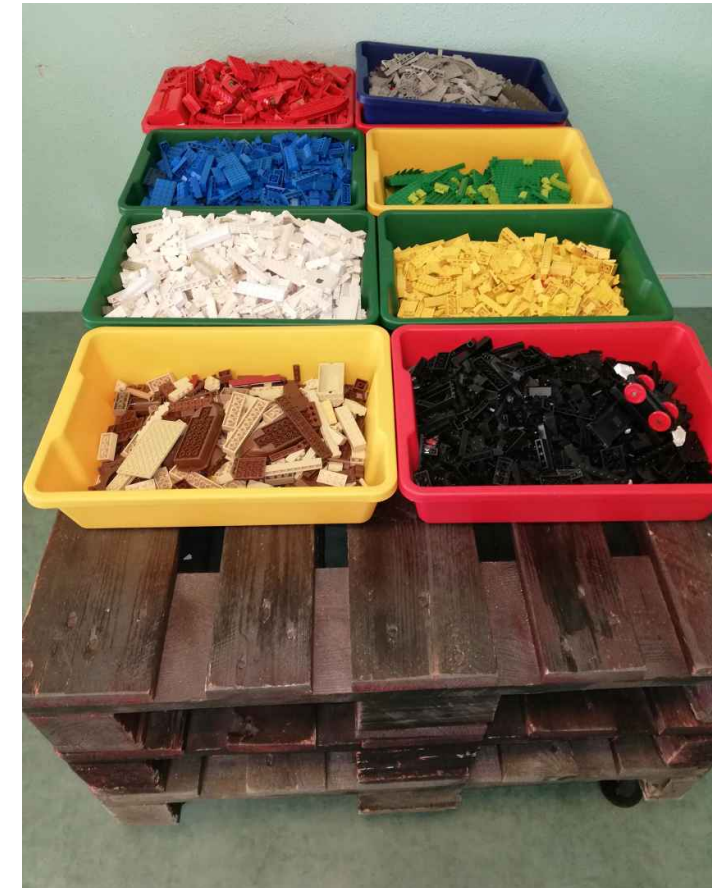


photo by JulienFou on reddit

BUCKETSORT

A	
1	.78
2	.17
3	.39
4	.21
5	.72
6	.94
7	.26
8	.12
9	.23
10	.68

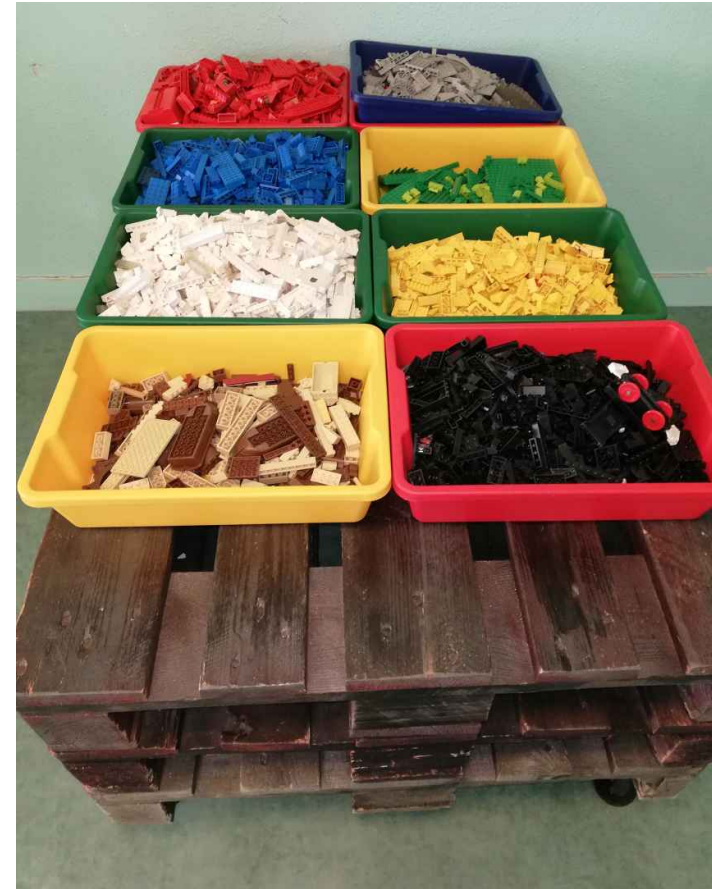


photo by JulienFou on reddit

BUCKETSORT

A

1	.78
2	.17
3	.39
4	.21
5	.72
6	.94
7	.26
8	.12
9	.23
10	.68



Eingabefeld $A[1 \dots n]$ enthält Zahlen,
zufällig und gleichverteilt aus $[0, 1)$ gezogen

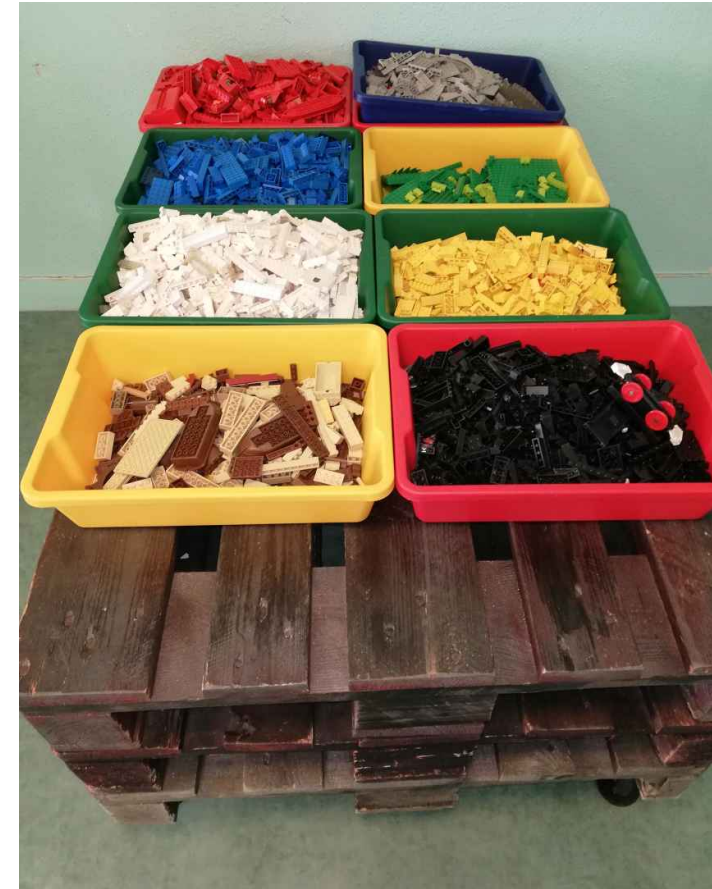


photo by JulienFou on reddit

BUCKETSORT

A

1	.78
2	.17
3	.39
4	.21
5	.72
6	.94
7	.26
8	.12
9	.23
10	.68



Eingabefeld $A[1 \dots n]$ enthält Zahlen,
zufällig und gleichverteilt aus $[0, 1)$ gezogen

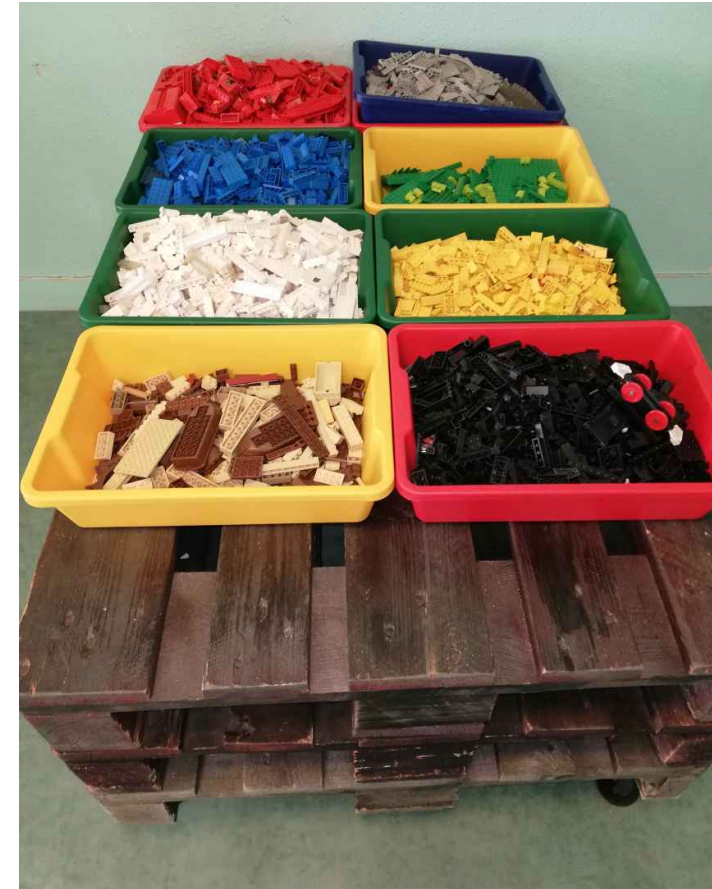
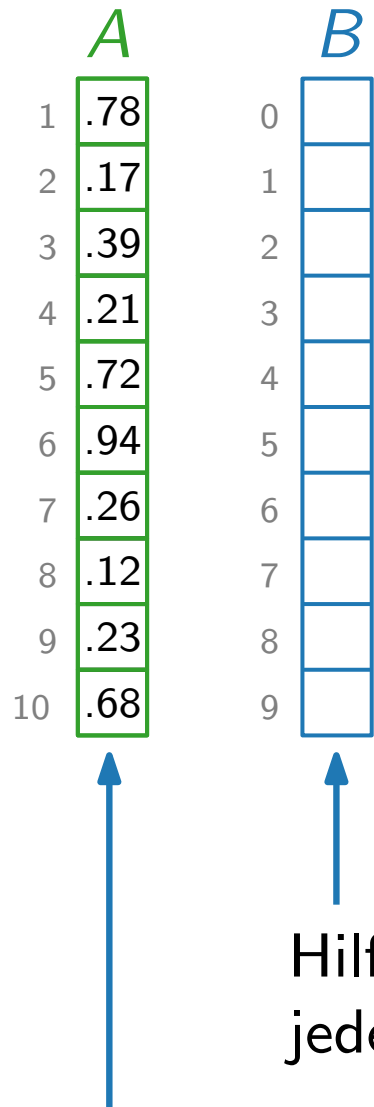


photo by JulienFou on reddit

Im Beispielt auf 2 Nach-
kommastellen gerundet

BUCKETSORT



Hilfsfeld $B[0 \dots n - 1]$;

jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen,
zufällig und gleichverteilt aus $[0, 1)$ gezogen

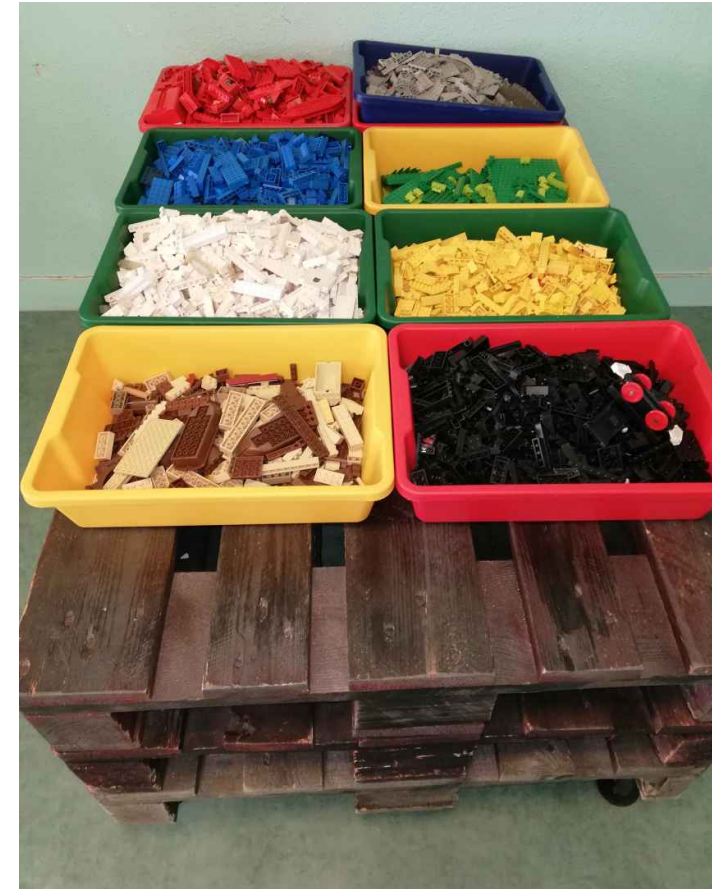
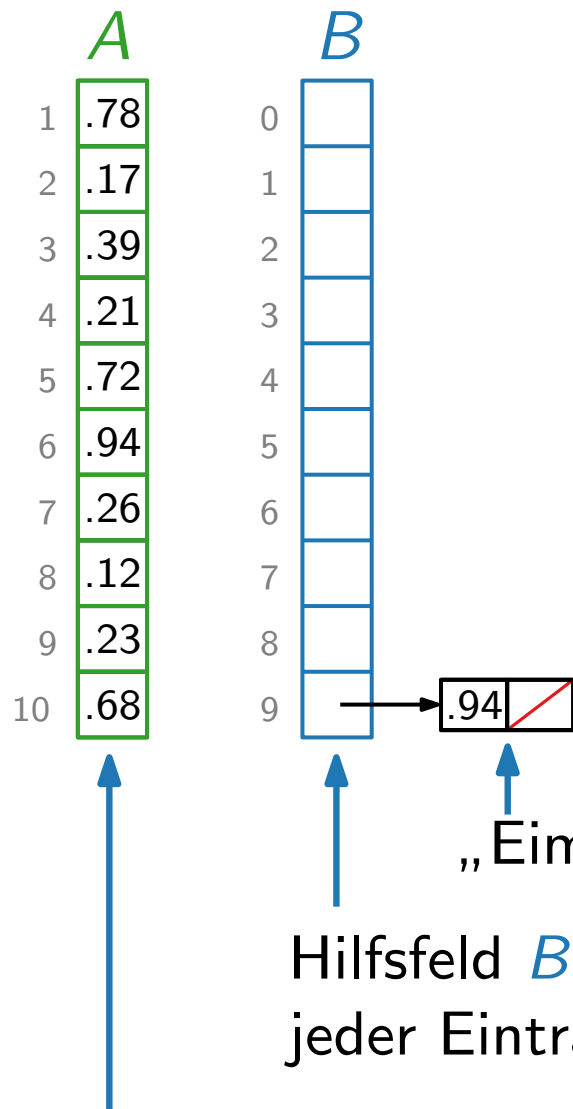


photo by JulienFou on reddit

Im Beispielt auf 2 Nachkommastellen gerundet

BUCKETSORT



„Eimerinhalt“: **Verkettete Liste** von Elementen aus A .

Hilfsfeld $B[0 \dots n - 1]$;

jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen,
zufällig und gleichverteilt aus $[0, 1)$ gezogen

Im Beispielt auf 2 Nach-
kommastellen gerundet

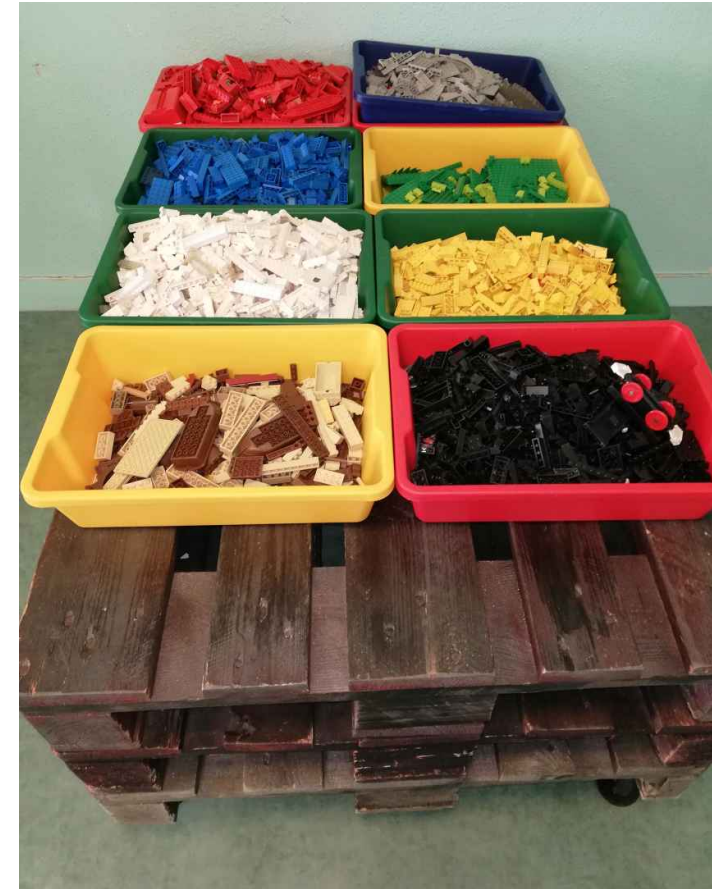
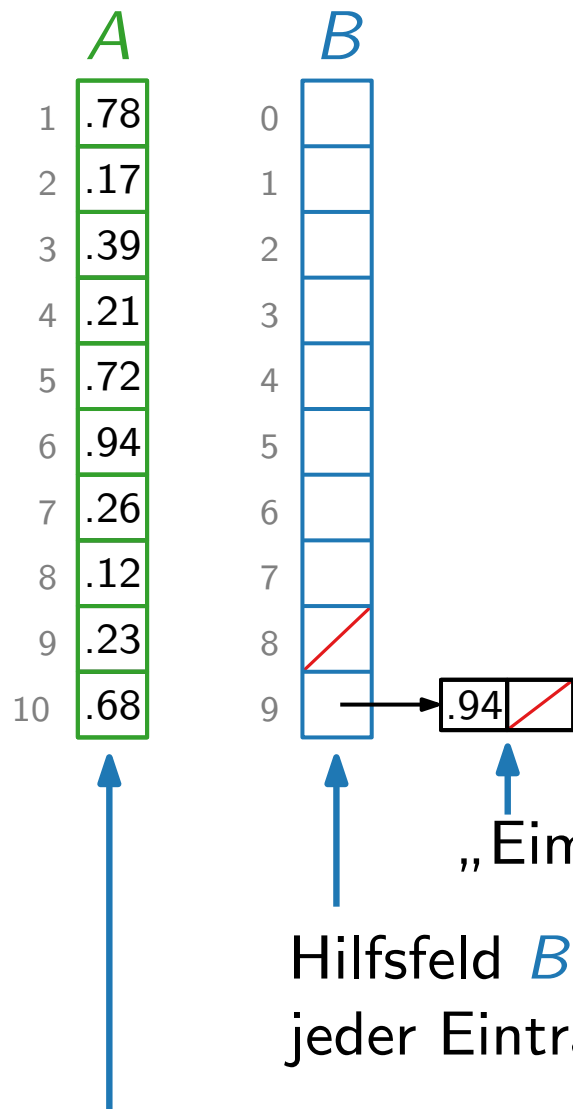


photo by JulienFou on reddit

BUCKETSORT



„Eimerinhalt“: **Verkettete Liste** von Elementen aus A .

