Was sind Rekursionen?

Ein Unterprogramm, dass sich im eigenen Programmablauf selbst aufruft.

Warum Rekursionen?

1. Beispiel: „Zur Wand und wieder zurück“

Aufgabe:
Ein Roboter soll bis zu einer Wand und wieder zurückgehen, ohne sich dabei die Schritte zu merken.

Lösung:
Unterprogramm „HinUndZurück“:
1. Steht der Roboter bereits direkt vor der Wand, so dreht sich einfach um.
2. Steht der Roboter nicht direkt vor der Wand
   a) so geht er einen Schritt vor,
   b) ruft „HinUndZurück“ auf und
   c) geht einen Schritt vor.

Bsp.:
Der Roboter steht zwei Schritte von der Wand entfernt.

Aufruf „HinUndZurück“,
    2. Roboter nicht vor der Wand
    a) Geht Schritt vor
    b) ruft „HinUndZurück“ auf
        2. Roboter nicht vor der Wand
        a) Geht Schritt vor
        b) ruft „HinUndZurück“ auf
            1. Steht vor der Wand und dreht sich um
        c) Geht Schritt vor
    c) Geht Schritt vor

2. Beispiel: Fakultät berechnen

Aufgabe:
Berechne die Zahl 5! (Erinnerung: $5! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$)

Rekursive Lösung:
$5! = 5 \cdot 4! \rightarrow 4! = 4 \cdot 3! \rightarrow ... \rightarrow 1! = 1$
Gedanklich:
5! kann ich nicht berechnen, ich weiß aber, dass es $5 \cdot 4!$ ist. 4! kann mein kleiner Bruder ausrechnen, der wiederum einen kleinen Bruder hat der 3! ausrechnen kann, usw.

Unterprogramm „Fakultät von n“:
    Falls n > 1: Result = n ·„Fakultät von (n-1)“;
    Sonst Result = 1;

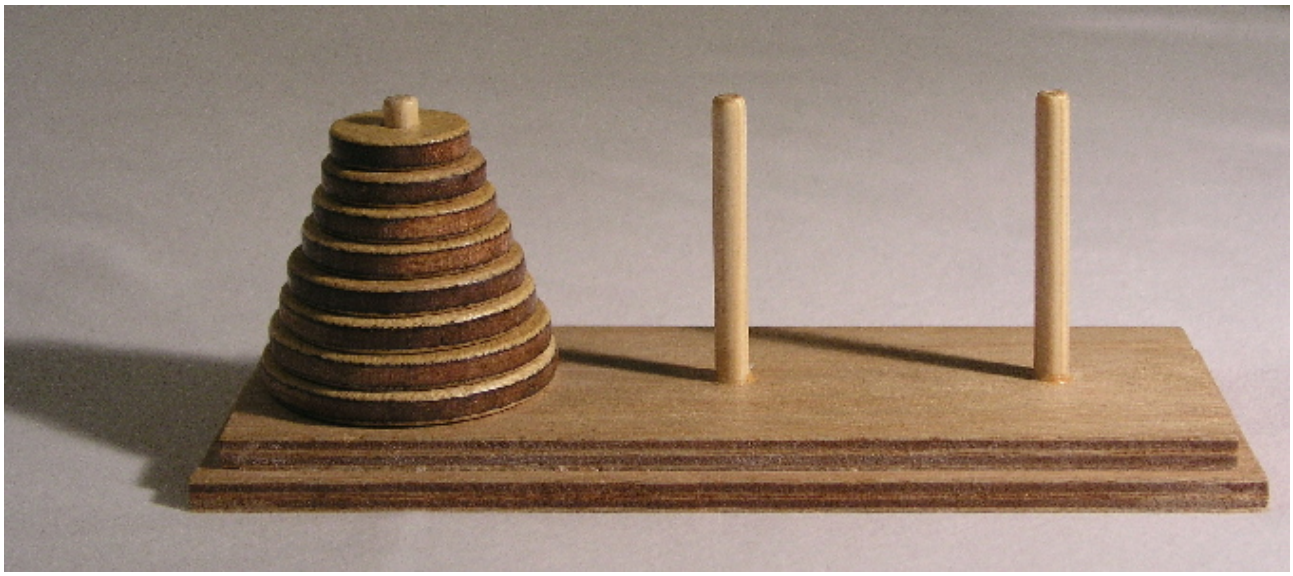
Achtung: Jede Rekursion braucht eine Abbruchbedingung, da sonst das Unterprogramm wieder und wieder aufgerufen, aber nie beendet wird. Da sich der Computer jeden Aufruf des Unterprogramms merken muss, gibt es ohne Abbruchbedingung einen Speicherüberlauf und somit stürzt der Computer, zumindest das Programm, irgendwann ab.

Abbruchbedingungen der Beispiele: 1. Beispiel (Es gibt eine Wand); 2. Beispiel (Irgendwann wird 1! aufgerufen).

Ein weiteres schönes Beispiel ist der euklidische Algorithmus zur Berechnung des ggT zweier natürlicher Zahlen.

→ Die Türme von Hanoi als Spiel

Die Türme von Hanoi



Türme von Hanoi mit acht Scheiben

Edouard Anatole Lucas, ein französischer Mathematiker, erzählte 1883 folgende Legende:

In einem Tempel in der Nähe der Stadt Hanoi liegen 64 kostbare Scheiben aus Diamant zu einem Turm aufgeschichtet. Jede Scheibe ist ein wenig kleiner als die Scheibe auf der sie ruht.
Eine Priesterorden hat nun die Aufgabe erhalten den Turm unter Beachtung heiliger Regeln zu einer anderen Stelle im Tempel zu bewegen. Die Scheiben sind so kostbar, dass sie nur auf einem von drei Plätzen im Tempel liegen dürfen, dem Platz wo der Turm zu Beginn stand, dem wo er aufgebaut werden soll und einem Platz dazwischen. Die Scheiben sind schwer und zerbrechlich, daher darf immer nur eine der Scheiben bewegt werden, niemals mehrere zur gleichen Zeit Die letzte Regel besagt, dass zu keiner Zeit eine Scheibe auf einer kleineren Scheibe liegen darf.
Wenn der Turm an einer Stelle abgebaut und an andere Stelle wieder ganz aufgebaut wurde, wird der Tempel und mit ihm die ganze Welt zu Staub zerfallen.

Wie lange brauchen die Mönche, wenn das Umlegen einer Scheibe nur eine Sekunde dauert und die Mönche rund um die Uhr (ohne Pausen) arbeiten?
Antwort: ca. 585 Milliarden Jahre
Zum Vergleich: Geschätztes Alter des Universums: 13,7 Milliarden Jahren

Warum ist das so?
Bei der rekursiven Lösung verdoppelt sich pro Scheibe die Anzahl der Züge.
1 Scheibe: 1 Zug
2 Scheiben: 3 Züge
3 Scheiben: 7 Züge
4 Scheiben: 15 Züge
5 Scheiben: 31 Züge
10 Scheiben: 1023 Züge
20 Scheiben: 1048575 Züge
30 Scheiben: 1073741823 Züge
64 Scheiben: 18446744073709551615 Züge

Als mathematische Formel: Anzahl der Züge = $2^{(\text{Anzahl der Scheiben})} - 1$

→ Schöne Java-Script-Version mit ausführlicher Problembeschreibung und rekursiver Lösung