Rekursion

Unter Rekusion verstehen wir eine Funktion, die sich selbst aufruft. Da sie das nicht immerzu tun kann (das Programm würde ewig laufen) benötigt jeder rekursive Aufruf eine Abbruchbedingung!

Beim rekursiven Aufruf einer Funktion wird jeweils ein Duplikat der gesamten Funktion im Speicher abgelegt.

Rekursive Algorithmen sind oft verblüffend einfach, benötigen aber oft auch viel Speicherplatz!

Beschreibe was der Aufruf

```
  echo fak(5);
```

leistet!

```
function fak($i){
  if ($i==1) return 1; else
  return fak($i-1)*$i;
}
```

```
fak(5)
fak(4)       *5
fak(3)     *4*5
fak(2)   *3*4*5
fak(1) *2*3*4*5
      1*2*3*4*5
=120
```

Beschreibe was der Aufruf

echo fak2(5);

leistet!

fak_iter(5,1);
fak_iter(4,5);
fak_iter(3,20);
fak_iter(2;60);
fak_iter(1,120);
120;

Quelltext:

```
function fak2($i){
  return fak_iter($i,1);
}


function fak_iter($i1,$i2){
  if ($i1==1) return $i2;
  else return fak_iter($i1-1,($i2*$i1));
}

function fak($i){
  if ($i==1) return 1; else
  return fak($i-1)*$i;
}
```

Andere Beispiele:

a) Größter gemeinsamer Teiler

es gilt:
```
ggT(m,0)=m;
ggT(m,n)=ggT(n,m);
ggT(m,n)=ggT(m,n-m) für n>=m;
```

euklidscher Algorithmus:

```
ggT(m,0)0m;
ggT(m,n)=ggT(m,n mod m) für m>0;
```

Erstelle die Funktionen ggT($m,$n) und ggT2($m,$n), die als Ergebnis den ggT zweier Zahlen liefern.

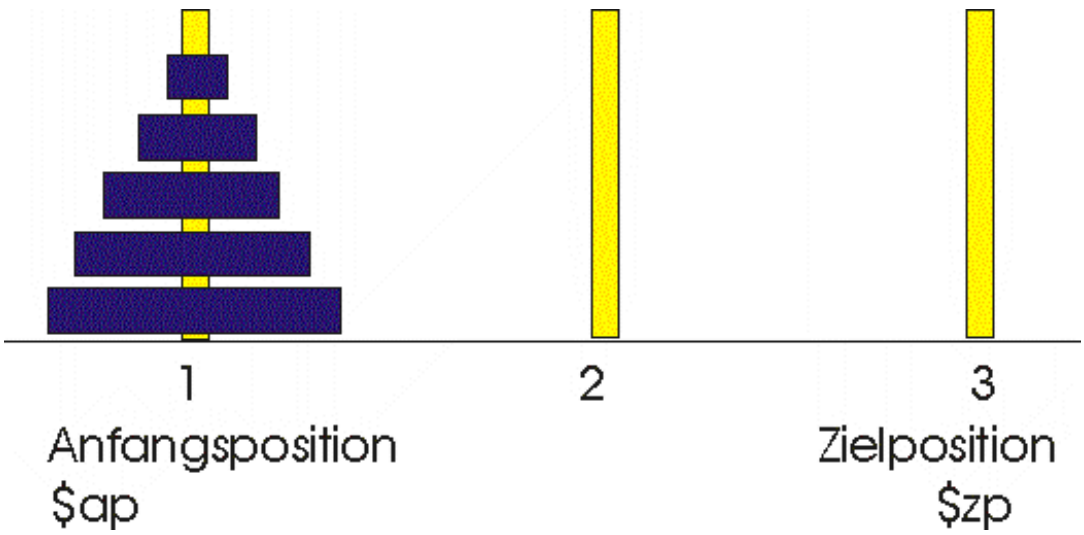
b) Türme von Hanoi

"Vermutlich wurde das Spiel 1883 vom französischen Mathematiker Edouard Lucas erfunden. Er dachte sich dazu die Geschichte aus, dass indische Mönche im großen Tempel zu Benares, im Mittelpunkt der Welt, einen Turm aus 64 goldenen Scheiben versetzen müssten, und wenn ihnen das gelungen sei, wäre das Ende der Welt gekommen.

Obwohl es auf den ersten Blick kompliziert klingt, gibt es für das Spiel einen ganz einfachen rekursiven Algorithmus. Da sich ein Computerprogramm zur Lösung des Spiels mit wenigen Zeilen schreiben lässt, ist Türme von Hanoi ein klassisches Beispiel für rekursive Programmierung.