Hilfsfeld $B[0 \dots n - 1]$;

jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen,
zufällig und gleichverteilt aus $[0, 1)$ gezogen

Im Beispielt auf 2 Nach-
kommastellen gerundet

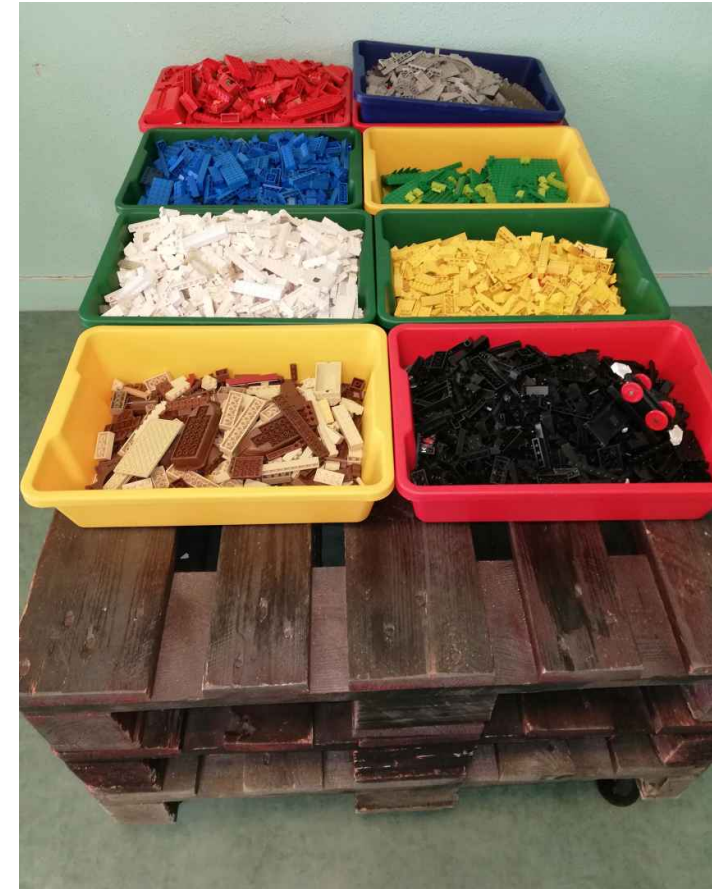
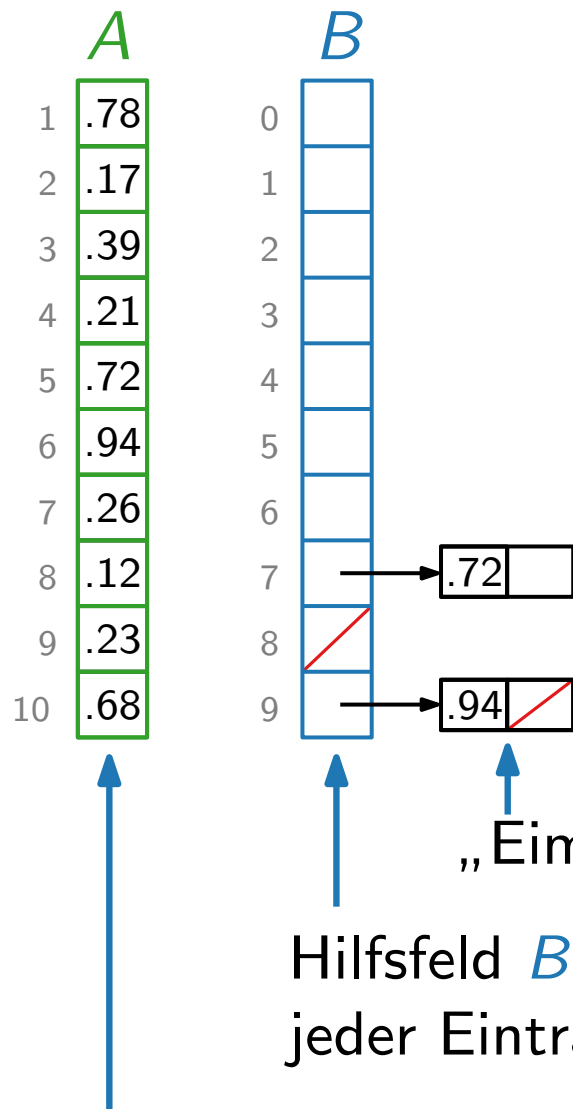


photo by JulienFou on reddit

BUCKETSORT



„Eimerinhalt“: **Verkettete Liste** von Elementen aus A .

Hilfsfeld $B[0 \dots n - 1]$;

jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen,
zufällig und gleichverteilt aus $[0, 1)$ gezogen

Im Beispielt auf 2 Nachkommastellen gerundet

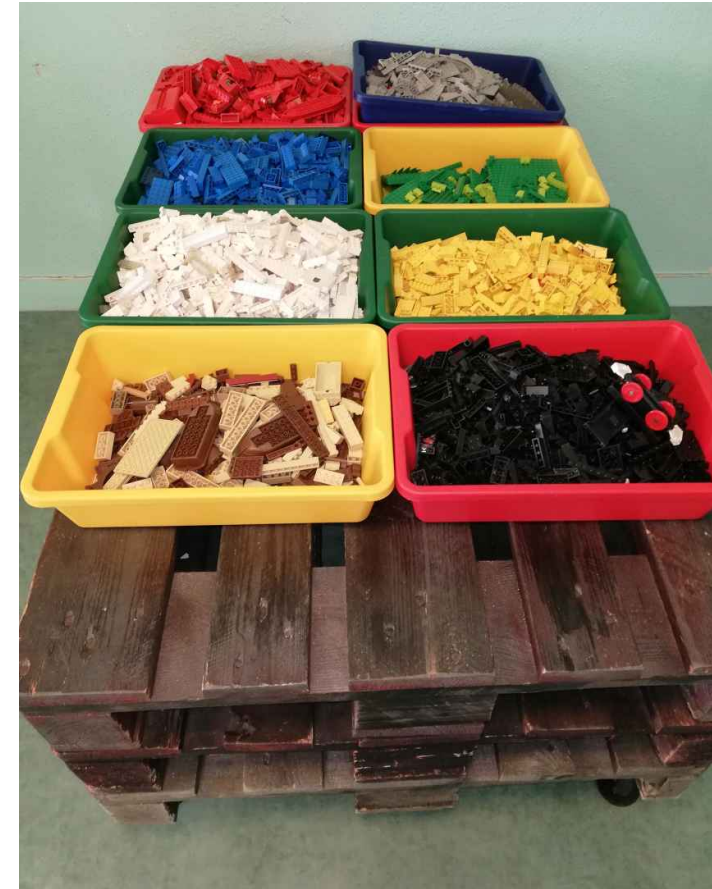
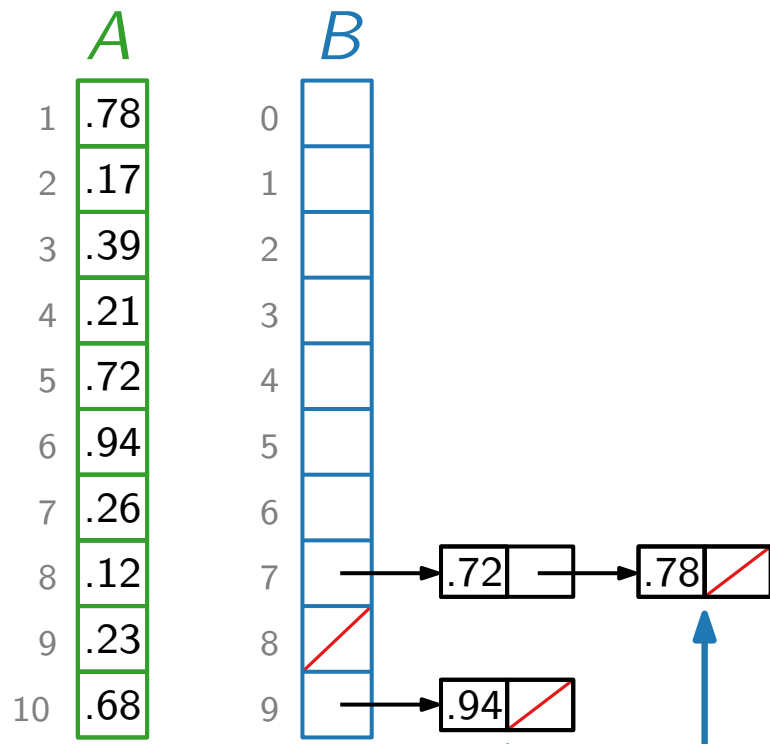


photo by JulienFou on reddit

BUCKETSORT



„Eimerinhalt“: **Verkettete Liste** von Elementen aus A .

Hilfsfeld $B[0 \dots n - 1]$;

jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen,
zufällig und gleichverteilt aus $[0, 1)$ gezogen

Im Beispielt auf 2 Nachkommastellen gerundet

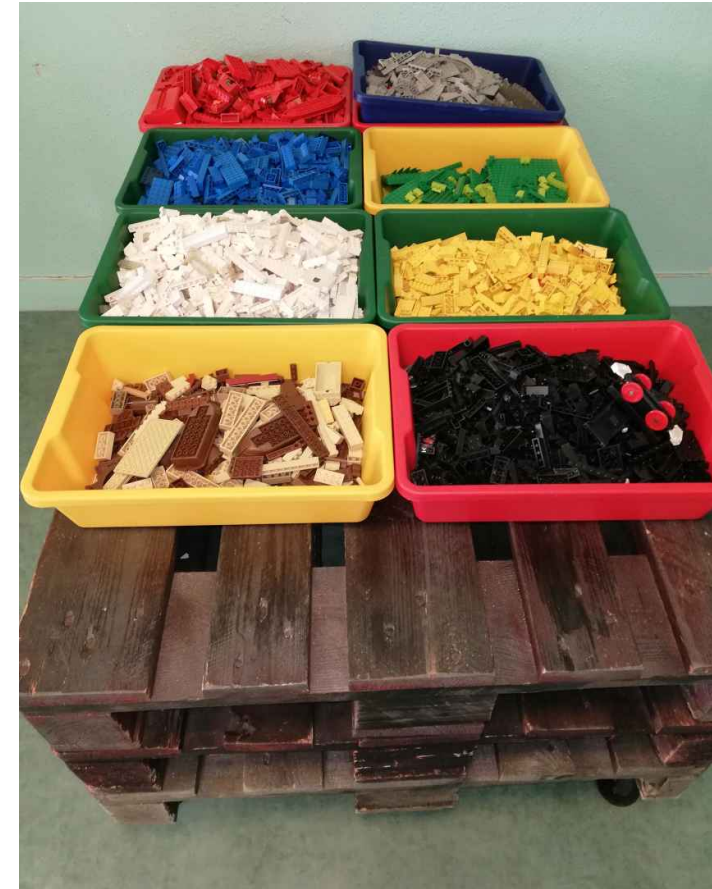
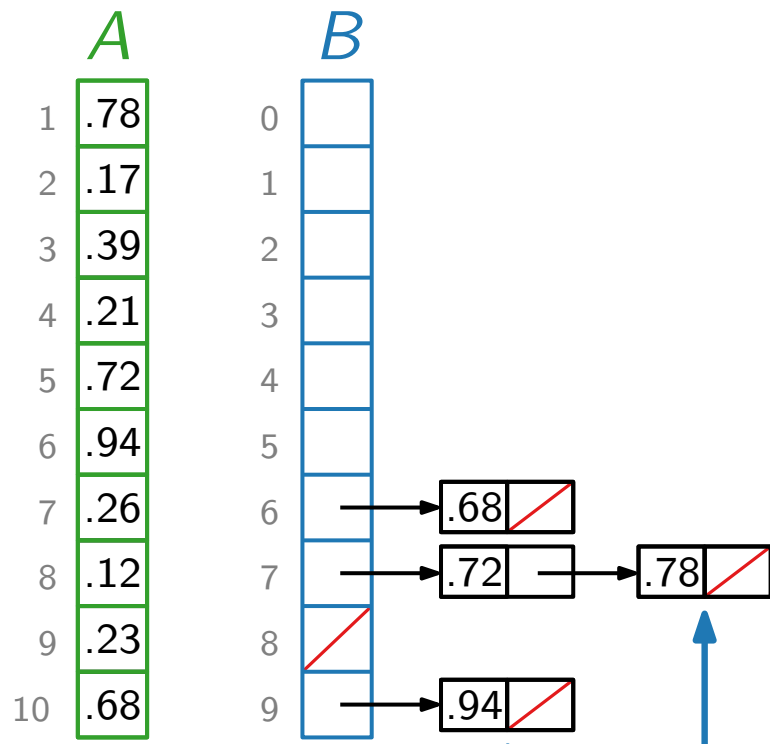


photo by JulienFou on reddit

BUCKETSORT



„Eimerinhalt“: **Verkettete Liste** von Elementen aus A .

Hilfsfeld $B[0 \dots n - 1]$;

jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen,
zufällig und gleichverteilt aus $[0, 1)$ gezogen

Im Beispielt auf 2 Nach-
kommastellen gerundet

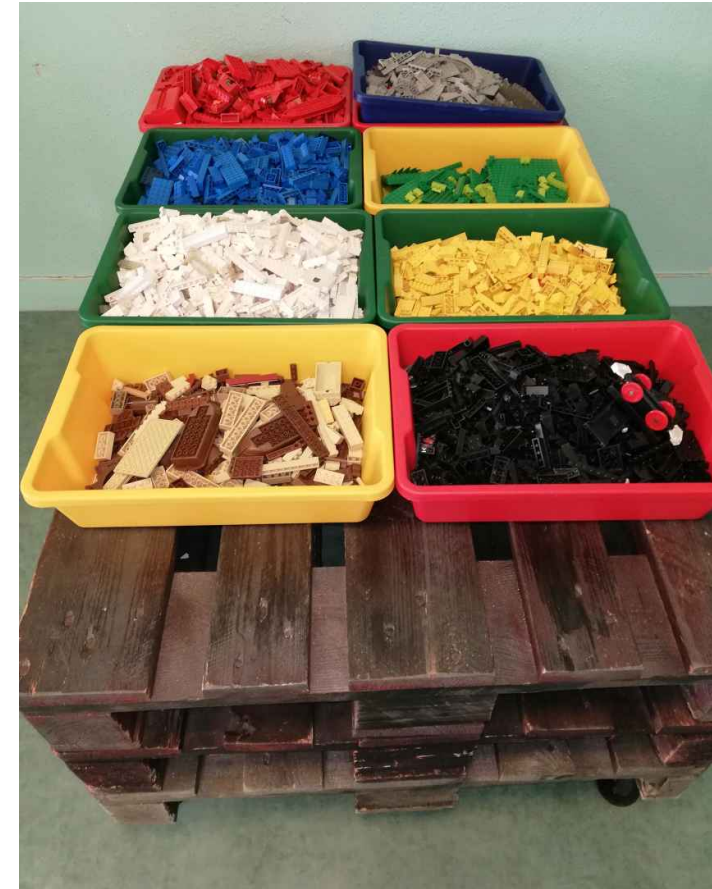
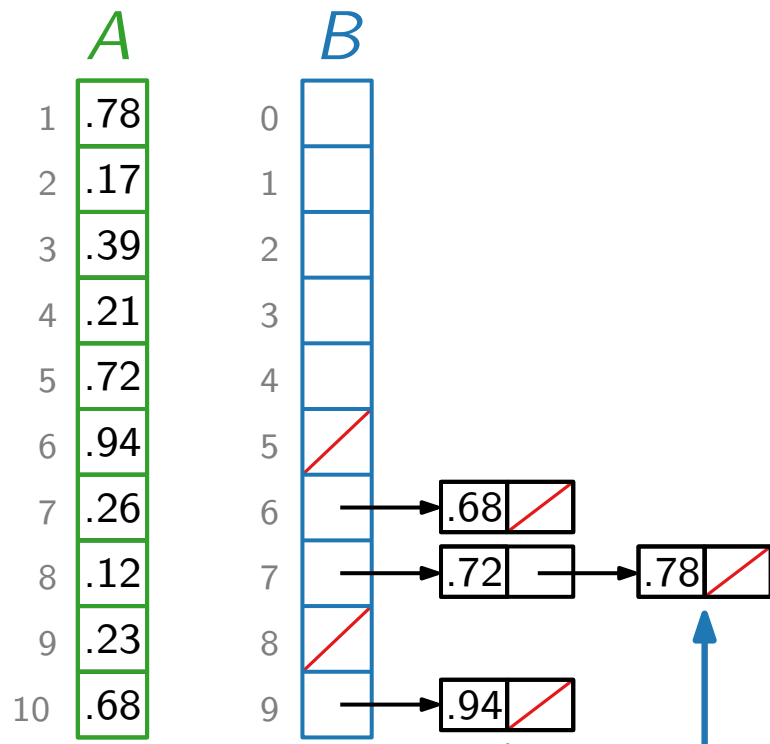


photo by JulienFou on reddit

BUCKETSORT



„Eimerinhalt“: **Verkettete Liste** von Elementen aus A .

Hilfsfeld $B[0 \dots n - 1]$;

jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen,
zufällig und gleichverteilt aus $[0, 1)$ gezogen

Im Beispielt auf 2 Nachkommastellen gerundet

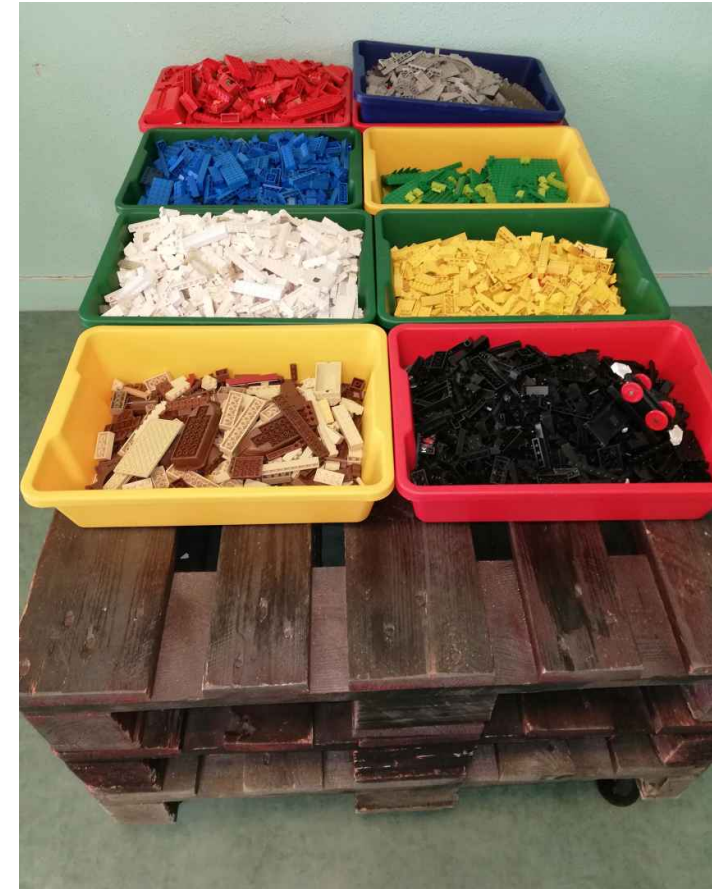
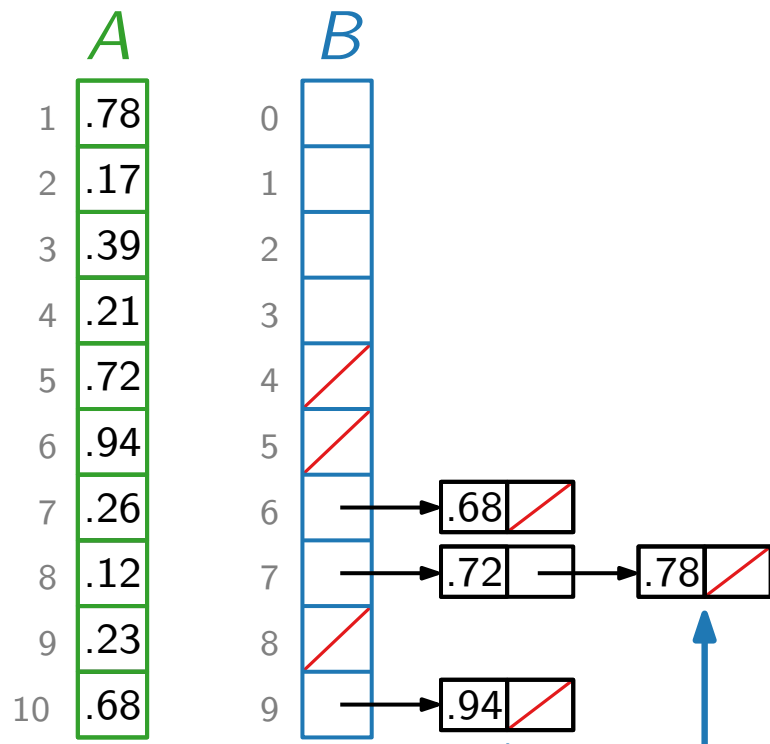


photo by JulienFou on reddit

BUCKETSORT



„Eimerinhalt“: **Verkettete Liste** von Elementen aus A .

