Church-Turing-These: Stets haltendes Rechnerverfahren zur Berechnung einer Funktion, oder Lösen eines Problems $\Leftrightarrow \exists$ Turingmaschine, die dies tut.

Sample-Sort: R_1 verteilt $\frac{n}{p}$ große Teile des Arrays an R_i, R_i sortieren und wählen im regelmäßigen Abstand p Samples, senden diese an R_1, R_1 sortiert diese und wählt dann $p-1$ Intervallgrenzen , durchläuft das Array und sendet die Zahlen in den Intervallen an den entsprechenden R_i, R_i sortieren und senden das an R_1.

Dijkstra: bitte auf die Kanten am Startknoten achten, 9 ist klein!

Huffman: Häufigkeitstabelle, kleinste immer zusammen, unter Größere. Der Huffman-Code wird unter Nutzung eines Min-Heaps erstellt. Dabei werden die beiden Knoten mit der niedrigsten Priorität extrahiert und anschließend durch einen Repräsentanten neu in den Heap eingefügt. Dieser Repräsentant hat als Priorität die Häufigkeit seiner beiden Kinder. Dies wird so lange ausgeführt, bis nur noch ein Knoten im Heap vorhanden ist.

Master-Theorem: $T(n) = a \cdot T(\frac{n}{b}) + f(n)$
Wenn $f(n) = O(n^{log_b(a)-\epsilon}), \epsilon > 0 \implies T(n) = \Theta(n^{log_b(a)})$
Wenn $f(n) = \Theta(n^{log_b(a)}) \implies T(n) = \Theta(n^{log_b(a)} \cdot log_b(n))$
Wenn $f(n) = \Omega(n^{log_b(a)+\epsilon}), \epsilon > 0 \implies T(n) = \Theta(f(n))$

Lineare Programmierung: 3A + 6B $\leq$ 420 $\rightarrow$ B = 0 setzen, durch 3 teilen $\implies$ max A = 140, Gleiches für B machen, diagonale durch Zielfunktion g = 36A+45B zu maximieren (Simplex-Algo)

Entscheidbarkeit: Ein Entscheidungsproblem $L \subseteq \Sigma^*$ heißt entscheidbar $\Leftrightarrow \exists$ Turingmaschine M, sodass $\forall x \in \Sigma^*$ gilt: M hält und akzeptiert x. $(x \in L \Leftrightarrow M\ akzeptiert\ x)$

Berechenbarkeit: f ist berechenbar $\Leftrightarrow \exists$ Turingmaschine M mit $\forall x \in \Sigma^* : M$ produziert f(x) und hält, wenn f auf x definiert ist, $\perp$ sonst.

Probleme lösen in? : Problem A hat Lösungs-Algorithmus mit Laufzeit $O(n^4)$ und B $\leq_p$ dauert $O(n^3)$. Dann ist B lösbar in:
$O(n^4 + n^3)$, falls die Transformation die Eingabe von A nicht ändert,
$O(n^12)$, falls die Länge nach der Transformation unbekannt ist
$O(n^8)$, falls die Länge der Eingabe durch die Transformation um n^2 größer wird.

Hält ein Programm? : höchstens: Counter, danach ist es gewiss, obere Schranke ist erreichbar
mindestens: Halteproblem nach k Schritten, untere Schranke nutzlos

Polynomielle Reduktion: Bekannt $\leq_p$ Unbekannt, LAUFZEIT DAZUSCHREIBEN! Das bekannte Problem muss so umgebaut werden, dass es aussieht wie das Unbekannte.

Rekursionsgleichung für max. unabhängigen Weg aus Kanten:

$$f(i) = \begin{cases} w_i, \ falls\ i = 1 \\ max\{w_1, w_2\}, \ falls\ i = 2 \\ max\{w_i + f(i-2), f(i-1)\}\ falls\ i > 2 \end{cases}$$

Iterative Berechnung:
int F [1..n]
F[1] = w_1
F[2] = max$\{w_1, w_»\}$
for (int i = 3; i $\leq$n; i++){
F[i] = max $\{w_i$ + F[i-2], F[i-1]$\}\}$
print (F[n])

Rekursionsgleichung für min Briefmarken:
T(0, P) = $\inf$, falls P ¿ 0 //mit 0 Briefmarken kann man kein P abbauen
T(i, 0) = 0 //nichts mehr zu bezahlen, keine Briefmarken benötigt
T(i, P) = $min\{T(i-1, P), T(i, P - w_i) + 1\}$ //falls die Marke die Marke nicht zu viel Wert ist

NP-Vollstädnig: rate Lösung, verifiziere in Polyzeit

Reduktion Clique $\leq_p$ Tirple-Clique:
Transformiere G zu G”’ durch verdreifachen von G
Korrektheit:
$\Longrightarrow$ Wenn G eine Clique der Größe k enthält, hat G”’ eine Triple-Clique
$\Longleftarrow$ Wenn G”’ mindestens 3 k-Cliquen enthält, muss auch G mindestens eine k-Clique enthalten haben. Diese Reduktion ist in $O(|E| + |V|)$ möglich.