.

Recurrence Relations

Recurrence Relations

$\star$ **Puzzle 2 Revisited:** You are given 27 golf balls, one of which is heavier than the rest, and a balancing scale. Can you find the heavier ball with 3 weighings?

Consider again the golf balls problem, in which there are n balls. The following algorithm reduces a problem of size n to a problem of size $n/3$ in one weighing. This algorithm can only handle inputs n which are a power of 3.

```
GolfBalls(S)    {S is a set of n balls where n is a
                 power of 3, and one ball is heavy.}
    if n = 1 then
        output the ball in S
    else
        divide S into 3 subsets of size n/3, say A,B,C
        put A and B on either side of the scale
        if wt(A) = wt(B) then GolfBalls(C)
        if wt(A) < wt(B) then GolfBalls(B)
        if wt(A) > wt(B) then GolfBalls(A)
```

Let $W(n)$ denote the number of weighings by GolfBalls(S),

where $|S| = n$. Then

$$\begin{aligned} W(n) &= 1 + W(n/3), \qquad n = 3^k,\ k \geq 1, \\ W(1) &= 0. \end{aligned}$$

This first equation here is called a **recurrence relation** because it describes $W(n)$ as a function of W on a smaller number (namely $n/3$). It tells us, for example,

- The number of weighings for 27 golf balls is 1 plus the number of weighings for 9 golf balls.
- The number of weighings for 9 golf balls is 1 plus the number for 3 golf balls.
- The number for 3 golf balls is 1 plus the number for 1 golf ball.

The second equation is called the **base case** and is needed to tell us how many weighings are needed for the simplest case in our algorithm.

Solving this recurrence will give us a closed form expression for the number of weighings as a function of n. We can solve using the **iteration method**:

⋆

$$\begin{aligned} W(n) &= 1 + W(n/3) \\ W(n) &= 1 + 1 + \\ W(n) &= 1 + 1 + 1 + \\ &\vdots \\ W(n) &= \underbrace{1 + 1 + \ldots + 1}_{i \text{ times}} + \end{aligned}$$

Let $n = 3^k$ and let $i = k$ in the last expression. Then

$$\begin{aligned} W(n) &= \\ W(n) &= \\ W(n) &= \end{aligned}$$

⋆ **Puzzle 3: Towers of Hanoi** Given 3 pegs and n disks of different sizes placed in order of size on one peg, transfer the disks form the original peg to another peg with the constraints that:

- Each disk is on a peg.
- No disk is ever on a smaller disk.
- Only one disk at a time is moved.

HANOI(START,TEMP,END,n)

{Solve the towers of Hanoi for $n \geq 1$ disks.}

if $n = 1$ then

Move START's top disk to END.

else

HANOI(START,END,TEMP,$n-1$)

Move START's top disk to END.

HANOI(TEMP,START,END,$n-1$)

⋆ Puzzle 3: Towers of Hanoi with 3 disks