Hilfsfeld $B[0 \dots n - 1]$;

jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen,
zufällig und gleichverteilt aus $[0, 1)$ gezogen

Im Beispielt auf 2 Nachkommastellen gerundet

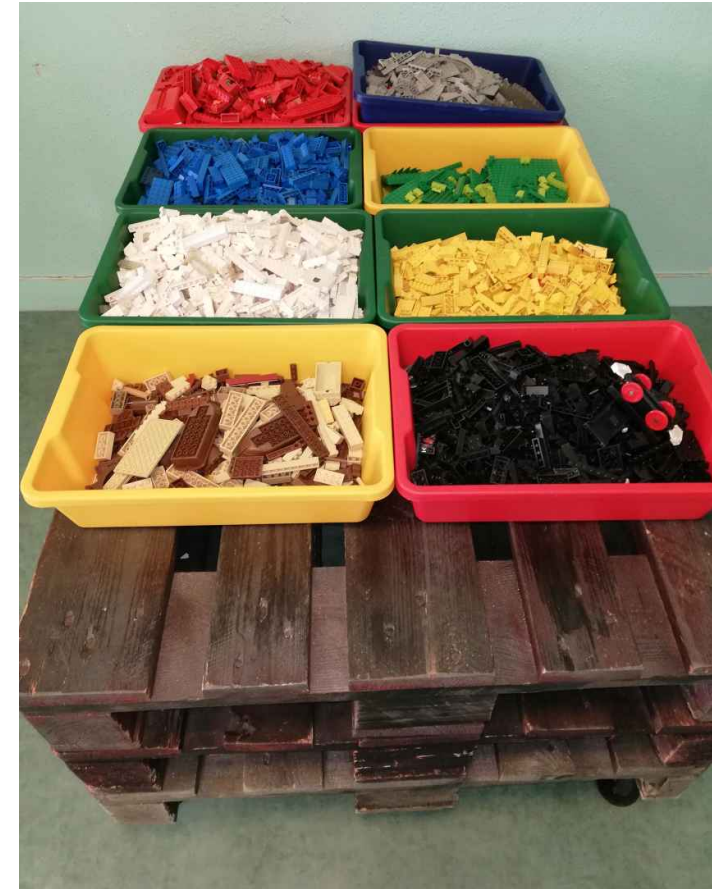
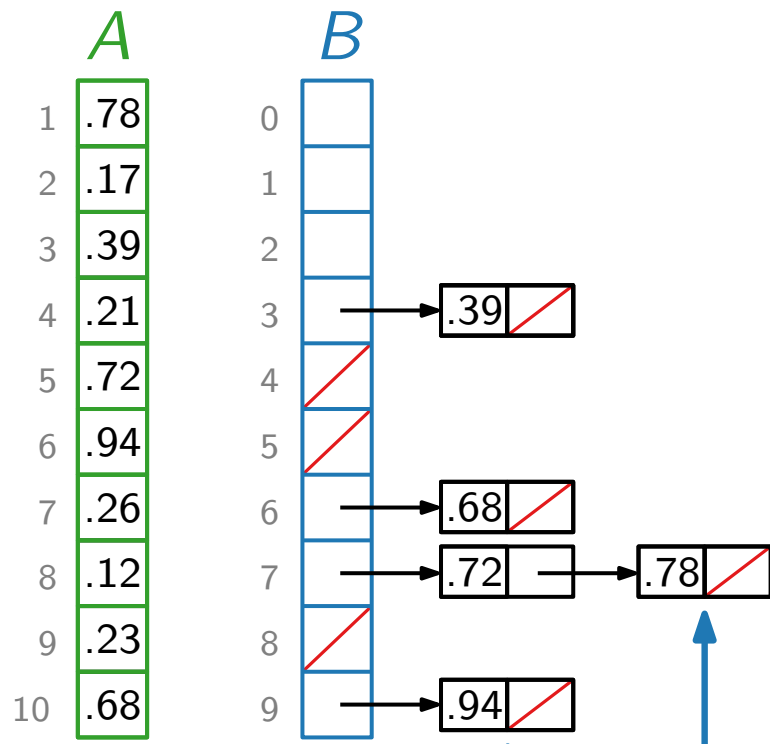


photo by JulienFou on reddit

BUCKETSORT



„Eimerinhalt“: **Verkettete Liste** von Elementen aus A .

Hilfsfeld $B[0 \dots n - 1]$;

jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen, zufällig und gleichverteilt aus $[0, 1)$ gezogen

Im Beispielt auf 2 Nachkommastellen gerundet

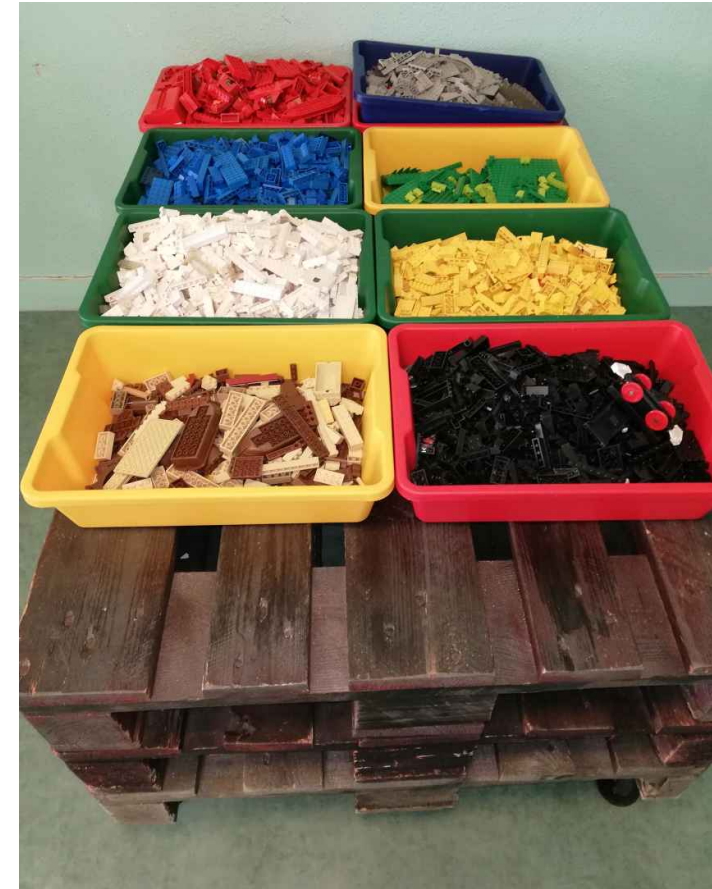
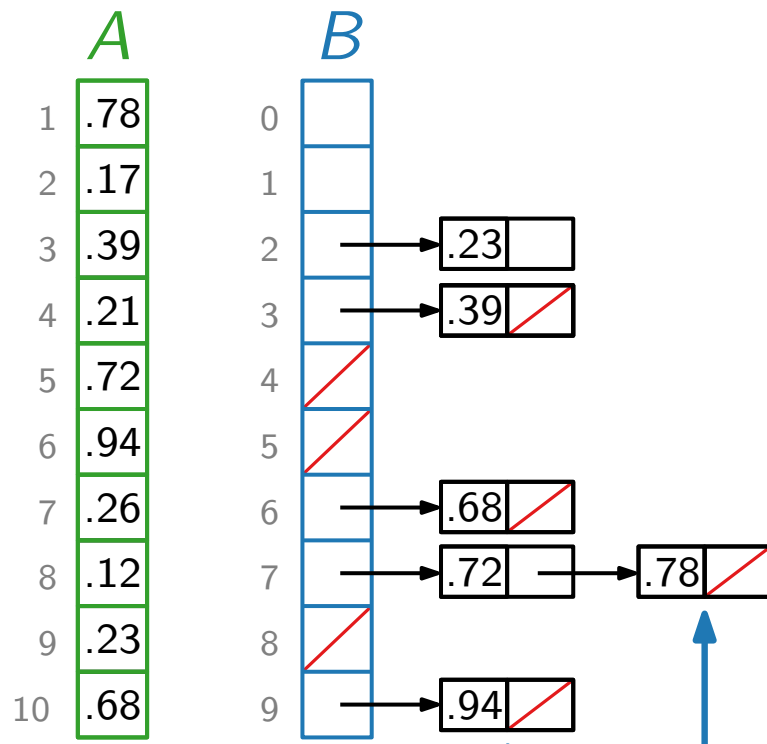


photo by JulienFou on reddit

BUCKETSORT



„Eimerinhalt“: **Verkettete Liste** von Elementen aus A .

Hilfsfeld $B[0 \dots n - 1]$;

jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen,
zufällig und gleichverteilt aus $[0, 1)$ gezogen

Im Beispielt auf 2 Nachkommastellen gerundet

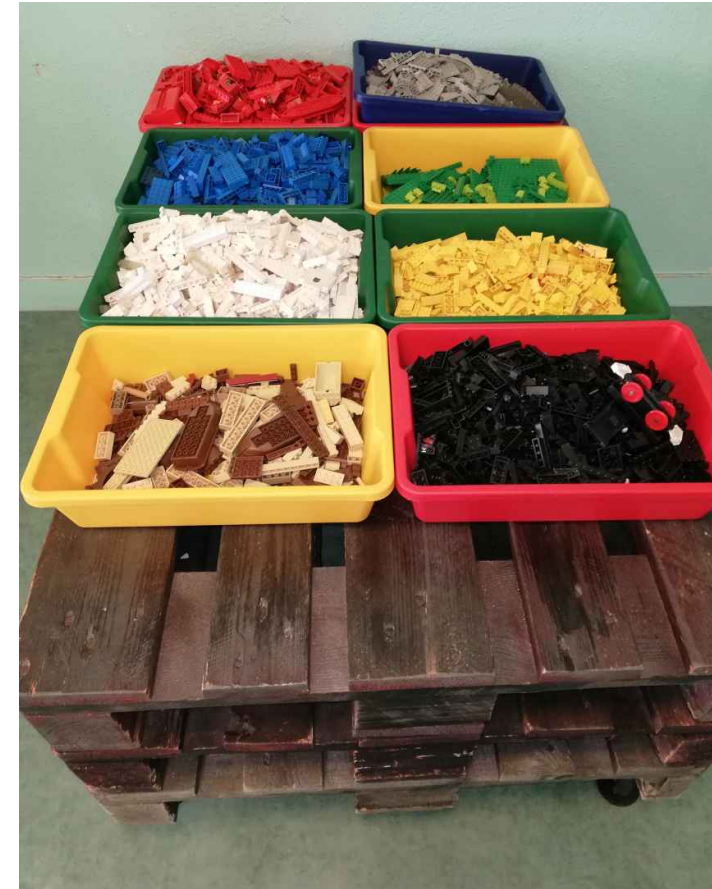
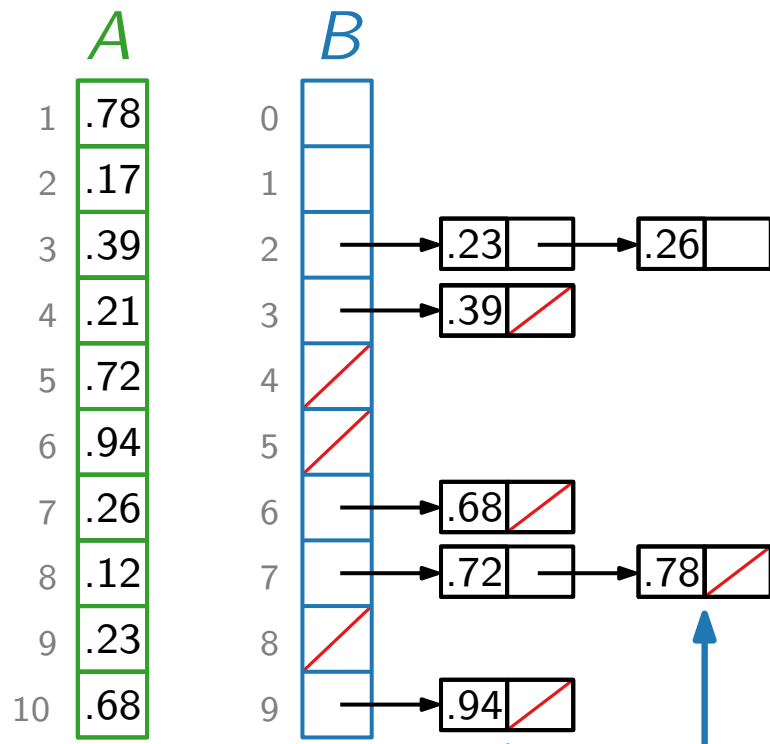


photo by JulienFou on reddit

BUCKETSORT



„Eimerinhalt“: **Verkettete Liste** von Elementen aus A .

Hilfsfeld $B[0 \dots n - 1]$;

jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen,
zufällig und gleichverteilt aus $[0, 1)$ gezogen

Im Beispielt auf 2 Nach-
kommastellen gerundet

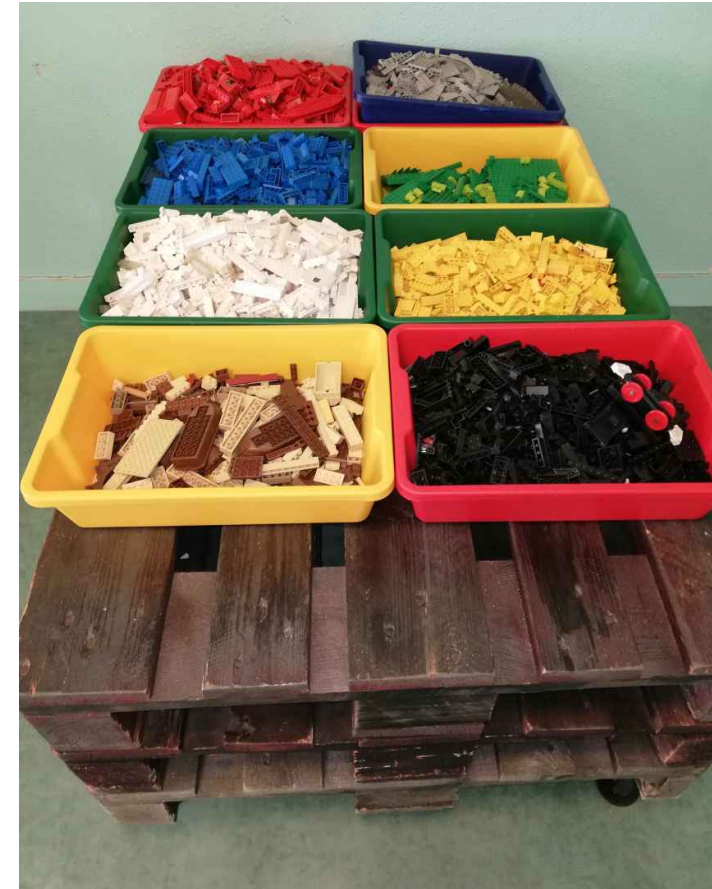
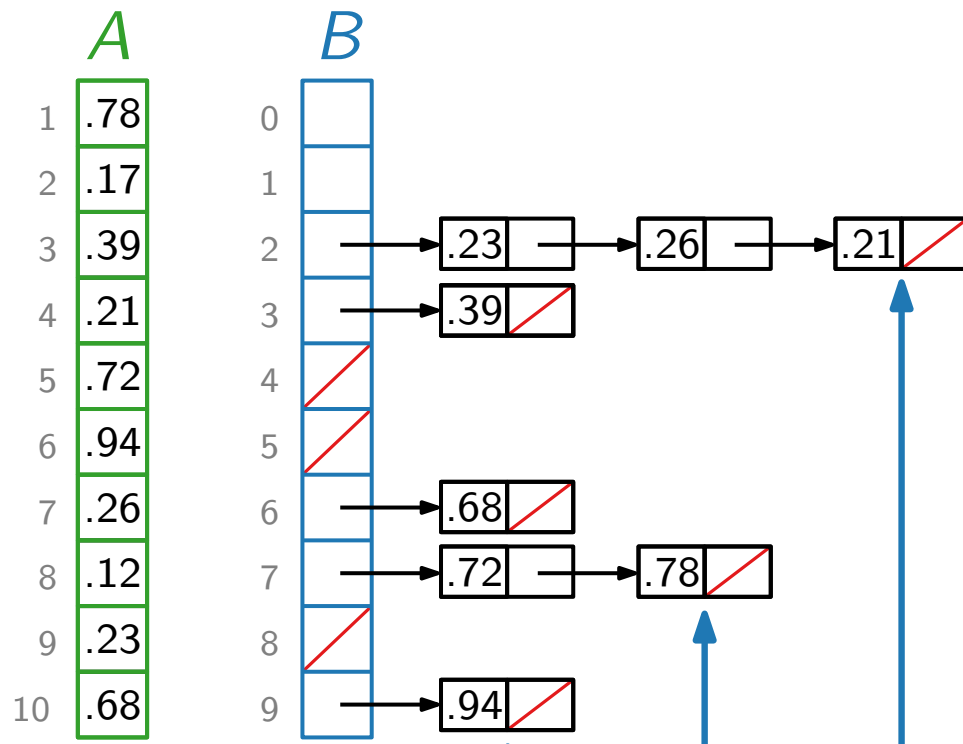


photo by JulienFou on reddit

BUCKETSORT



„Eimerinhalt“: **Verkettete Liste** von Elementen aus A .

