Rucursive(遞迴)程式設計

Rucursive 函式: 函式在其內部呼叫自己。

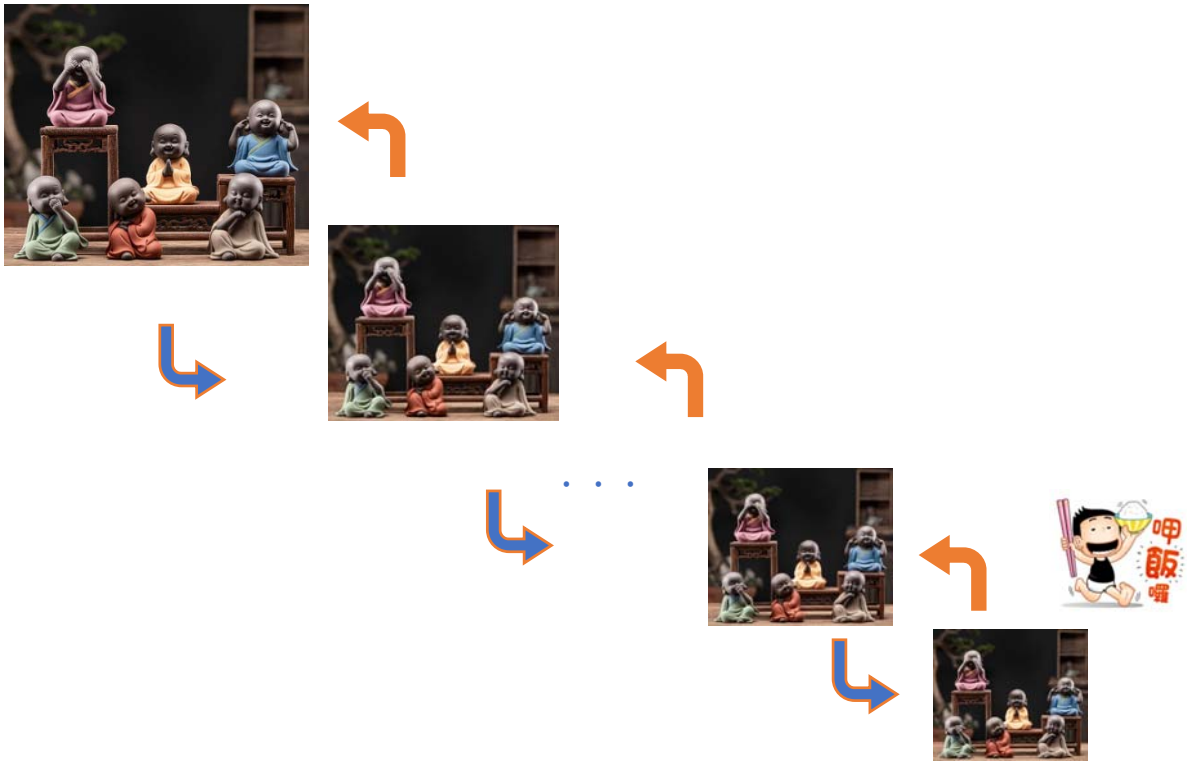
從前從前有一座山，山裡面有一座廟，廟裡的老和尚，對著一群小和尚講故事:

從前從前有一座山，山裡面有一座廟，廟裡的老和尚，對著一群小和尚講故事:

...

...

從前從前有一座山，山裡面有一座廟，廟裡的老和尚，對著一群小和尚講故事。



Q: 請計算 1+2+3+…+(n-1)+n = 的總和。

假設 sum(n) 可以計算出上述算式的結果，同時我們發現有底下的規則:

```
sum(n) = 1+2+3+...+(n-1) + n
       = sum(n-1) + n
       sum(n-2) +(n-1)
       ...
      sum(1) + 2
```

遞迴程式設計的解題規則:

1) 假設存在可以解決問題的遞迴函式。
2) 找出解法過程中，問題重複的規則。
3) 確定問題可以結束的條件。

最後就好像會"莫名其妙"地解決了問題。

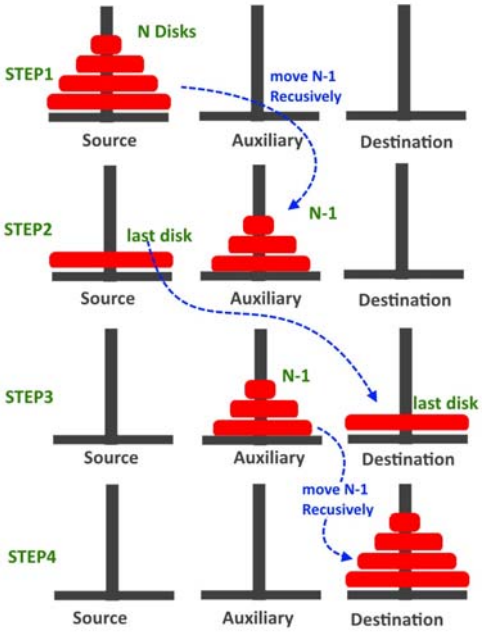
Q: **Hanoi tower(河內塔)**

任務就是要將 n 個 disks 從 A 柱搬移到 C 柱。

但是，在過程中，一次只能搬一個 disk，且不可以讓大 disk 壓住小 disk，

這裡我們假設號碼越大表示 disk 也越大。

顯然依據上述規則，光是兩個柱子是無法完成這個任務的，因此必須要有一個用來暫放的輔助柱子。



hanoi(n, A, B, C): 能將 n 個碟子從 A 搬到 C。

step 1: hanoi(*n-1*, A, C, B)

step 2: move disk n from A to C

step 3: hanoi(*n-1*, B, A, C)

```
def   hanoi(n, A, B, C):
      if n == 1:
            print(f"move disk {n} from {A} to {C}.")
      else:
            hanoi(n-1, A, C, B)
            print(f"move disk {n} form {A} to {C}.")
            hanoi(n-1, B, A, C)

hanoi(3, 'A', 'B', 'C')
```