

Lernplan: Algorithmen & Datenstrukturen

Ziel: Vorbereitung auf den Test am Dienstag (18.11.25) durch fokussiertes Lernen an drei Tagen (Sa, So, Mo) in jeweils unter 30 Minuten, plus eine schnelle Wiederholung am Testtag.

Legende der Quellen

- **PDF 1:** Folien 38-49 (Verifikation, Halteproblem, Programmierparadigmen, Mergesort)
- **PDF 2:** Folien 99-140 (Sortierproblem, Bäume, Counting Sort, Bucket Sort, Radix Sort)
- **PDF 3:** Folien 51-97 (Einführung, einfache und komplexe Sortierverfahren, Quicksort)
- **PDF 4:** Folien 1-34 (Einführung, Laufzeitanalyse, Komplexität)
- **Notizen:** Deine handschriftlichen Mitschriften.

Tag 1 (Samstag): Sortieralgorithmen verstehen & klassifizieren

(ca. 30 Minuten)

Fokus: Den Unterschied zwischen vergleichsbasierten und nicht-vergleichsbasierten Algorithmen verstehen und die Funktionsweise von Bubblesort im Detail nachvollziehen.

1. Konzept: Vergleichsbasiert vs. Nicht-vergleichsbasiert

- **Vergleichsbasierte Algorithmen** (z.B. Bubblesort, Quicksort) vergleichen Elemente paarweise ($<$, $>$).
- **Nicht-vergleichsbasierte Algorithmen** (z.B. Counting Sort, Radix Sort) nutzen andere Eigenschaften (z.B. numerischen Wert) zur Sortierung in "Fächer".
- **Aktion:** Gehe die Sortieralgorithmen aus den Folien durch und ordne sie gedanklich diesen beiden Kategorien zu.
- **Quellen:**
 - PDF 2, Seite 107 (Lineare Sortierverfahren)
 - PDF 3, Seiten 55-56 (Bubblesort), 62 (Mergesort), 69 (Quicksort)

2. Algorithmus im Detail: Bubblesort

- **Fokus:** Verstehe, warum die Frage nach dem "letzten Vergleich" gestellt wird. Der zu sortierende Bereich wird nach jeder Runde um ein Element kleiner.
- **Ablauf:** Das größte Element "blubbert" in der ersten Runde ganz nach rechts. In der zweiten Runde das zweitgrößte an die vorletzte Stelle usw.
- **Aktion:** Spiele den ersten Durchlauf von Bubblesort mit der Menge [0, 8, 1, 16, 0, 32, 19] auf Papier durch. Überlege, welche beiden Elemente im finalen Vergleich des gesamten Algorithmus übrig bleiben.
- **Quellen:**
 - PDF 3, Seiten 55-56 (Bubblesort-Code und Analyse)

3. Wiederholung: Laufzeitkomplexität

- **Aktion:** Verinnerliche die O-Notation der heute besprochenen Algorithmen.
 - Bubblesort: $O(n^2)$
 - Counting Sort: $O(n+k)$

- Radix Sort: $O(n)$
 - **Quellen:**
 - PDF 4, Seiten 24-28 (Allgemeine O-Notation)
-

Tag 2 (Sonntag): Effiziente Algorithmen & Median

(ca. 25 Minuten)

Fokus: Das “Divide and Conquer”-Prinzip verinnerlichen und den Median einer Zahlenmenge sicher bestimmen können.

1. Konzept: Divide and Conquer (Mergesort & Quicksort)

- **Prinzip:** 1. Teile das Problem, 2. Herrsche über die Teile, 3. Kombiniere die Lösungen.
- **Mergesort:** Fokus auf den merge -Schritt (Reißverschlussverfahren).
- **Quicksort:** Fokus auf den partition -Schritt (Umsortieren um ein Pivot-Element).
- **Aktion:** Zeichne den kompletten Mergesort-Ablauf (Zerlegung und Zusammenfügung) für eine kleine Menge von 5-6 Zahlen selbst auf.
- **Quellen:**
 - PDF 1, Seiten 46-49 (Divide and Conquer, Mergesort)
 - PDF 3, Seiten 62-66 (Mergesort), 69-87 (Quicksort)
 - Notizen, Seiten 2 & 3 (Visuelle Beispiele)

2. Anwendung: Der Median

- **Definition:** Der Median ist der Wert, der in der Mitte einer **sortierten** Liste steht.
 - **Vorgehen:**
 1. Liste **immer zuerst sortieren**.
 2. Bei ungerader Anzahl: Das mittlere Element ist der Median.
 3. Bei gerader Anzahl: Der Durchschnitt der beiden mittleren Elemente (falls nicht anders definiert).
 - **Aktion:** Bestimme den Median für die Menge [1, 9, 12, 3, 15]. Bilde dann 2-3 eigene kleine Listen und wiederhole die Bestimmung.
 - **Quellen:**
 - PDF 3, Seite 92 (Medianberechnung)
-

Tag 3 (Montag): Spezialthemen & Beweisführung

(ca. 20 Minuten)

Fokus: Die theoretischen Konzepte hinter der Korrektheit von Algorithmen und deren prinzipiellen Grenzen verstehen.

1. Konzept: Korrektheit & Schleifeninvarianten

- **Idee:** Eine Schleifeninvariante ist eine Bedingung, die **immer** wahr ist (vor dem ersten Durchlauf und nach jedem Durchlauf). Sie dient als Werkzeug zum Beweis der Korrektheit.

- **Aktion:** Sieh dir das Beispiel zur Zusicherungsmethode an. Versuche nachzuvollziehen, warum die angegebene Invariante zu Beginn, während und am Ende der Schleife ihre Gültigkeit behält.
- **Quellen:**
 - PDF 1, Seiten 38-41 (Verifikation, Zusicherungsmethode)
 - Notizen, Seite 1 (Beispiel zur Schleifeninvariante)

2. Konzept: Die untere Schranke $\Omega(n \log n)$

- **Kernaussage:** Kein vergleichsbasierter Sortieralgorithmus kann diese Schranke im Average- oder Worst-Case unterbieten. Schneller geht es prinzipiell nicht, wenn man nur Paare vergleicht.
 - **Aktion:** Versteh, warum diese Aussage die Existenz von schnelleren, nicht-vergleichsbasierten Algorithmen wie Counting Sort erklärt.
 - **Quellen:**
 - PDF 2, Seiten 102-105 (Entscheidungsbäume, Herleitung der unteren Schranke)
 - Notizen, Seite 6 (Beweisidee)
-

Tag 4 (Dienstag im Zug): Finale Wiederholung

- **Aktion:**
 1. Gehe diesen Lernplan noch einmal im Kopf durch.
 2. Fokussiere dich auf die drei Beispielfragen und die dahinterliegenden Lösungsstrategien.
 3. Vergewissere dich, dass du die wichtigsten Laufzeitkomplexitäten den Algorithmen zuordnen kannst.