Hilfsfeld $B[0 \dots n - 1]$;

jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen, zufällig und gleichverteilt aus $[0, 1)$ gezogen

Im Beispielt auf 2 Nachkommastellen gerundet

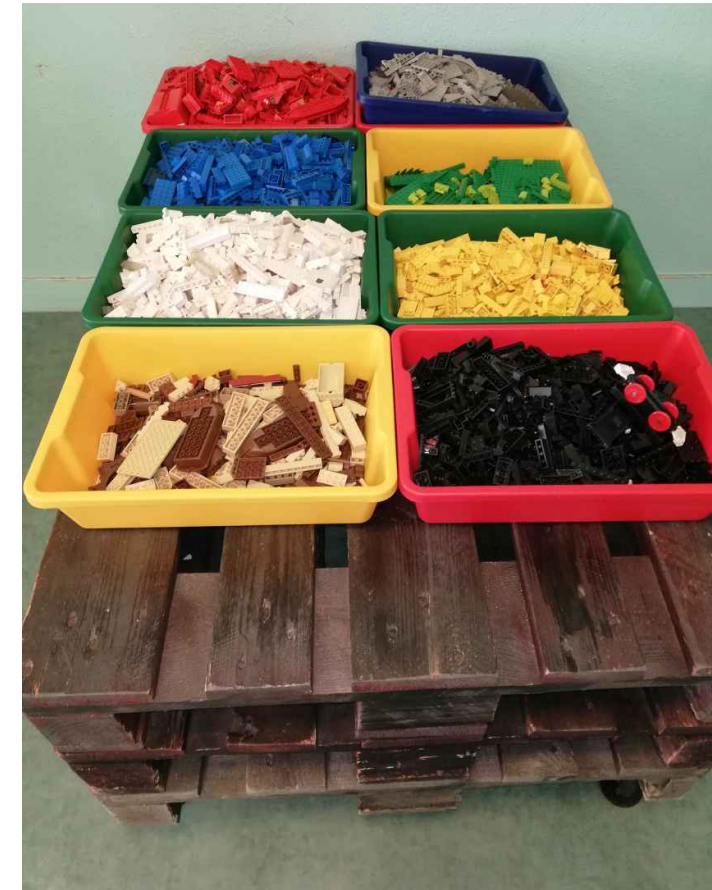
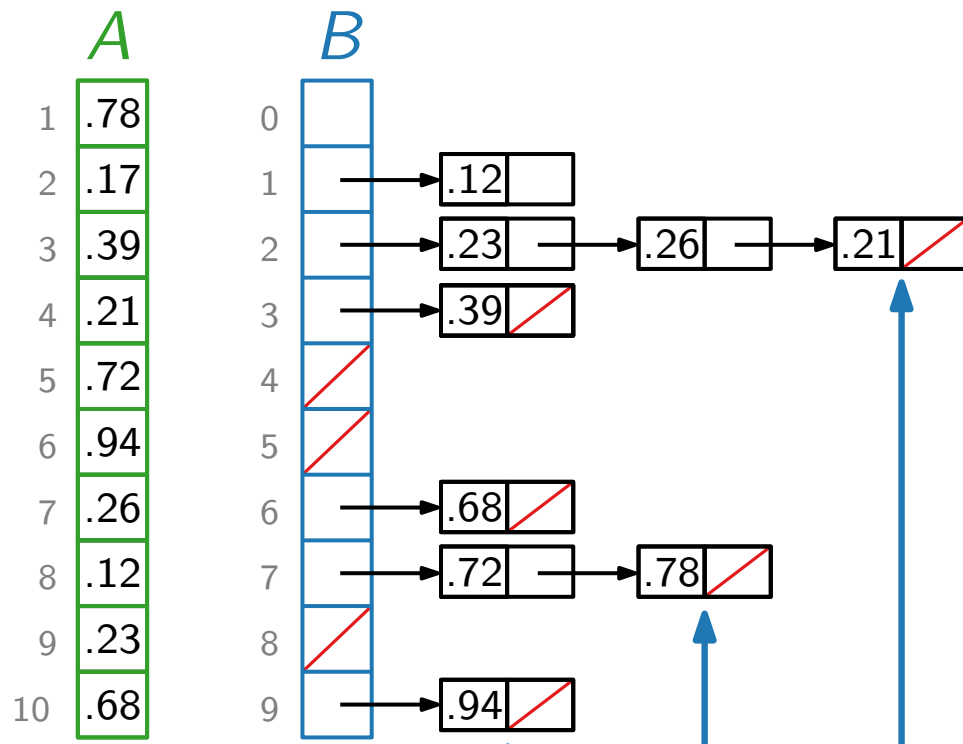


photo by JulienFou on reddit

BUCKETSORT



„Eimerinhalt“: **Verkettete Liste** von Elementen aus A .

Hilfsfeld $B[0 \dots n - 1]$;

jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen,
zufällig und gleichverteilt aus $[0, 1)$ gezogen

Im Beispielt auf 2 Nachkommastellen gerundet

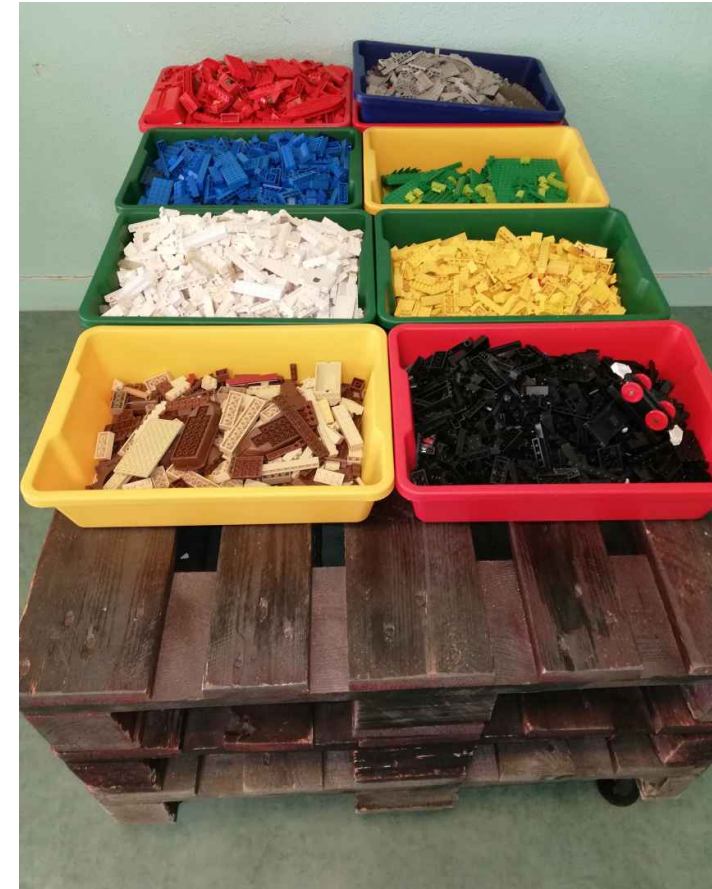
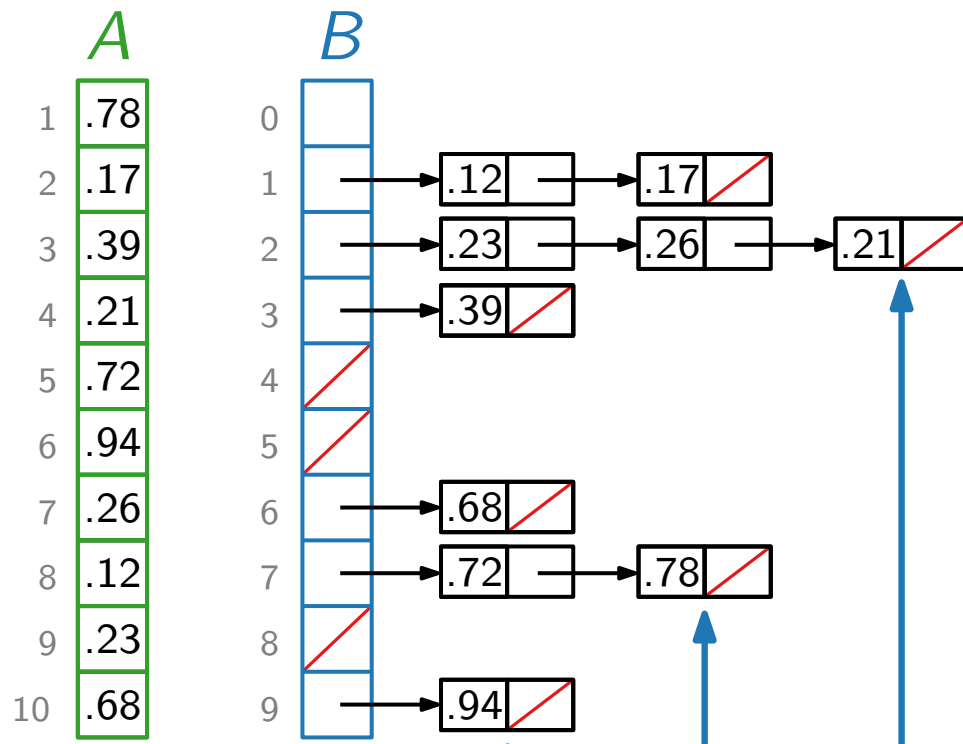


photo by JulienFou on reddit

BUCKETSORT



„Eimerinhalt“: Verkettete Liste von Elementen aus A .
 Hilfsfeld $B[0 \dots n - 1]$;
 jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen,
 zufällig und gleichverteilt aus $[0, 1)$ gezogen

Im Beispielt auf 2 Nachkommastellen gerundet

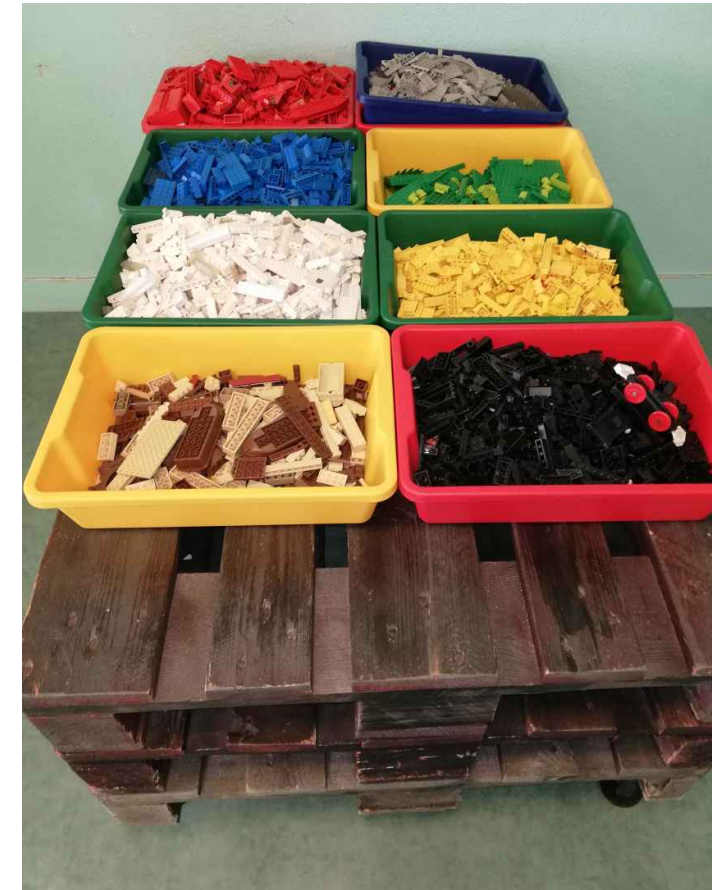
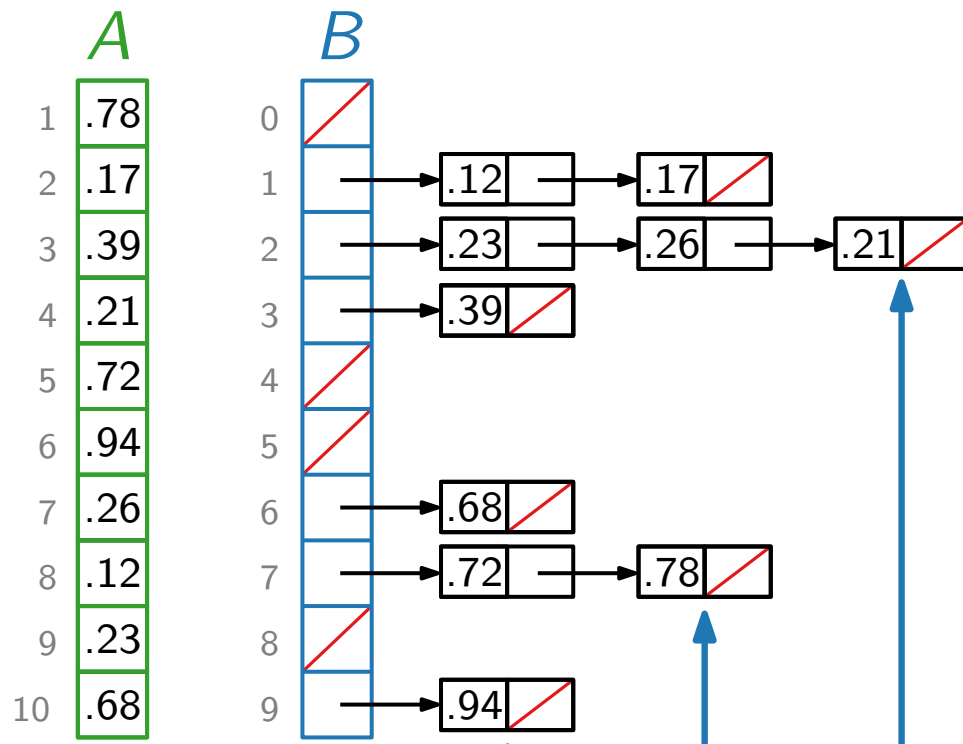


photo by JulienFou on reddit

BUCKETSORT



„Eimerinhalt“: **Verkettete Liste** von Elementen aus A .

Hilfsfeld $B[0 \dots n - 1]$;

jeder Eintrag entspricht einem „Eimer“ der Weite $1/n$

Eingabefeld $A[1 \dots n]$ enthält Zahlen, zufällig und gleichverteilt aus $[0, 1)$ gezogen

Im Beispielt auf 2 Nachkommastellen gerundet

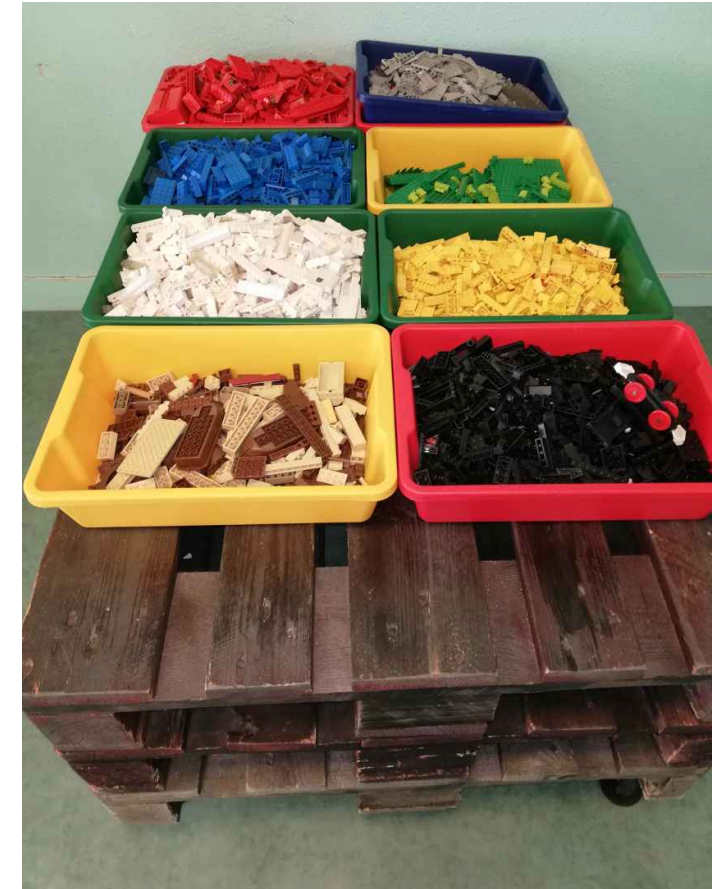
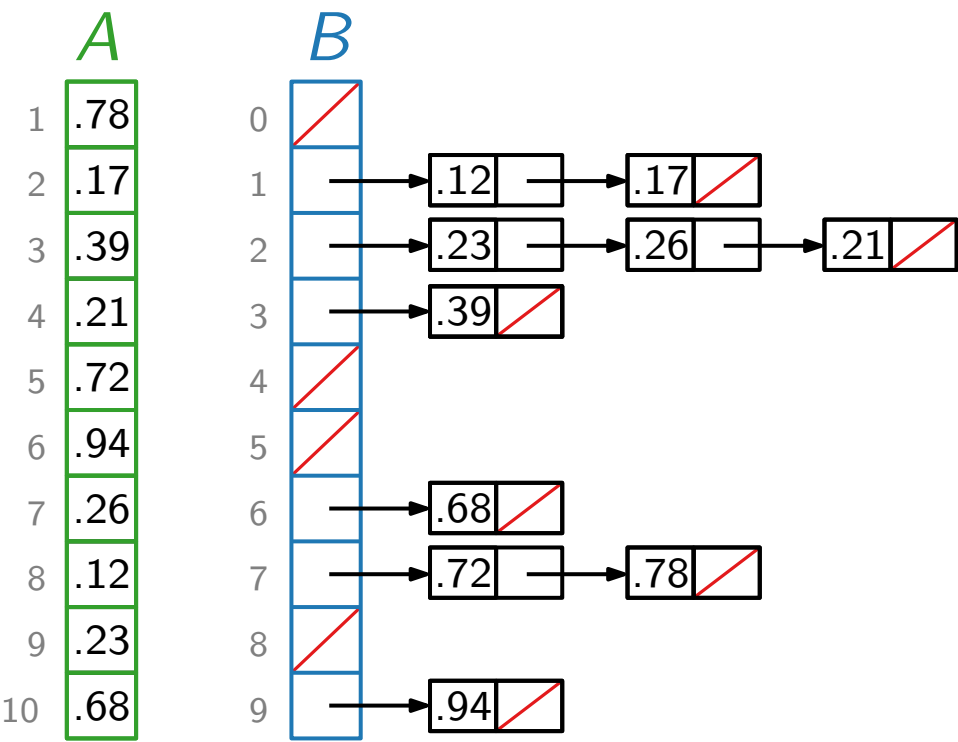


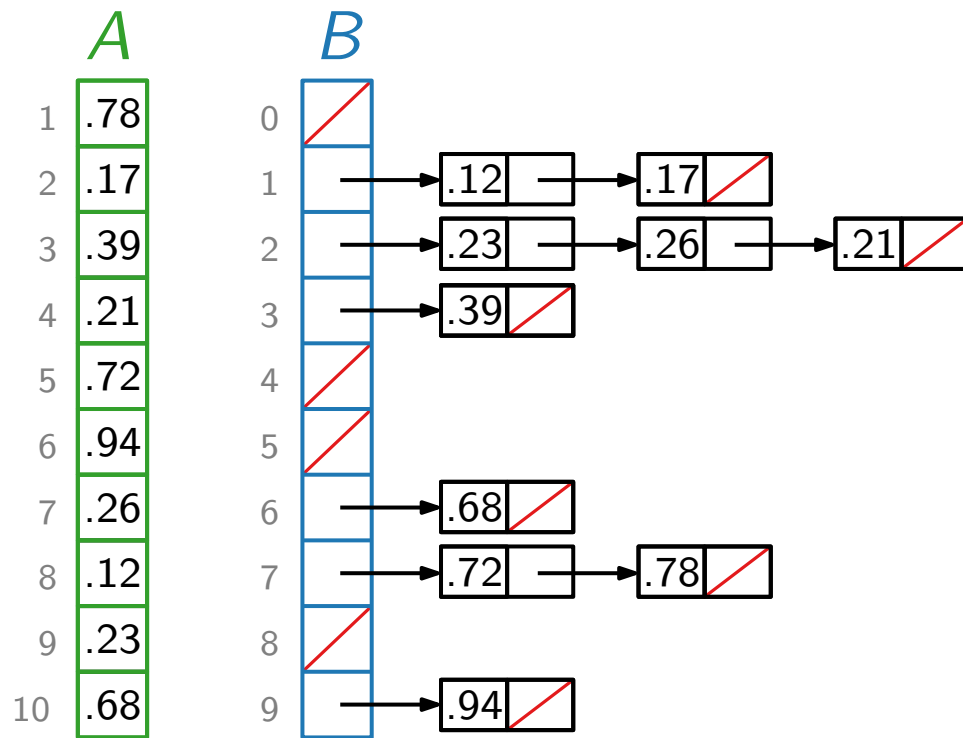
photo by JulienFou on reddit

BUCKETSORT



BUCKETSORT(Feld *A* von Zahlen in [0, 1))

