## 1.4. Theorie der Berechenbarkeit

### 1.4.1. Komplexitätsklassen

Computer sind heute allgegenwärtig. Wie wird die Zukunft aussehen? Wird alles vom Computer berechenbar sein, wenn Prozessoren schnell genug sind und Speicher „unbegrenzt“ zur Verfügung steht?

Um Probleme von Computern lösen zu lassen, müssen diese formalisiert werden.

- **Ein Problem ist FORMALISIERBAR, wenn man es syntaktisch und semantisch korrekt ausdrücken kann.**

Das funktioniert bei der Berechnung von Nullstellen, aber für das Schreiben von Gedichten wird dies wohl nicht gelingen.

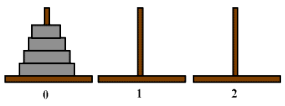
Über die Berechenbarkeit von Problemen machte sich u.a. Alan M. Turing (1912 – 1954) Gedanken. Heute formuliert man:

- **Ein Problem ist BERECHENBAR, wenn es einen Algorithmus gibt, mit dem man bei Eingabe der Frage die Antwort berechnen kann.**

Heute unterscheidet man verschiedene Gruppen von Programmen:

a) *effizient berechenbare Probleme*
Hierzu gehören auf jeden Fall Formeln als Teilmenge der Algorithmen. Das Bilden von Ableitungen, Binomialverteilungen oder auch Lageuntersuchungen von Ebenen gehören in diese Gruppe. Die Laufzeit des Programms und der Speicherbedarf lassen sich über Polynome berechnen. (Klasse P – polynomial)

b) *nicht effizient berechenbare Probleme*
Zu den bekanntesten Problemen dieser Gruppe gehört der „Turm von Hanoi“. Für 4 Scheiben sind $2^4 - 1 = 15$ „Rechenschritte“ nötig. Diese Anzahl wächst allerdings exponentiell (10 Scheiben: 1023 Schritte, 30 Scheiben: 4294967295 Schritte). Es handelt sich um die Klasse NP (nicht polynomial).




c) nicht lösbare Probleme
Ein nicht lösbares Problem ist das plötzliche „Einfrieren“ des Computers. Der PC reagiert auf nichts mehr. Es kann nicht entschieden werden, ob das Verhalten einem besonders hohen Rechenaufwand des Programms oder einem Programmierfehler (Endlosschleife) zuzuordnen ist.