

# Algorithmen 4

Gerald Futschek

# Phasen der Problemlösung mit Programmen

- **Problemanalyse:** **Was** soll getan werden?  
→ Spezifikation
- **Lösungsdesign:** **Wie** kommt man zur Lösung?  
→ Algorithmus
- **Implementierung:** Realisierung in einer Programmiersprache  
→ Programm
- **Testen:** hat der Algorithmus / das Programm Fehler?  
→ Testprotokoll

# Beispiel: Suchen in einem großen Datenbestand 1

- **Problemanalyse**

- Datenbestand besteht aus vielen gleichgestaltigen Einzeldaten
- Fragestellung: Kommt ein gegebener Wert  $x$  im Datenbestand vor oder nicht?
- Ergebnis ist „wahr“ oder „falsch“
- Es gibt keine Sortierung der Daten

- **Lösungsdesign**

- Vergleiche der Reihe nach alle Werte des Datenbestandes mit Wert  $x$  (sequentielle/lineare Suche)
- ist mindestens einer gleich  $x$ , ist Ergebnis „wahr“
- ist keiner mit  $x$  gleich, ist Ergebnis „falsch“

# Beispiel: Suchen in einem großen Datenbestand 2

## Fragen zur Implementierung:

- In welcher Reihenfolge sollen die Daten verglichen werden?
- Gibt es bessere / schlechtere Reihenfolgen?
- Gibt es überhaupt immer eine Reihenfolge?
- Gibt es immer ein Ergebnis?
- Was ist die Precondition?

# Beispiel: Suchen in einem großen Datenbestand 3

Lineare Suche ( $\downarrow x$ ,  $\uparrow$  gefunden):

$y \leftarrow$  erster Wert

gefunden  $\leftarrow$  „falsch“

**solange** (noch ein weiterer Wert existiert)

falls  $(y = x)$  gefunden  $\leftarrow$  „wahr“

$y \leftarrow$  nächster Wert

- mit welchen Eingangswerten (Testdaten) soll man den Algorithmus testen?
- kein, ein, mehrere Werte im Datenbestand
- gesuchter Wert  $x$  am Anfang, in der Mitte, am Ende
- gesuchter Wert  $x$  nicht enthalten

# Beispiel: Suchen in einem großen Datenbestand 4 (Verbesserte Version)

Lineare Suche ( $\downarrow x$ ,  $\uparrow$  gefunden):

$y \leftarrow$  erster Wert

**falls**  $(y = x)$  gefunden  $\leftarrow$  „wahr“

**sonst** gefunden  $\leftarrow$  „falsch“

**solange** (noch ein weiterer Wert existiert)

$y \leftarrow$  nächster Wert

**falls**  $(y = x)$  gefunden  $\leftarrow$  „wahr“

- Beobachtung: gefunden enthält stets das Ergebnis für den bereits untersuchten Datenbestand
- Der bereits untersuchte Datenbestand wächst in jedem Schritt um einen Wert an (allgemeine Strategie!)
- Was ist der Aufwand des Verfahrens?
- Kann man das Verfahren beschleunigen?

# Optimierte Lineare Suche

**Lineare Suche 2 ( $\downarrow x$ ,  $\uparrow$ gefunden):**

$y \leftarrow$  erster Wert

**solange** ( $y \neq x$ ) und (noch ein weiterer Wert existiert)

$y \leftarrow$  nächster Wert

**falls** ( $y = x$ )

gefunden  $\leftarrow$  „wahr“

**sonst**

gefunden  $\leftarrow$  „falsch“

- In welchen Fällen ist die optimierte Version besser geworden?
- In welchen Fällen ist die optimierte Version gleich gut?
- In welchen Fällen ist die optimierte Version schlechter geworden?

# Beispiel: größte Wert in einem großen Datenbestand

- Aufgabenanalyse
  - existiert immer ein größter Wert?
  - was ist der größte Wert von (rot gelb grün) ?
- Lösungsdesign
  - Allgemeine Strategie der schrittweisen Vergrößerung des untersuchten Datenbestandes anwenden
  - bei einem einelementigen Datenbestand ist der einzige Wert auch der größte (Anfangssituation)
  - Wenn noch ein Wert hinzukommt, ist er der größte, wenn er größer als der bisher größte Wert ist.

# Beispiel: größte Wert in einem großen Datenbestand 2

Maximale Element ( $\uparrow$ max):

$y \leftarrow$  erster Wert

$\text{max} \leftarrow y$

**solange** (noch ein weiterer Wert existiert)

$y \leftarrow$  nächster Wert

**falls** ( $y > \text{max}$ )  $\text{max} \leftarrow y$

- Welche Bedeutung hat Variable max?
- Was ist der Aufwand des Verfahrens?
- Kann man das Verfahren beschleunigen?