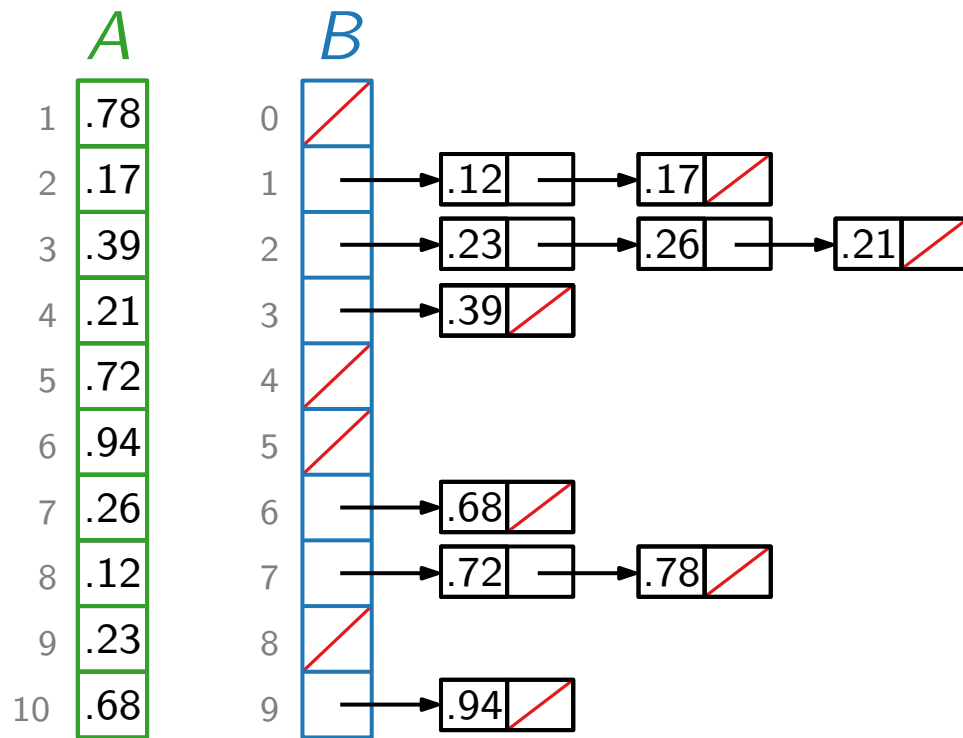
BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

BUCKETSORT

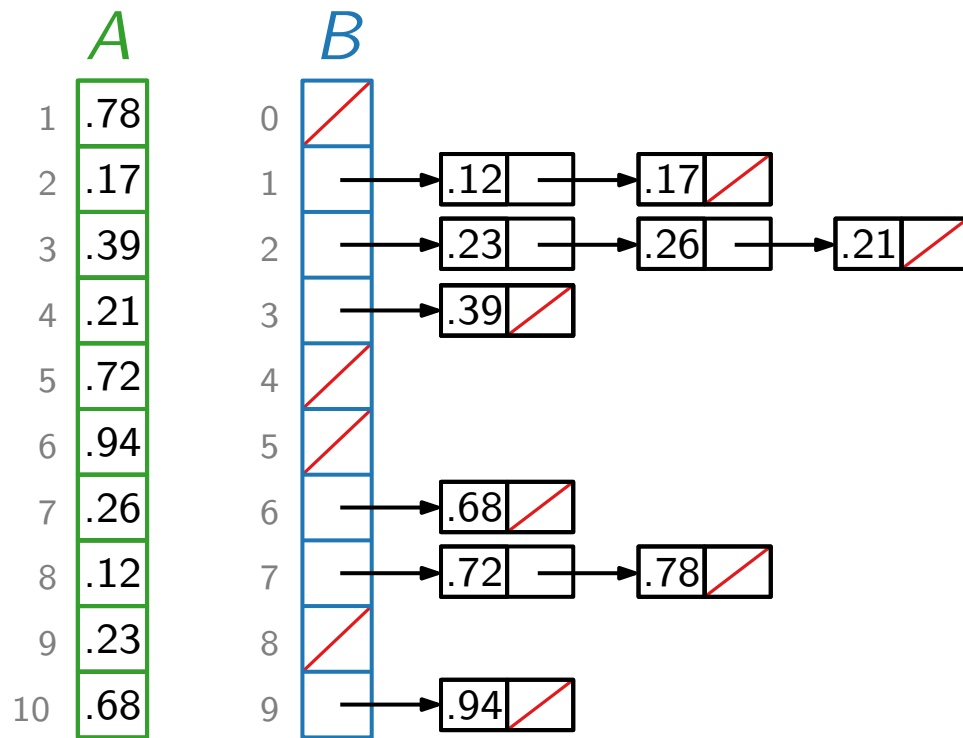


BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

BUCKETSORT



BUCKETSORT(Feld **A** von Zahlen in $[0, 1)$)

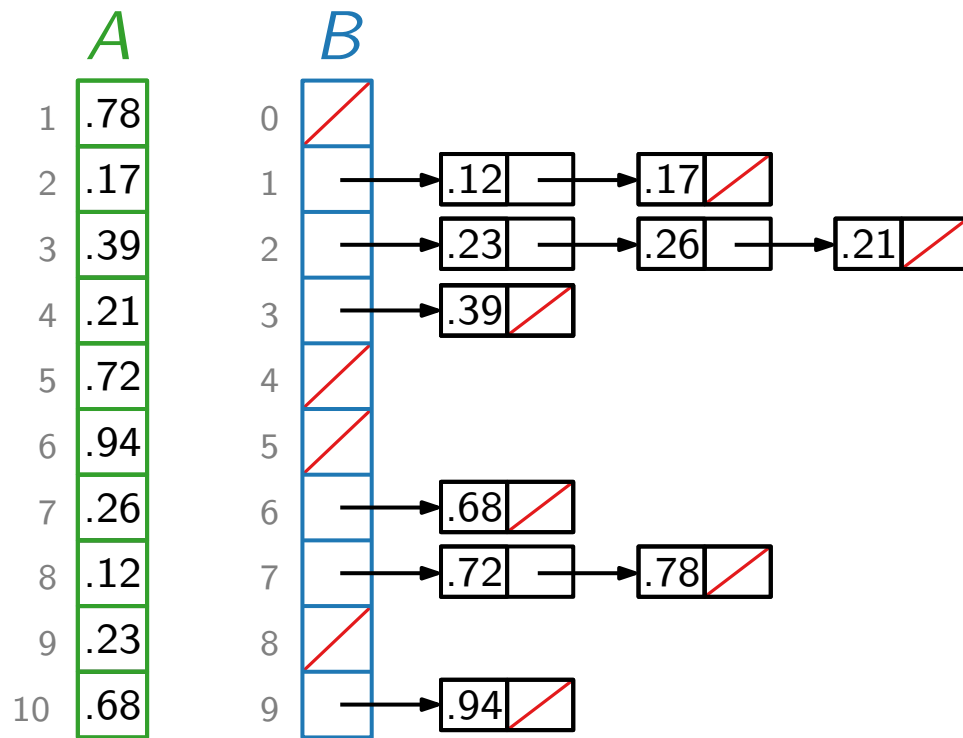
$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└

BUCKETSORT



BUCKETSORT(Feld **A** von Zahlen in $[0, 1)$)

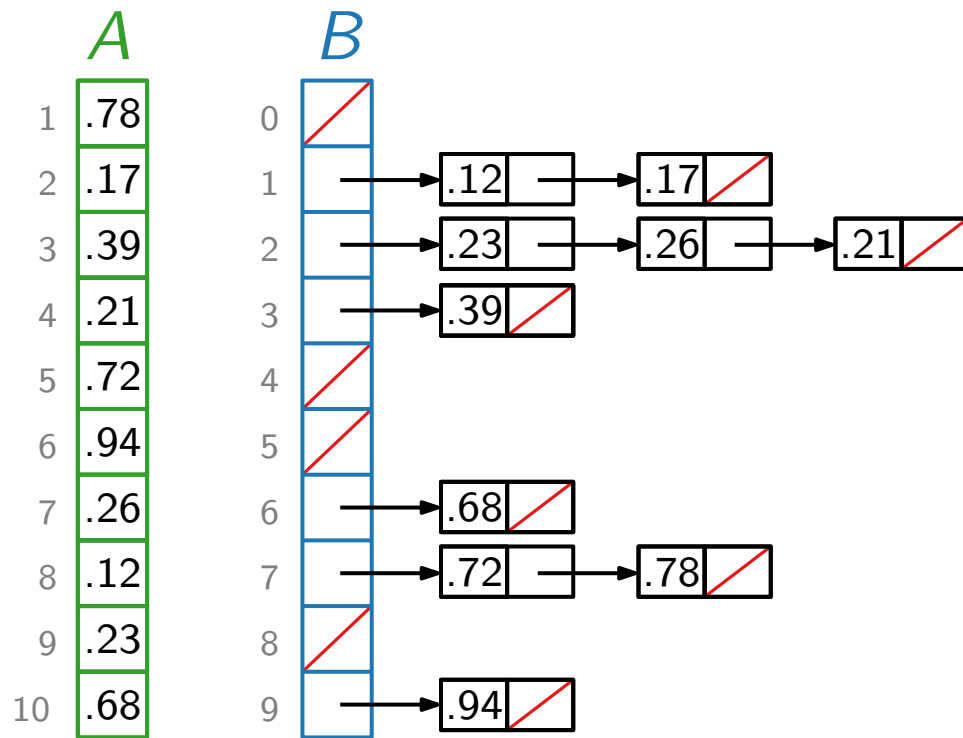
$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\text{orange}]$ ein

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

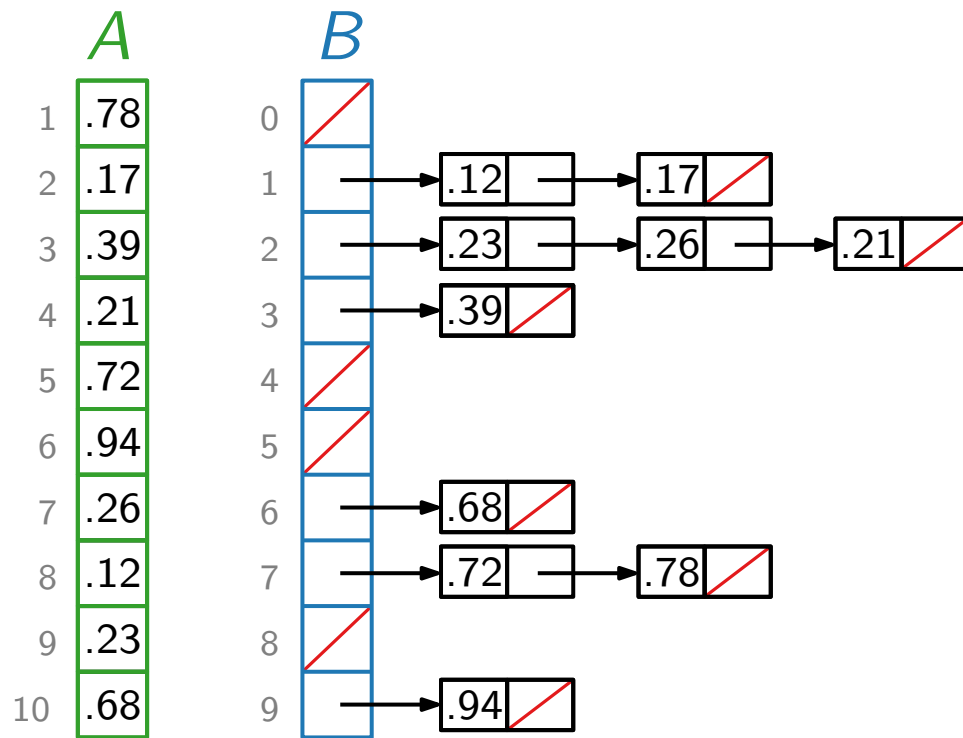
for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\text{orange}]$ ein

for $i = 0$ **to** $n - 1$ **do**

└

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

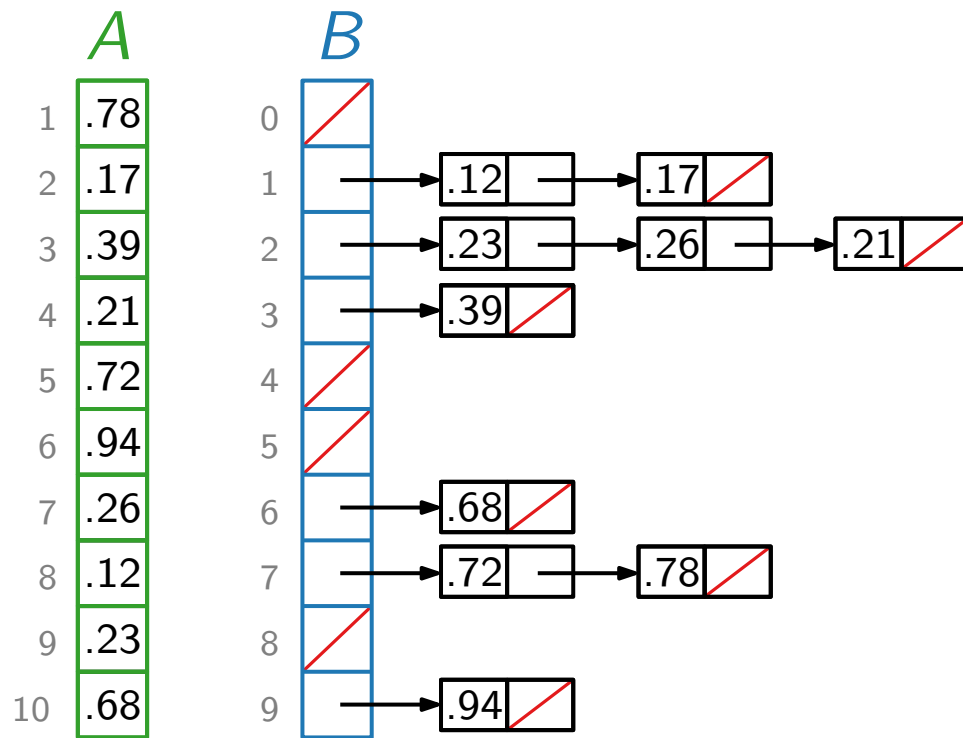
for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\text{orange}]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

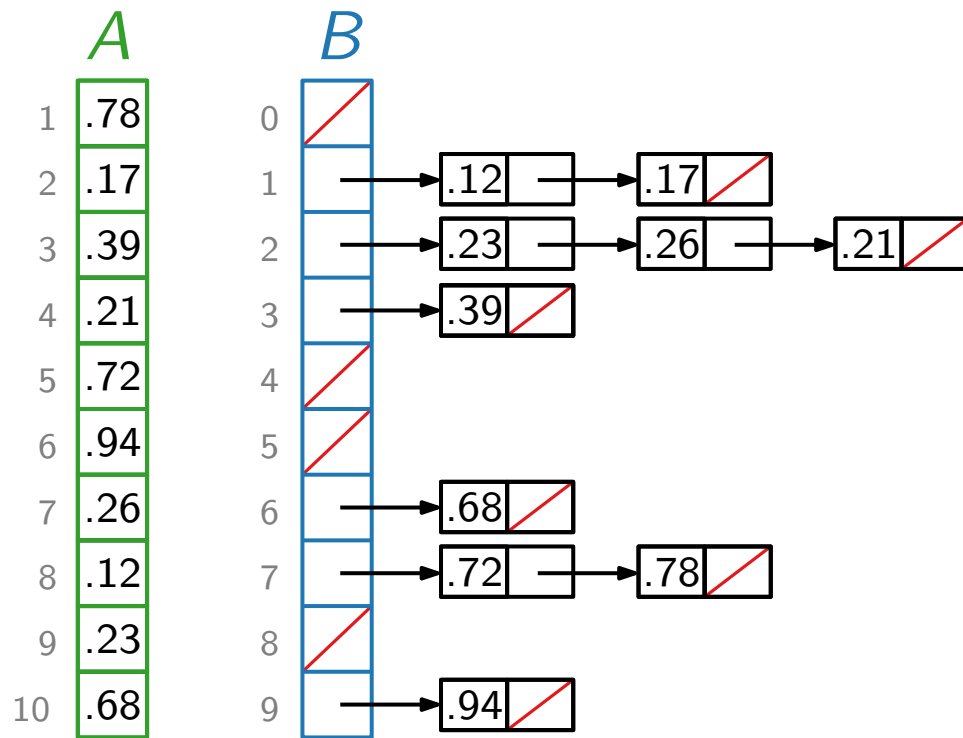
└ füge $A[j]$ in Liste $B[\text{orange}]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$



BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\text{orange}]$ ein

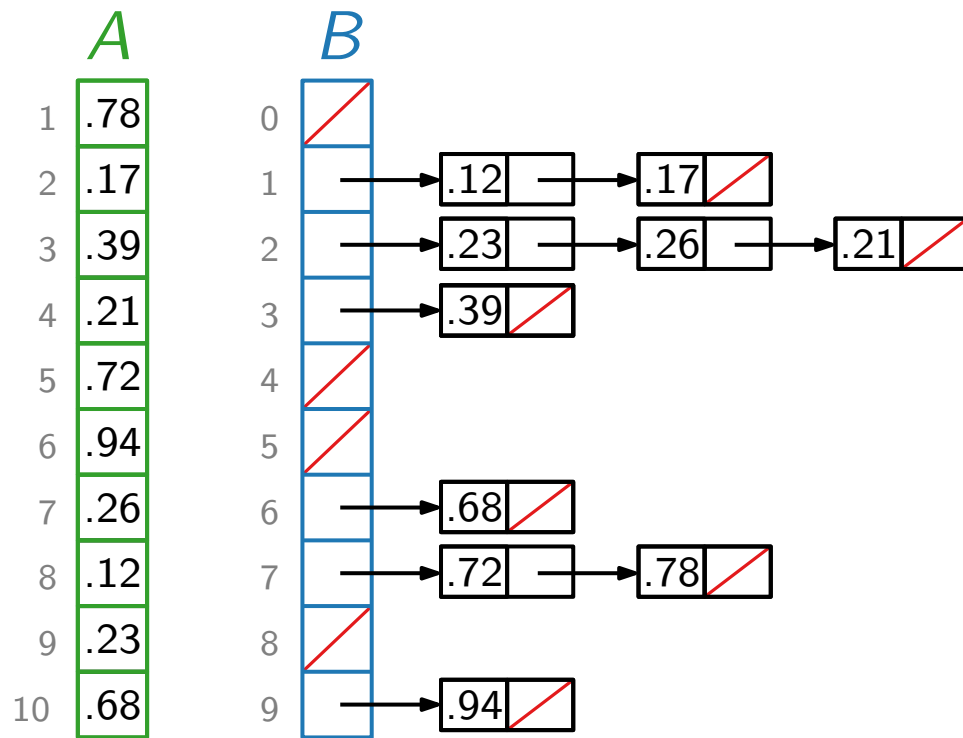
for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

Aufgabe:

Füllen Sie die Felder mit Code,
der BUCKETSORT umsetzt!

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

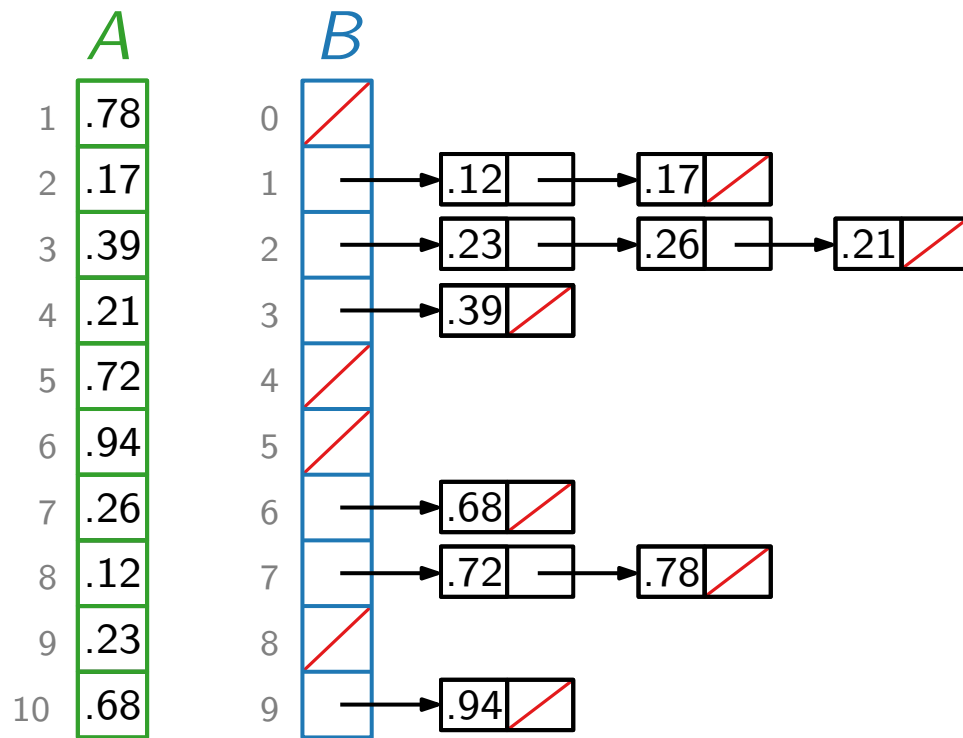
for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

Aufgabe:

Füllen Sie die Felder mit Code,
der BUCKETSORT umsetzt!