Die minimale Anzahl von Zügen für einen Stapel aus *n* Scheiben beträgt 2^n-1, bei einem Turm von 8 Scheiben (die gängigste Variante) also 255 Züge. Für den Stapel aus 64 Scheiben würden 18.446.744.073.709.551.615, also mehr als 18 Trillionen Züge benötigt. Würde man jede Sekunde eine Scheibe bewegen, bräuchte man dafür über 584 Milliarden Jahre."

Quelle: http://www.wissensnetz.de/lexikon/wiki,index,goto,Turm_von_Hanoi.html geladen am 15.02.2005



Im Laufe der Aufrufe ist $ap und $zp stets bekannt. Wie kann man die Nummer der Hilfsstelle (des dritten Stapels) erhalten?

$Hilfsstelle = 6 - $ap-$zp

Warum? ;-)

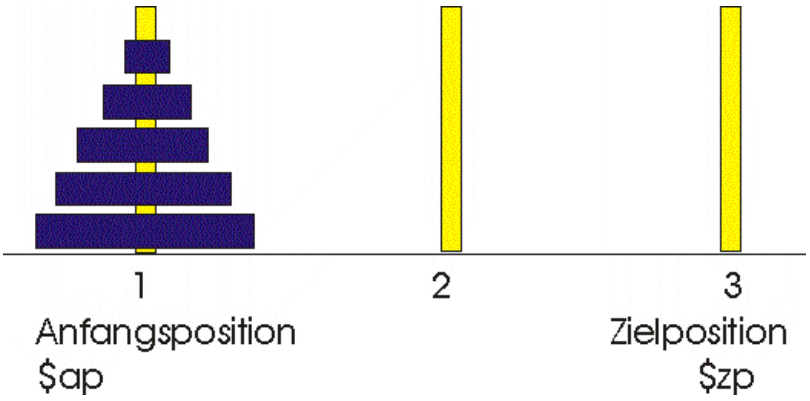
Lösungsalgorithmus für n=5:
BewegeTurm (4,1,2);
BewegeTurm (1,1,3);
BewegeTurm(4,2,3);

Lösungsalgorithmus für n=4 von 1 nach 2:
BewegeTurm (3,1,3);
BewegeTurm (1,1,2);
BewegeTurm(3,3,2);

Lösungsalgorithmus für n=3 von 1 nach 3:
BewegeTurm (2,1,2);
BewegeTurm (1,1,3);
BewegeTurm(2,2,3);

Lösungsalgorithmus für n=2 von 1 nach 2:
BewegeTurm (1,1,3);
BewegeTurm (1,1,2);
BewegeTurm(1,3,2);

Lösungsalgorithmus für n=1 von 1 nach 3:
BewegeTurm (1,1,3);



Gegeben seien n Scheiben unterschiedlichen Durchmessers, die der Größer nach geordnet zu einem Turm geschichtet sind; die unterste Scheibe ist die größte. Der Turm steht auf einem Platz 1. Unter Verwendung eines Hilfsplatzes 3 soll der Turm auf einen Platz 2 transportiert werden. Beim Transport sind folgende Bedingungen einzuhalten:

(1)Es darf stets nur eine Scheibe, und zwar die oberste eines Turms, bewegt werden.
(2)Zu keiner Zeit darf eine größere Scheibe auf einer kleineren liegen.

Einen rekursiven Lösungsalgorithmus erhält man durch die Aufspaltung des Gesamtproblems

P = "Transportiere n Scheiben von Platz 1 nach Platz 2"

in die drei Teilprobleme
P1 = "Transportiere n-1 Scheiben von platz 1 nach Platz 3",
P2 = "Transportiere letzte Scheibe von Platz 1 nach Platz 2"
und
P3 = "Transportiere n-1 Scheiben von Platz 3 nach Platz 2".

Dies führt zu folgender Prozedur, wobei ap die Nummer des Anfangsplatzes, zp die Nummer des Zielplatzes und deshalb 6-ap-zp die Nummer des Hilfsplatzes ist:

```
procedure hanoi(n,ap,zp:integer);

begin
  if n>1 then
     hanoi(n-1,ap,6-ap-zp) fi;
  write("Scheibe", n, "von Platz", ap, "nach Platz", zp);
  if n>1 then
     hanoi(n-1,6-ap-zp,zp) fi;
end;
```

Aufruf: hanoi(n,1,2); für n>=1

Quelle: Informatik-Duden

Schreibe den pascal-ähnlichen Quelltext um in PHP! (Portiere das Programm nach PHP!)