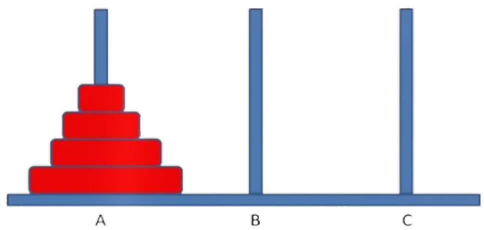
Türme von Hanoi

Das Spiel TÜRME VON HANOI besteht aus drei Holzstäben A, B und C. Auf Stab A sind n Scheiben mit nach oben sukzessiv kleiner werdenden Radien aufgestapelt. Es soll ein Algorithmus gefunden werden, der den Turm von Stab A auf den Stab C versetzt. Dabei darf in jedem Schritt immer nur eine Scheibe auf einen leeren Stab oder auf eine größere Scheibe versetzt werden.




```
# hanoi.py
def hanoi(n, start, end, helper):
  if n == 1:
  print(start + "->" + end)
  else:
    hanoi(n-1, start, helper, end)
    print(start + "->" + end)
    hanoi(n-1, helper, end, start)

def user_input():
  return int(input())

if __name__ == "__main__":
  n = user_input()
  hanoi(n, 'A', 'C', 'B')
```

Da man für eine Scheibe einen Zug, für 2 Scheiben 3 Züge und für 3 Scheiben 7 Züge benötigt, lässt sich ein Pattern naheliegen, dass man allgemein für n Scheiben $2^n - 1$ Züge benötigt.

Proof. **Beweisen Sie induktiv, dass diese Formel richtig ist**

Vollständige Induktion

Induktionsanfang:

Für $n = 1 \Rightarrow 2^1 - 1 = 1 \longleftarrow$ ist wahr

Induktionsvoraussetzung:

Gilt für jede $n \in \mathbb{N}$

Induktionsschritt:

Für $n = n + 1 \Rightarrow 2^{n+1} - 1 = 2 * 2^n - 1 = 2 * (2^n - 1) \longleftarrow$ Voraussetzung

$\square$

Es ist nicht möglich, den Turm mit weniger Zügen zu versetzen

Annahme: Es ist möglich, den Turm von Hanoi mit weniger als $2^n - 1$ Zügen zu versetzen.

$$n = 1 \quad \Rightarrow \quad 2^{n-1} - 1 = 2^{1-1} - 1 = 2^0 - 1 = 0.$$

Da aber mindestens 1 Zug benötigt wird, folgt, $\lightning$

Die Fibonacci-Folge

Die Fibonachi-Folge wird rekursiv definiert

$$a_n = \begin{cases} 1, & \text{für} n = 1 \\ 1, & \text{für } n = 2 \\ a_{n-1} + a_{n-2}, & \text{für} n > 2 \end{cases}$$