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

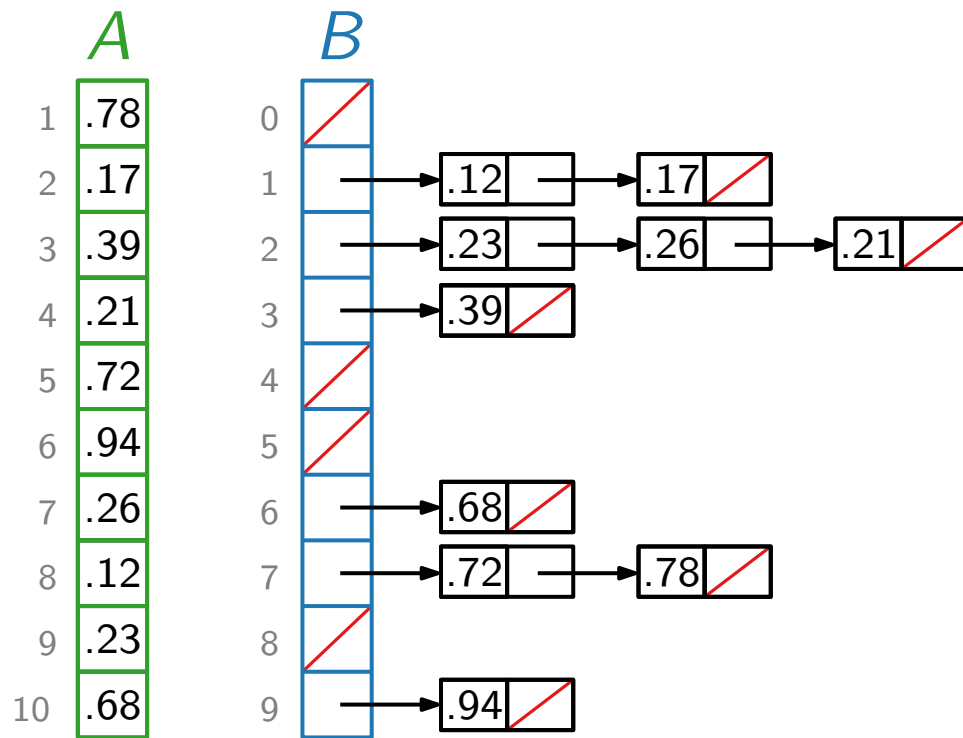
└ sortiere Liste $B[i]$

hänge $B[0], \dots, B[n - 1]$ aneinander

Aufgabe:

Füllen Sie die Felder mit Code,
der BUCKETSORT umsetzt!

BUCKETSORT



BUCKETSORT(Feld **A** von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge **A**[j] in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

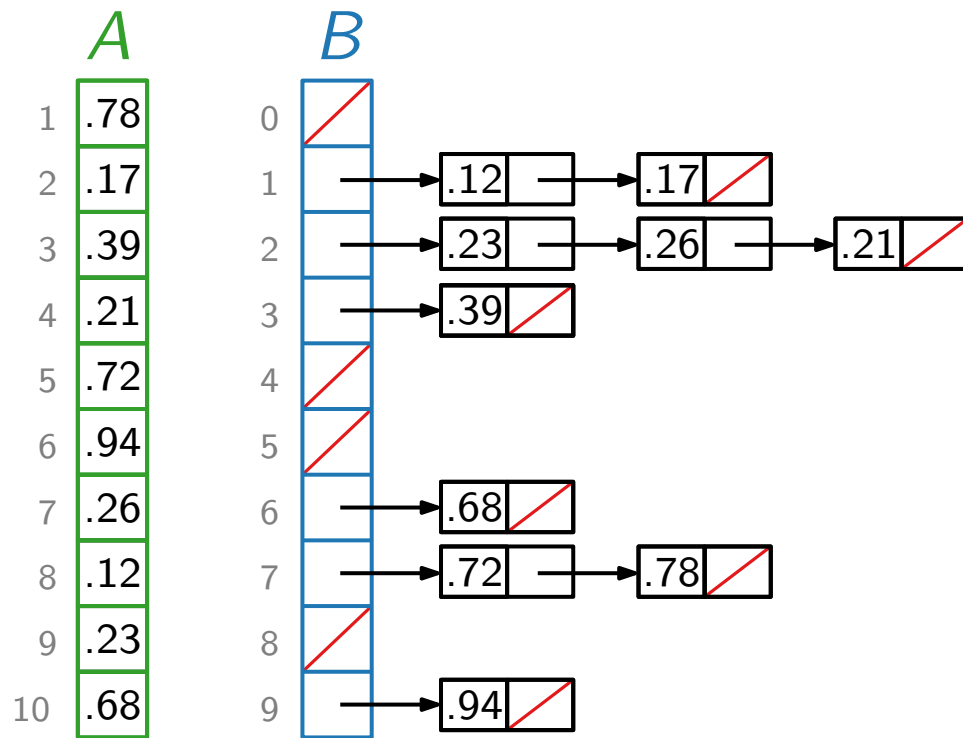
hänge $B[0], \dots, B[n - 1]$ aneinander

kopiere das Ergebnis nach **A**[$1 \dots n$]

Aufgabe:

Füllen Sie die Felder mit Code,
der BUCKETSORT umsetzt!

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

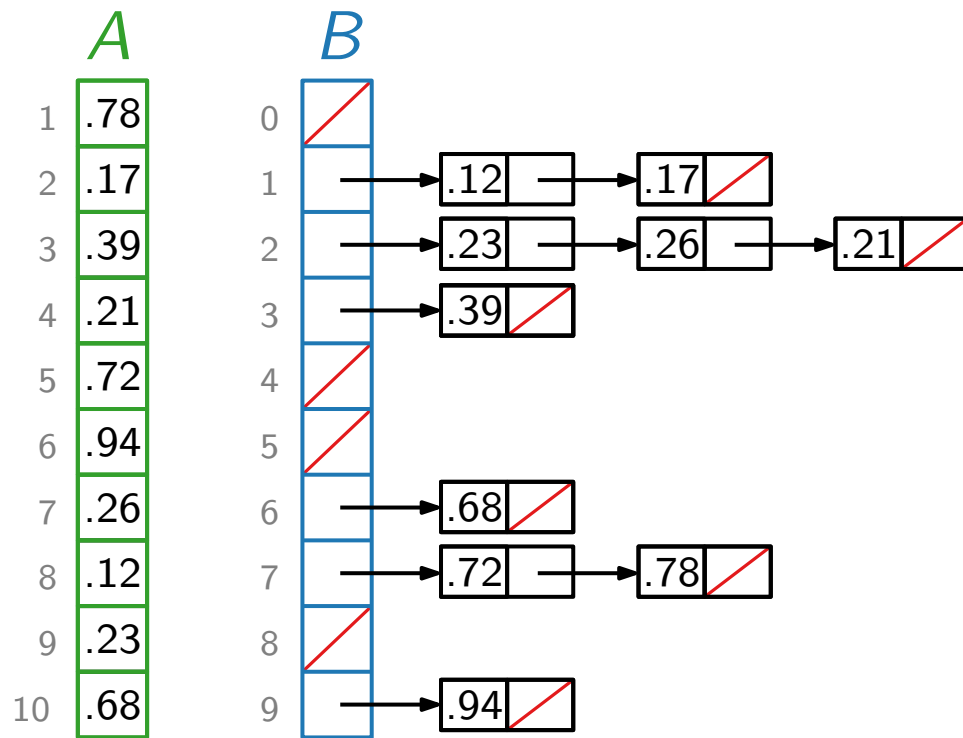
for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

hänge $B[0], \dots, B[n - 1]$ aneinander

kopiere das Ergebnis nach $A[1 \dots n]$

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

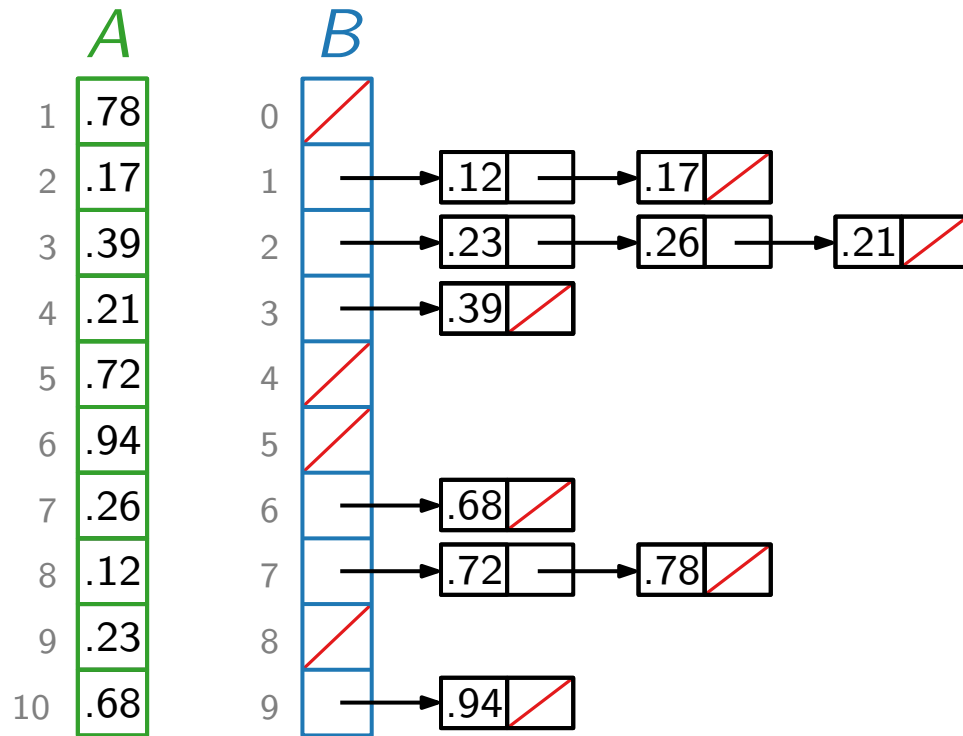
└ sortiere Liste $B[i]$

hänge $B[0], \dots, B[n - 1]$ aneinander

kopiere das Ergebnis nach $A[1 \dots n]$

$$= \left[\frac{i}{n}, \frac{i+1}{n} \right) \cap A$$

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

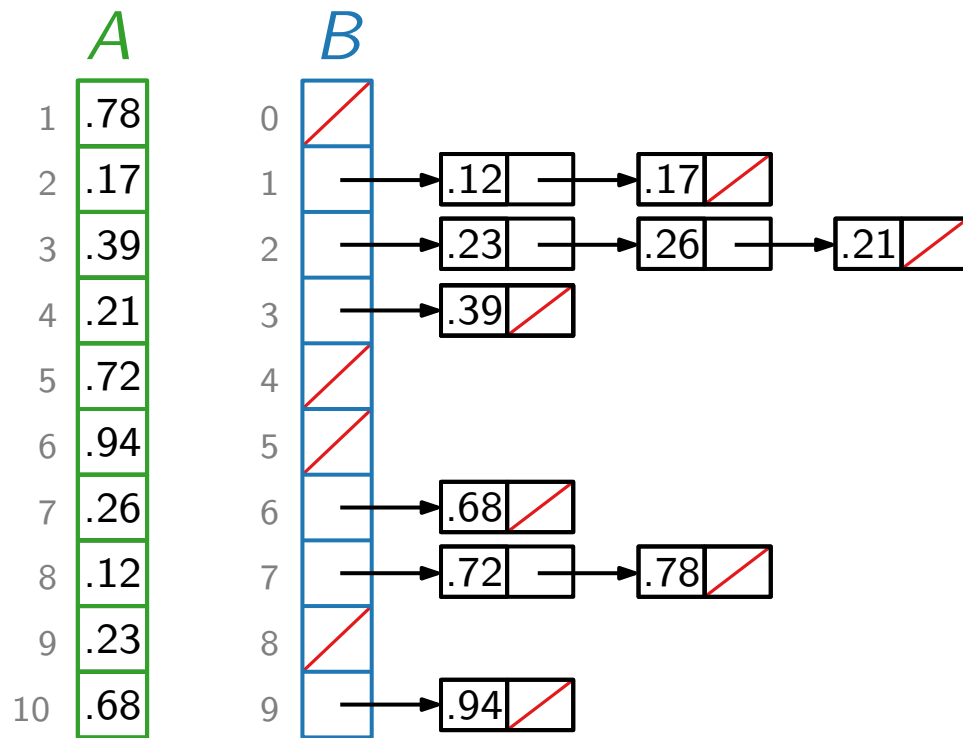
hänge $B[0], \dots, B[n - 1]$ aneinander

kopiere das Ergebnis nach $A[1 \dots n]$

$$= \left[\frac{i}{n}, \frac{i+1}{n} \right) \cap A$$

Korrektheit? 2 Fälle:

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

hänge $B[0], \dots, B[n - 1]$ aneinander

kopiere das Ergebnis nach $A[1 \dots n]$

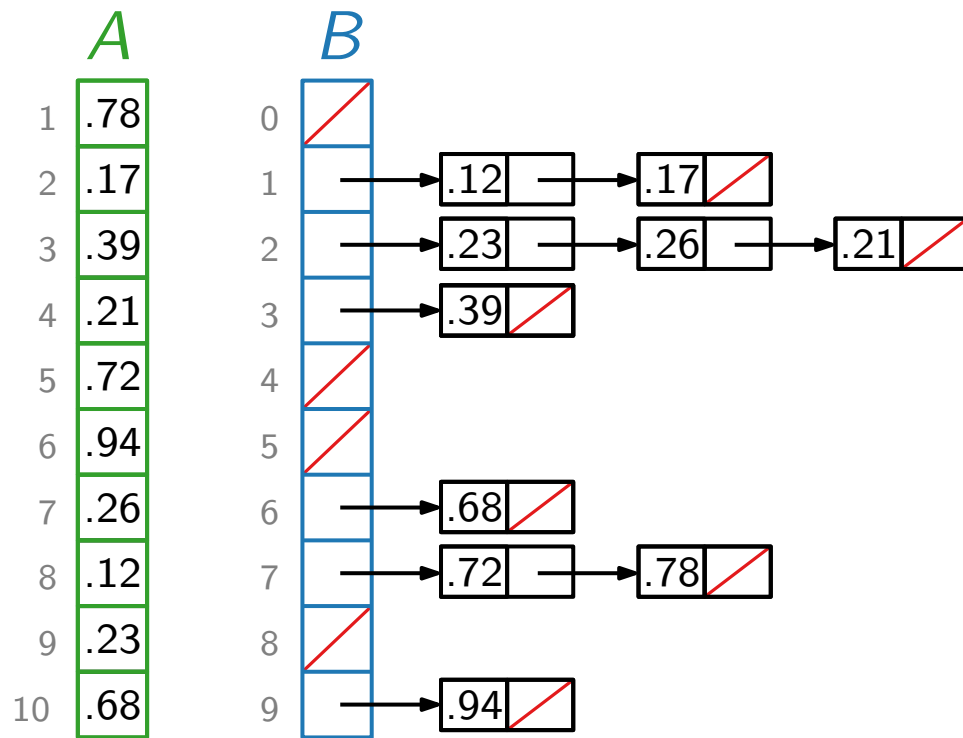
$$= \left[\frac{i}{n}, \frac{i+1}{n} \right) \cap A$$

Korrektheit?

2 Fälle:

- $A[i]$ und $A[j]$ in der gleichen Liste
- $A[i]$ und $A[j]$ in verschiedenen Listen

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

hänge $B[0], \dots, B[n - 1]$ aneinander

kopiere das Ergebnis nach $A[1 \dots n]$

$$= \left[\frac{i}{n}, \frac{i+1}{n} \right) \cap A$$

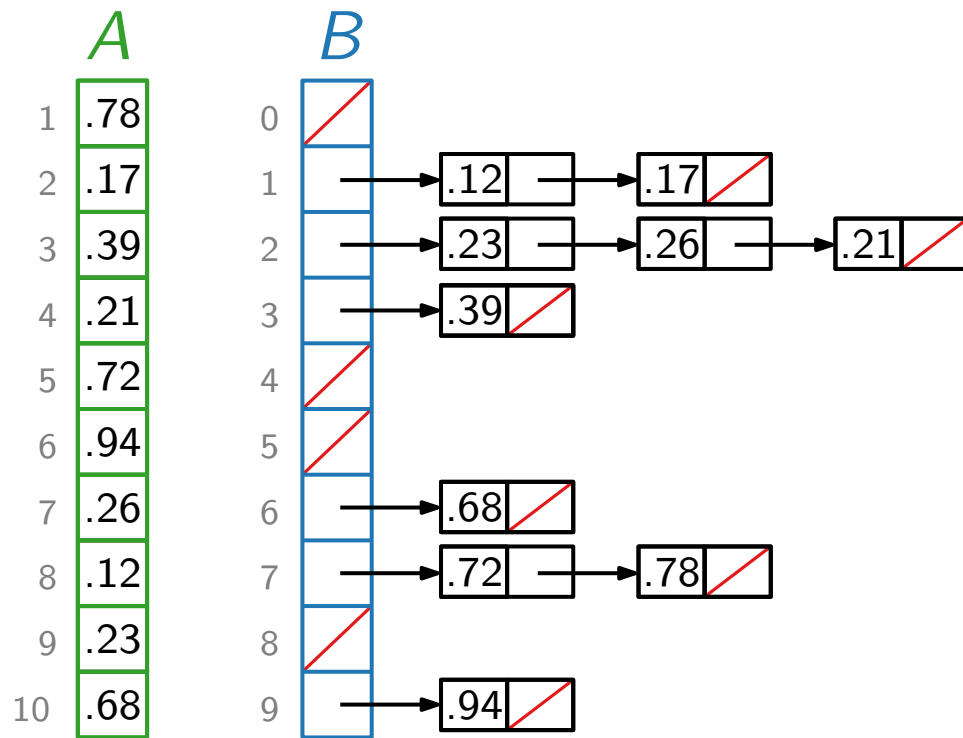
Korrektheit?

2 Fälle:

- $A[i]$ und $A[j]$ in der gleichen Liste
- $A[i]$ und $A[j]$ in verschiedenen Listen

Laufzeit?

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

hänge $B[0], \dots, B[n - 1]$ aneinander

kopiere das Ergebnis nach $A[1 \dots n]$

$$= \left[\frac{i}{n}, \frac{i+1}{n} \right) \cap A$$

Korrektheit?

