ALGORITHMEN

Begriffsbildung

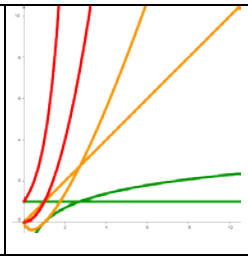
Algorithmus	Schrittweises präzises Verfahren zur Lösung eines Problems
Zeitkomplexität	sagt etwas über die Laufzeit
Speicherkomplexität	sagt etwas über den Speicherbedarf

Vorgehen und Komplexitätsklassen

Anzahl Ausführungen abhängig von n
$2 * proz() + 3n * proz() + 2n^2 * proz()$

↓

O-Notation (Teile ohne Bedeutung bei grossen n werden weggelassen)
$Schritte = 2\boldsymbol{n^2} + 3n + 2$
$\rightarrow O(n^2)$



exponentiell	$O(2^n)$
polynominal	$O(n^k)$
logarithmisch	$O(n * \log n)$
linear	$O(n)$
logarithmisch	$O(\log n)$
konstant	$O(1)$

Gleichwertige Lösungen

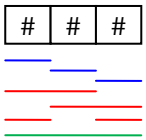
Maximale Teilsummen

Problem: Welche Teilfolge hat die grösste Summe

Tag	1	2	3	4	5	6	7	8	9	10
Gewinn	+5	-8	+3	+3	-5	+7	-2	-7	+3	+5

Lösungen

1. Intuitive Lösung
Jede Teilsumme wird berechnet.



Zeitkomplexität: $O(n^3)$
Speicherkompl.: $O(1)$

2. Zeit für Raum
Teilsummen zwischenspeichern

Länge / Von\	1	2	3
1	#	#	#
2	#	#	
3	#		

Zeitkomplexität: $O(n^2)$
Speicherkompl.: $O(n^2)$

3. Teile und Herrsche
„divide et impera“
in Teilprobleme aufteilen

1. Teile dein Problem
2. Löse jedes Teilproblem
3. Füge Teillösungen zusammen

Zeitkomplexität: $O(n * \log n)$
Speicherkompl.: $O(n)$

4. Optimale Lösung
Jedes Element nur 1x auffassen

1. max. Teilsumme linker Teilfolge
2. rechtes Randmax. linker Teilfolge

Zeitkomplexität: $O(n)$
Speicherkompl.: $O(n)$

Definitionen

Randfolge	 Teilfolge, die irgendwo beginnt und am Rand aufhört.
Rechtes/Linkes Maximum	Maximale Summe aller möglichen Randfolgen.
Gleichwertigkeit von Algorithmen	Wenn sie für jede Eingabe dieselbe Ausgabe produzieren.
Gleichwertigkeit von Datenstrukturen	Wenn sie in jedem Fall die gleichen Resultate liefern.
Beweis -> Plausibilisieren durch Test	Allgemeiner Beweis zur Gleichwertigkeit ist nicht möglich.