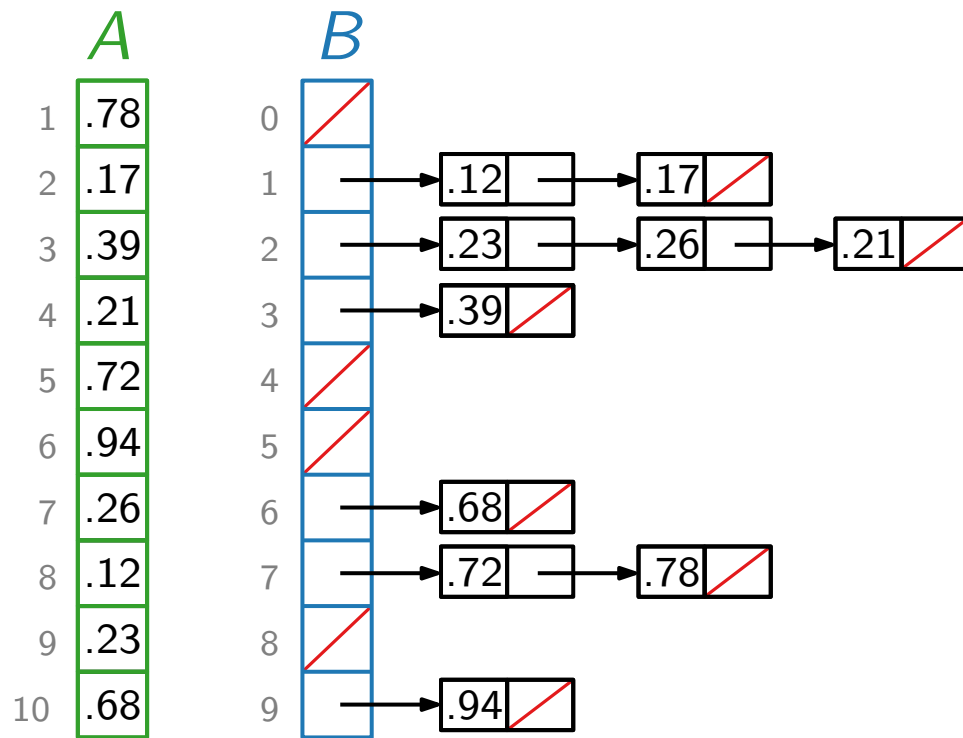
2 Fälle:

- $A[i]$ und $A[j]$ in der gleichen Liste
- $A[i]$ und $A[j]$ in verschiedenen Listen

Laufzeit?

- **erwartet**, hängt von den zufälligen Zahlen in A ab

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

hänge $B[0], \dots, B[n - 1]$ aneinander

kopiere das Ergebnis nach $A[1 \dots n]$

$$= \left[\frac{i}{n}, \frac{i+1}{n} \right) \cap A$$

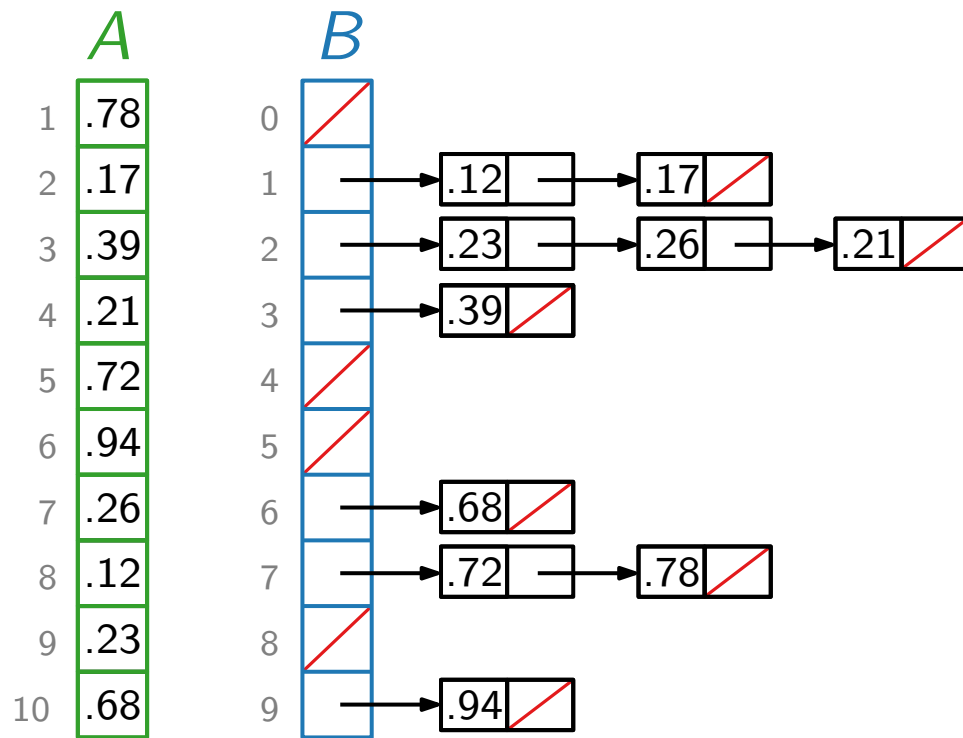
Korrektheit?

- 2 Fälle:
- $A[i]$ und $A[j]$ in der gleichen Liste
 - $A[i]$ und $A[j]$ in verschiedenen Listen

Laufzeit?

- **erwartet**, hängt von den zufälligen Zahlen in A ab
- hängt vom Sortieralgorithmus in Zeile 6 ab;

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

hänge $B[0], \dots, B[n - 1]$ aneinander

kopiere das Ergebnis nach $A[1 \dots n]$

$$= \left[\frac{i}{n}, \frac{i+1}{n} \right) \cap A$$

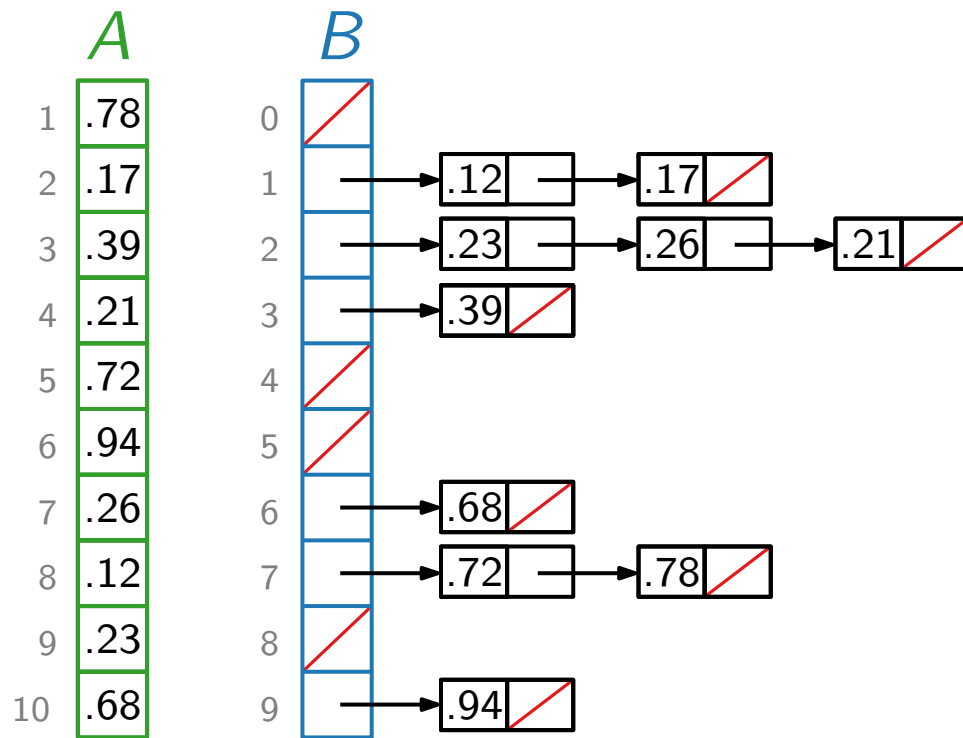
Korrektheit?

- 2 Fälle:
- $A[i]$ und $A[j]$ in der gleichen Liste
 - $A[i]$ und $A[j]$ in verschiedenen Listen

Laufzeit?

- **erwartet**, hängt von den zufälligen Zahlen in A ab
- hängt vom Sortieralgorithmus in Zeile 6 ab;
wir nehmen INSERTIONSORT:

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

hänge $B[0], \dots, B[n - 1]$ aneinander

kopiere das Ergebnis nach $A[1 \dots n]$

$$= \left[\frac{i}{n}, \frac{i+1}{n} \right) \cap A$$

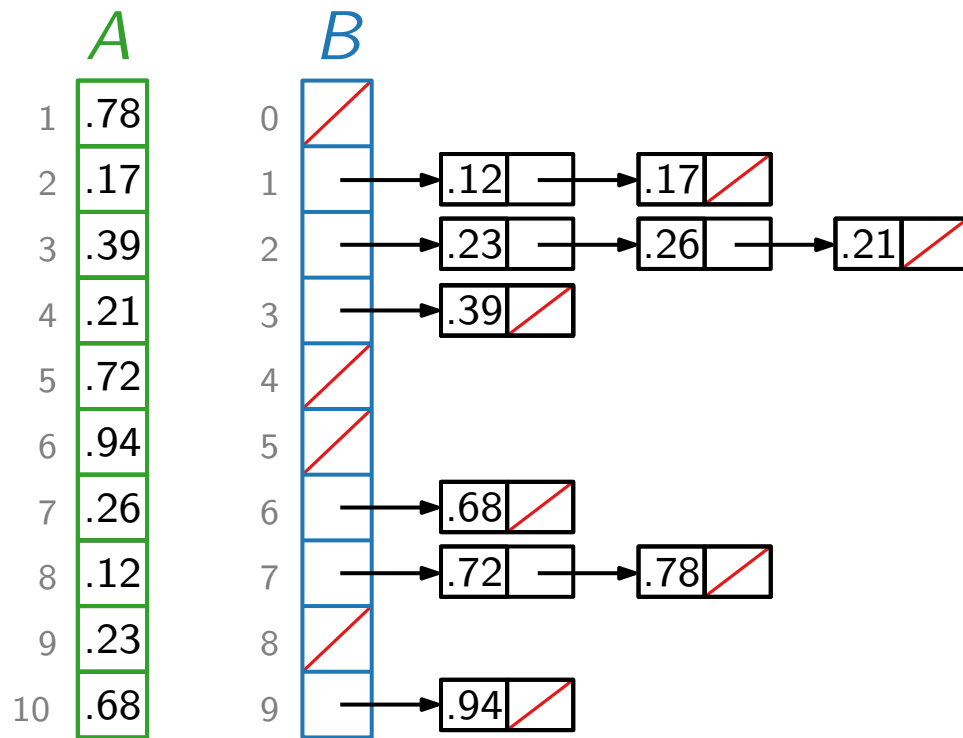
Korrektheit?

- 2 Fälle:
- $A[i]$ und $A[j]$ in der gleichen Liste
 - $A[i]$ und $A[j]$ in verschiedenen Listen

Laufzeit?

- **erwartet**, hängt von den zufälligen Zahlen in A ab
- hängt vom Sortieralgorithmus in Zeile 6 ab;
wir nehmen INSERTIONSORT: schnell auf kurzen Listen!

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

hänge $B[0], \dots, B[n - 1]$ aneinander

kopiere das Ergebnis nach $A[1 \dots n]$

Korrektheit?

2 Fälle:

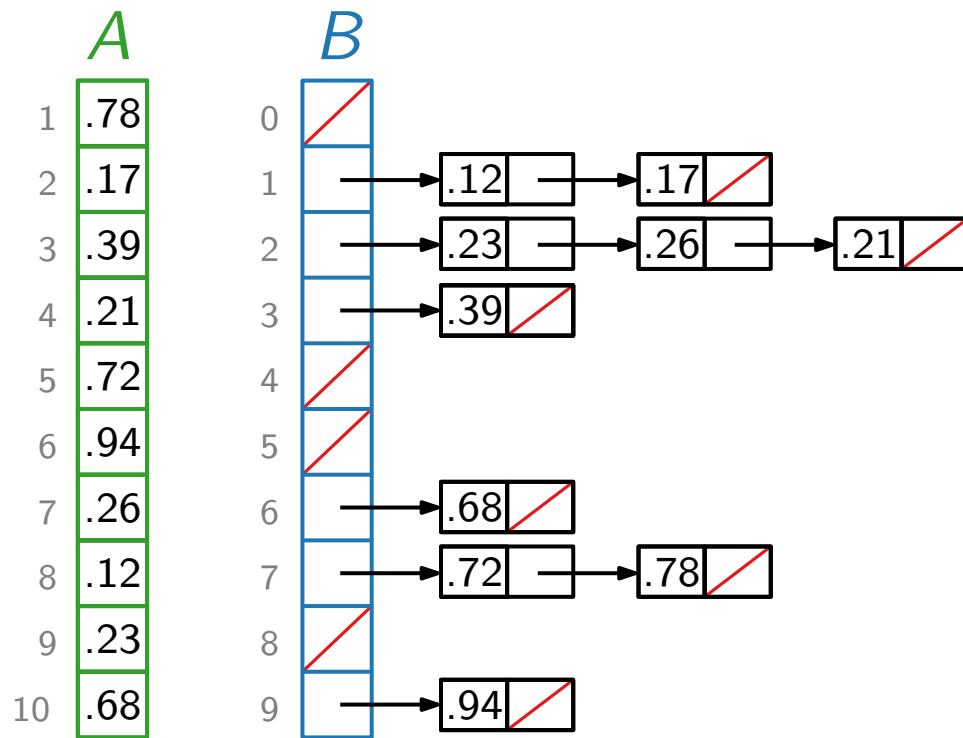
■ $A[i]$ und $A[j]$ in der gleichen Liste

■ $A[i]$ und $A[j]$ in verschiedenen Listen

Laufzeit?

$T_{BS}(n) =$

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

hänge $B[0], \dots, B[n - 1]$ aneinander

kopiere das Ergebnis nach $A[1 \dots n]$

Korrektheit?

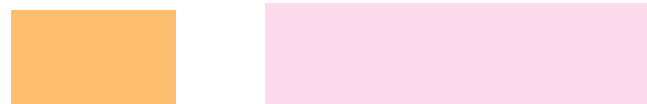
2 Fälle:

■ $A[i]$ und $A[j]$ in der gleichen Liste

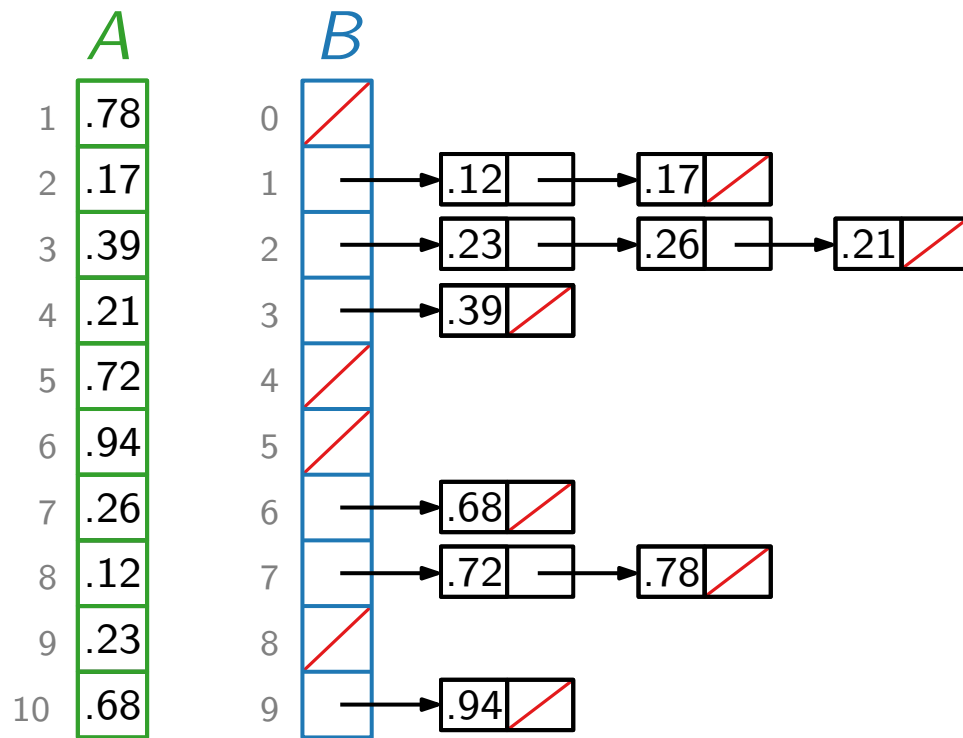
■ $A[i]$ und $A[j]$ in verschiedenen Listen

Laufzeit?

$T_{BS}(n) =$



BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

hänge $B[0], \dots, B[n - 1]$ aneinander

kopiere das Ergebnis nach $A[1 \dots n]$

Korrektheit?

2 Fälle:

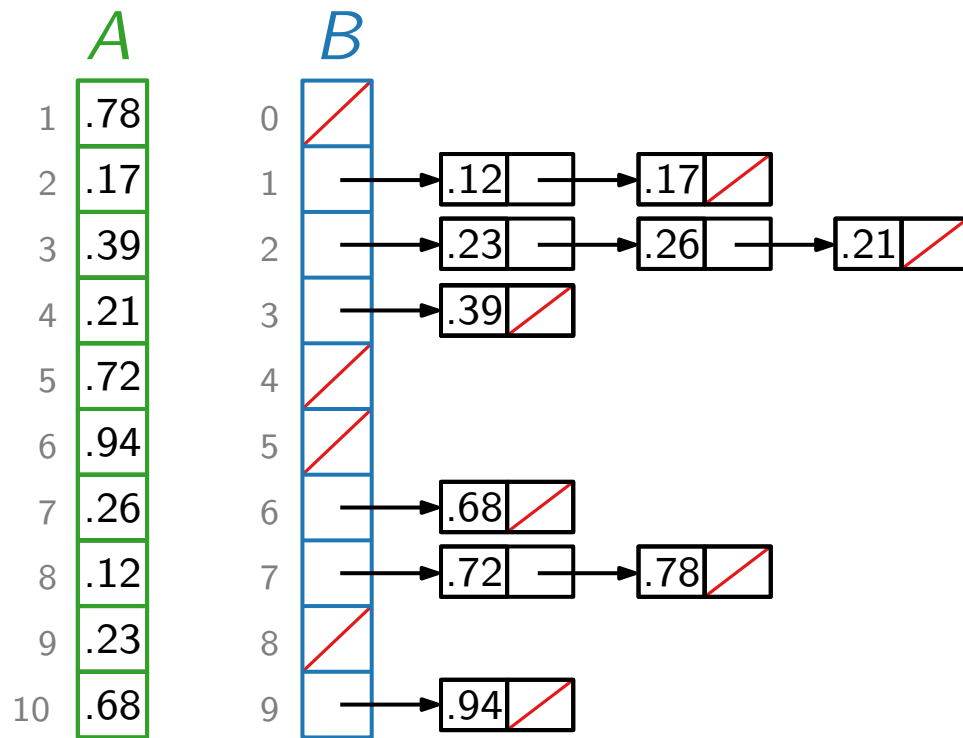
■ $A[i]$ und $A[j]$ in der gleichen Liste

■ $A[i]$ und $A[j]$ in verschiedenen Listen

Laufzeit?

$$T_{BS}(n) = \Theta(n) +$$

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

hänge $B[0], \dots, B[n - 1]$ aneinander
kopiere das Ergebnis nach $A[1 \dots n]$

Korrektheit?

2 Fälle:

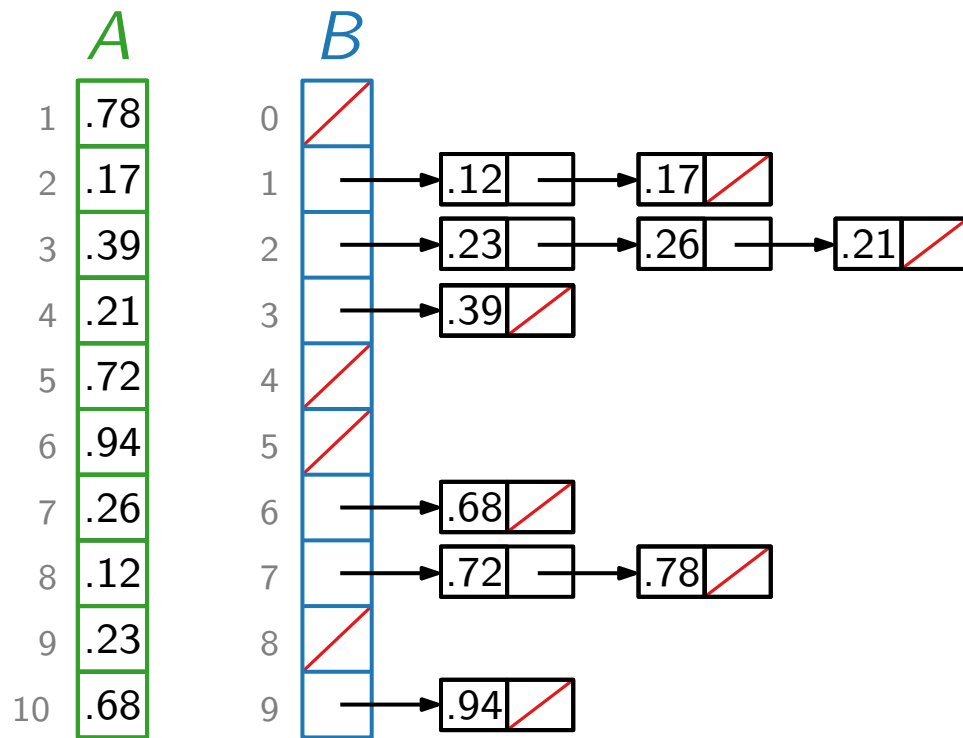
■ $A[i]$ und $A[j]$ in der gleichen Liste

■ $A[i]$ und $A[j]$ in verschiedenen Listen

Laufzeit?

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i),$$

BUCKETSORT



BUCKETSORT(Feld A von Zahlen in $[0, 1)$)

$n = A.length$

lege Feld $B[0 \dots n - 1]$ von Listen an

for $j = 1$ **to** n **do**

└ füge $A[j]$ in Liste $B[\lfloor n \cdot A[j] \rfloor]$ ein

for $i = 0$ **to** $n - 1$ **do**

└ sortiere Liste $B[i]$

hänge $B[0], \dots, B[n - 1]$ aneinander
kopiere das Ergebnis nach $A[1 \dots n]$

Korrektheit?

2 Fälle:

- $A[i]$ und $A[j]$ in der gleichen Liste
- $A[i]$ und $A[j]$ in verschiedenen Listen

Laufzeit?

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i),$$

wobei $T_{IS}(\cdot)$ Laufzeit von INSERTIONSORT

n_i Zufallsvariable für Länge der Liste $B[i]$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i)$$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] =$$

Erwartete Laufzeit von BUCKETSORT

$$T_{\text{BS}}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{\text{IS}}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$\textcolor{blue}{E}[T_{\text{BS}}(n)] = \textcolor{blue}{E}[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$\begin{aligned} E[T_{BS}(n)] &= E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)] \\ &= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)] \end{aligned}$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$\begin{aligned} E[T_{BS}(n)] &= E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)] \\ &= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)] \end{aligned}$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$\begin{aligned} E[T_{BS}(n)] &= E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)] \\ &= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)] \\ &= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) \end{aligned}$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

$$= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)]$$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

$$= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) =$$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

$$= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)]$$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

$$= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) =$$

Behauptung: $E[n_i^2] \leq 2 - \frac{1}{n}$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

$$= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)]$$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

$$= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) = \Theta(n)$$

Behauptung: $E[n_i^2] \leq 2 - \frac{1}{n}$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

$$= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)]$$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

$$= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) = \Theta(n)$$

Behauptung: $E[n_i^2] \leq 2 - \frac{1}{n}$

Beweis.

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

$$= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)]$$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

$$= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) = \Theta(n)$$

Behauptung: $E[n_i^2] \leq 2 - \frac{1}{n}$

Beweis. Definiere Indikator-Zufallsvariable $X_j := 1$, falls $A[j]$ in Eimer i fällt.

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

$$= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)]$$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

$$= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) = \Theta(n)$$

Behauptung: $E[n_i^2] \leq 2 - \frac{1}{n}$

Beweis. Definiere Indikator-Zufallsvariable $X_j := 1$, falls $A[j]$ in Eimer i fällt.

fest!

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

$$= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)]$$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

$$= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) = \Theta(n)$$

Behauptung: $E[n_i^2] \leq 2 - \frac{1}{n}$

Beweis. Definiere Indikator-Zufallsvariable $X_j := 1$, falls $A[j]$ in Eimer i fällt. fest!

$$\Rightarrow n_i =$$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

$$= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)]$$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

$$= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) = \Theta(n)$$

Behauptung: $E[n_i^2] \leq 2 - \frac{1}{n}$

Beweis. Definiere Indikator-Zufallsvariable $X_j := 1$, falls $A[j]$ in Eimer i fällt. fest!

$$\Rightarrow n_i = \sum_{j=1}^n X_j$$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

$$= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)]$$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

$$= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) = \Theta(n)$$

Behauptung: $E[n_i^2] \leq 2 - \frac{1}{n}$

Beweis. Definiere Indikator-Zufallsvariable $X_j := 1$, falls $A[j]$ in Eimer i fällt. fest!

$$\Rightarrow n_i = \sum_{j=1}^n X_j \quad E[X_j] =$$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

$$= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)]$$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

$$= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) = \Theta(n)$$

Behauptung: $E[n_i^2] \leq 2 - \frac{1}{n}$

Beweis. Definiere Indikator-Zufallsvariable $X_j := 1$, falls $A[j]$ in Eimer i fällt. fest!

$$\Rightarrow n_i = \sum_{j=1}^n X_j \quad E[X_j] = \Pr[X_j = 1] =$$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

$$= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)]$$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

$$= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) = \Theta(n)$$

Behauptung: $E[n_i^2] \leq 2 - \frac{1}{n}$

Beweis. Definiere Indikator-Zufallsvariable $X_j := 1$, falls $A[j]$ in Eimer i fällt. fest!

$$\Rightarrow n_i = \sum_{j=1}^n X_j \quad E[X_j] = \Pr[X_j = 1] = 1/n$$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

$$= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)]$$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

$$= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) = \Theta(n)$$

Behauptung: $E[n_i^2] \leq 2 - \frac{1}{n}$

Beweis. Definiere Indikator-Zufallsvariable $X_j := 1$, falls $A[j]$ in Eimer i fällt. fest!

$$\Rightarrow n_i = \sum_{j=1}^n X_j \quad E[X_j] = \Pr[X_j = 1] = 1/n$$

$$\Rightarrow n_i^2 = \left(\sum_{j=1}^n X_j \right)^2 =$$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

$$= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)]$$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

$$= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) = \Theta(n)$$

Behauptung: $E[n_i^2] \leq 2 - \frac{1}{n}$

Beweis.

Definiere Indikator-Zufallsvariable $X_j := 1$, falls $A[j]$ in Eimer i fällt. fest!

$$\Rightarrow n_i = \sum_{j=1}^n X_j \quad E[X_j] = \Pr[X_j = 1] = 1/n$$

$$\Rightarrow n_i^2 = \left(\sum_{j=1}^n X_j \right)^2 = \sum_{j=1}^n \sum_{k=1}^n X_j X_k$$

Erwartete Laufzeit von BUCKETSORT

$$T_{BS}(n) = \Theta(n) + \sum_{i=0}^{n-1} T_{IS}(n_i) = \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)$$

$$E[T_{BS}(n)] = E[\Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(n_i^2)]$$

Linearität des Erwartungswerts: $E[X + Y] = E[X] + E[Y]$

$$= \Theta(n) + \sum_{i=0}^{n-1} E[\mathcal{O}(n_i^2)]$$

Für $a \in \mathbb{R}$: $E[aX] = a \cdot E[X]$

$$= \Theta(n) + \sum_{i=0}^{n-1} \mathcal{O}(E[n_i^2]) = \Theta(n)$$

Behauptung: $E[n_i^2] \leq 2 - \frac{1}{n}$

Beweis.

Definiere Indikator-Zufallsvariable $X_j := 1$, falls $A[j]$ in Eimer i fällt. fest!

$$\Rightarrow n_i = \sum_{j=1}^n X_j \quad E[X_j] = \Pr[X_j = 1] = 1/n$$

$$\Rightarrow n_i^2 = \left(\sum_{j=1}^n X_j \right)^2 = \sum_{j=1}^n \sum_{k=1}^n X_j X_k$$

$$= \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n x_j^2 + \sum_{j=1}^n \sum_{k \neq j} x_j x_k$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n x_j^2 + \sum_{j=1}^n \sum_{k \neq j} x_j x_k$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n x_j^2 + \sum_{j=1}^n \sum_{k \neq j} x_j x_k$

$\Rightarrow E[n_i^2] =$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n x_j^2 + \sum_{j=1}^n \sum_{k \neq j} x_j x_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[x_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[x_j x_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] =$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$\begin{aligned} E[X_j^2] &= 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0] \\ &= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] \end{aligned}$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] =$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] =$$

für $j \neq k$ sind X_j und X_k unabhängig

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] = E[X_j] \cdot E[X_k]$$

für $j \neq k$ sind X_j und X_k unabhängig

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] = E[X_j] \cdot E[X_k] = \frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$$

für $j \neq k$ sind X_j und X_k unabhängig

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] = E[X_j] \cdot E[X_k] = \frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$$

für $j \neq k$ sind X_j und X_k unabhängig

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] = E[X_j] \cdot E[X_k] = \frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$$

unabhängig von j und k

für $j \neq k$ sind X_j und X_k unabhängig

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] = E[X_j] \cdot E[X_k] = \frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$$

unabhängig von j und k

für $j \neq k$ sind X_j und X_k unabhängig

Fasse die Zwischenergebnisse zusammen:

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] = E[X_j] \cdot E[X_k] = \frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$$

unabhängig von j und k

für $j \neq k$ sind X_j und X_k unabhängig

Fasse die Zwischenergebnisse zusammen:

$$E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] = E[X_j] \cdot E[X_k] = \frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$$

unabhängig von j und k

für $j \neq k$ sind X_j und X_k unabhängig

Fasse die Zwischenergebnisse zusammen:

$$E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] = E[X_j] \cdot E[X_k] = \frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$$

unabhängig von j und k

für $j \neq k$ sind X_j und X_k unabhängig

Fasse die Zwischenergebnisse zusammen:

$$E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

=

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] = E[X_j] \cdot E[X_k] = \frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$$

unabhängig von j und k

für $j \neq k$ sind X_j und X_k unabhängig

Fasse die Zwischenergebnisse zusammen:

$$E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

$$= \quad \cdot \frac{1}{n} + \quad \cdot \quad \cdot \frac{1}{n^2}$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] = E[X_j] \cdot E[X_k] = \frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$$

unabhängig von j und k

für $j \neq k$ sind X_j und X_k unabhängig

Fasse die Zwischenergebnisse zusammen:

$$E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

$$= n \cdot \frac{1}{n} + n \cdot \quad \cdot \frac{1}{n^2}$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] = E[X_j] \cdot E[X_k] = \frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$$

unabhängig von j und k

für $j \neq k$ sind X_j und X_k unabhängig

Fasse die Zwischenergebnisse zusammen:

$$E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

$$= n \cdot \frac{1}{n} + n \cdot (n-1) \cdot \frac{1}{n^2} =$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] = E[X_j] \cdot E[X_k] = \frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$$

unabhängig von j und k

für $j \neq k$ sind X_j und X_k unabhängig

Fasse die Zwischenergebnisse zusammen:

$$E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

$$= n \cdot \frac{1}{n} + n \cdot (n-1) \cdot \frac{1}{n^2} = 1 + \frac{n-1}{n} =$$

Erwartete Laufzeit von BUCKETSORT

Es gilt $n_i^2 = \sum_{j=1}^n X_j^2 + \sum_{j=1}^n \sum_{k \neq j} X_j X_k$

$$\Rightarrow E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

Behauptung:

$$E[n_i^2] \leq 2 - \frac{1}{n}$$

Behandle die beiden Typen von Erwartungswerten getrennt:

$$E[X_j^2] = 1 \cdot \Pr[X_j^2 = 1] + 0 \cdot \Pr[X_j^2 = 0]$$

$$= 1 \cdot \Pr[X_j = 1] + 0 \cdot \Pr[X_j = 0] = 1 \cdot \frac{1}{n} + 0 = \frac{1}{n}$$

unabhängig von j

$$E[X_j X_k] = E[X_j] \cdot E[X_k] = \frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n^2}$$

unabhängig von j und k

für $j \neq k$ sind X_j und X_k unabhängig

Fasse die Zwischenergebnisse zusammen:

$$E[n_i^2] = \sum_{j=1}^n E[X_j^2] + \sum_{j=1}^n \sum_{k \neq j} E[X_j X_k]$$

$$= n \cdot \frac{1}{n} + n \cdot (n-1) \cdot \frac{1}{n^2} = 1 + \frac{n-1}{n} = 2 - \frac{1}{n}$$

□

Zusammenfassung

- Jedes vergleichsbasierte Sortierverfahren braucht im Worst-Case $\Omega(n \log n)$ Vergleiche.
- COUNTINGSORT sortiert Zahlen in $\{0, \dots, k\}$. (**stabil!**)
Laufzeit für n Zahlen: $\mathcal{O}(n + k)$
- RADIXSORT sortiert s -stellige b -adische Zahlen.
Laufzeit für n Zahlen: $\mathcal{O}(s \cdot (n + b))$
- BUCKETSORT sortiert gleichverteilte zufällige Zahlen.
Erwartete Laufzeit für n Zahlen: $\mathcal{O}(n)$

Zusammenfassung

- Jedes vergleichsbasierte Sortierverfahren braucht im Worst-Case $\Omega(n \log n)$ Vergleiche.
- COUNTINGSORT sortiert Zahlen in $\{0, \dots, k\}$. (**stabil!**)
Laufzeit für n Zahlen: $\mathcal{O}(n + k)$
- RADIXSORT sortiert s -stellige b -adische Zahlen.
Laufzeit für n Zahlen: $\mathcal{O}(s \cdot (n + b))$
- BUCKETSORT sortiert gleichverteilte zufällige Zahlen.
Erwartete Laufzeit für n Zahlen: $\mathcal{O}(n)$

Bemerkung. Die Idee mit den (gleichgroßen) Eimern ist natürlich nicht nur auf Zufallszahlen beschränkt, aber hier lässt sie sich hübsch analysieren.

Vergleich Sortieralgorithmen

	Bester Fall	Erw. Fall	Schl. Fall	in-situ	stabil
INSERTIONSORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
SELECTIONSORT	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✗
BUBBLESORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
MERGESORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✗	✓
HEAPSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✓	✗
QUICKSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n^2)$	✓	✗

Vergleich Sortieralgorithmen

	Bester Fall	Erw. Fall	Schl. Fall	in-situ	stabil
INSERTIONSORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
SELECTIONSORT	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✗
BUBBLESORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
MERGESORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✗	✓
HEAPSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✓	✗
QUICKSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n^2)$	✓	✗
COUNTINGSORT					
RADIXSORT					
BUCKETSORT					

Vergleich Sortieralgorithmen

	Bester Fall	Erw. Fall	Schl. Fall	in-situ	stabil
INSERTIONSORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
SELECTIONSORT	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✗
BUBBLESORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
MERGESORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✗	✓
HEAPSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✓	✗
QUICKSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n^2)$	✓	✗
COUNTINGSORT	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$		
RADIXSORT					
BUCKETSORT					

Vergleich Sortieralgorithmen

	Bester Fall	Erw. Fall	Schl. Fall	in-situ	stabil
INSERTIONSORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
SELECTIONSORT	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✗
BUBBLESORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
MERGESORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✗	✓
HEAPSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✓	✗
QUICKSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n^2)$	✓	✗
COUNTINGSORT	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$		
RADIXSORT	$\mathcal{O}(s \cdot (n + b))$	$\mathcal{O}(s \cdot (n + b))$	$\mathcal{O}(s \cdot (n + b))$		
BUCKETSORT					

Vergleich Sortieralgorithmen

	Bester Fall	Erw. Fall	Schl. Fall	in-situ	stabil
INSERTIONSORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
SELECTIONSORT	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✗
BUBBLESORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
MERGESORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✗	✓
HEAPSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✓	✗
QUICKSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n^2)$	✓	✗
COUNTINGSORT	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$		
RADIXSORT	$\mathcal{O}(s \cdot (n + b))$	$\mathcal{O}(s \cdot (n + b))$	$\mathcal{O}(s \cdot (n + b))$		
BUCKETSORT	$\mathcal{O}(n)$	$\mathcal{O}(n)$			

Vergleich Sortieralgorithmen

	Bester Fall	Erw. Fall	Schl. Fall	in-situ	stabil
INSERTIONSORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
SELECTIONSORT	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✗
BUBBLESORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
MERGESORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✗	✓
HEAPSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✓	✗
QUICKSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n^2)$	✓	✗
COUNTINGSORT	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$		
RADIXSORT	$\mathcal{O}(s \cdot (n + b))$	$\mathcal{O}(s \cdot (n + b))$	$\mathcal{O}(s \cdot (n + b))$		
BUCKETSORT	$\mathcal{O}(n)$	$\mathcal{O}(n)$ wenn Eingabe zufällig und gleichverteilt			

Vergleich Sortieralgorithmen

	Bester Fall	Erw. Fall	Schl. Fall	in-situ	stabil
INSERTIONSORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
SELECTIONSORT	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✗
BUBBLESORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
MERGESORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✗	✓
HEAPSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✓	✗
QUICKSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n^2)$	✓	✗
COUNTINGSORT	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$		
RADIXSORT	$\mathcal{O}(s \cdot (n + b))$	$\mathcal{O}(s \cdot (n + b))$	$\mathcal{O}(s \cdot (n + b))$		
BUCKETSORT	$\mathcal{O}(n)$	$\mathcal{O}(n)$ wenn Eingabe zufällig und gleichverteilt	$\mathcal{O}(n^2)$		

Vergleich Sortieralgorithmen

	Bester Fall	Erw. Fall	Schl. Fall	in-situ	stabil
INSERTIONSORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
SELECTIONSORT	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✗
BUBBLESORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
MERGESORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✗	✓
HEAPSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✓	✗
QUICKSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n^2)$	✓	✗
COUNTINGSORT	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$	✗	
RADIXSORT	$\mathcal{O}(s \cdot (n + b))$	$\mathcal{O}(s \cdot (n + b))$	$\mathcal{O}(s \cdot (n + b))$	✗	
BUCKETSORT	$\mathcal{O}(n)$	$\mathcal{O}(n)$ wenn Eingabe zufällig und gleichverteilt	$\mathcal{O}(n^2)$	✗	

Vergleich Sortieralgorithmen

	Bester Fall	Erw. Fall	Schl. Fall	in-situ	stabil
INSERTIONSORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
SELECTIONSORT	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✗
BUBBLESORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
MERGESORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✗	✓
HEAPSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✓	✗
QUICKSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n^2)$	✓	✗
COUNTINGSORT	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$	✗	✓
RADIXSORT	$\mathcal{O}(s \cdot (n + b))$	$\mathcal{O}(s \cdot (n + b))$	$\mathcal{O}(s \cdot (n + b))$	✗	✓
BUCKETSORT	$\mathcal{O}(n)$	$\mathcal{O}(n)$ wenn Eingabe zufällig und gleichverteilt	$\mathcal{O}(n^2)$	✗	

Vergleich Sortieralgorithmen

	Bester Fall	Erw. Fall	Schl. Fall	in-situ	stabil
INSERTIONSORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
SELECTIONSORT	$\Theta(n^2)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✗
BUBBLESORT	$\Theta(n)$	$\Theta(n^2)$	$\Theta(n^2)$	✓	✓
MERGESORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✗	✓
HEAPSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n \log n)$	✓	✗
QUICKSORT	$\Theta(n \log n)$	$\Theta(n \log n)$	$\Theta(n^2)$	✓	✗
COUNTINGSORT	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$	$\mathcal{O}(n + k)$	✗	✓
RADIXSORT	$\mathcal{O}(s \cdot (n + b))$	$\mathcal{O}(s \cdot (n + b))$	$\mathcal{O}(s \cdot (n + b))$	✗	✓
BUCKETSORT	$\mathcal{O}(n)$	$\mathcal{O}(n)$ wenn Eingabe zufällig und gleichverteilt	$\mathcal{O}(n^2)$	✗	✓ wenn verwendeter Sortieralg. stabil