Recurrence Relations

Definition: A *recurrence relation* is a recursive way of defining a sequence $\{a_0, a_1, a_2, \ldots, a_n\}$. In a recurrence relation, a_n is expressed in terms of $a_0, a_1, \ldots, a_{n-1}$.

An example of a recurrence relation is

$$a_n = 2a_{n-1}, \quad a_0 = 1.$$

In this case, $a_0 = 1$ is the *initial condition* or *boundary condition*. The initial condition corresponds to the base case in a proof by induction.

Example: The number of bacteria in a colony doubles every day. If the colony begins with 1 cell, how many are there after 6 days?

- $a_n = 2a_{n-1}$, $a_0 = 1$
- $a_n = \#$ of cells on the nth day
- $a_0 = 1$, $a_1 = 2$, $a_2 = 4$, $a_3 = 8$, $a_4 = 16$, $a_5 = 32$, $a_6 = 64$, $a_7 = 128$
- By inspection, we conclude that $a_n = 2^n$.

Example: In a radioactive decay process, $\frac{1}{3}$ of the material is lost due to decay every year. What is the amount of material left after n years if at $t = 0$ there are x grams of material present?

- The initial condition is $a_0 = x$.
- $a_n =$ material left after n years
- In the next year, a third (i.e. $\frac{1}{3}a_n$) decays away.
- So, the amount left after $n + 1$ years is

$$a_{n+1} = a_n - \frac{1}{3}a_n = \frac{2}{3}a_n$$

- In general, if the decay rate is μ (i.e. a fraction μ of the material is lost in every year), then

$$a_{n+1} = (1 - \mu)a_n, \quad a_0 = x$$

$$a_n = (1 - \mu)^n x$$

- For $\mu = \frac{1}{3}$ and $x = 1$:

$$a_n = \left(\frac{2}{3}\right)^n$$

For example, after five years, $a_5 = \left(\frac{2}{3}\right)^5$.

Example: "Fibonacci Numbers." A pair of newborn rabbits (one of each sex) is placed on an island. Let's say that a pair of rabbits will not breed until they are two months old, and after that, they produce another pair (again, one of each sex) every month. How many pairs of rabbits are there after n months, assuming no rabbit ever dies? (It's best not to think about which rabbits are mating with which.)

- The number of pairs after n months is

$$\begin{aligned} F_n &= \#\text{ in the } (n-1)\text{th month plus the newborn rabbits.} \\ &= F_{n-1} + \text{ the number of newborn pairs} \end{aligned}$$

- # of newborn pairs $= F_{n-2}$, since there is a new one for every pair that was on the island two months ago.

- $F_1 = 1$ and $F_2 = 1$, because the first pair does not breed in the first two months.

$$F_n = F_{n-1} + F_{n-2}, \quad F_1 = 1, F_2 = 1$$

This sequence is called the Fibonacci numbers:

$$F_3 = 2,\ F_4 = 3,\ F_5 = 5,\ F_6 = 8,\ F_7 = 13,\ F_8 = 21,\ F_9 = 34,\ \ldots$$

Example: "Compound Interest." A person deposits \$100 in a savings account at a bank yielding 10% per annum interest, compounded every year. What is the amount in the account after 10 years?

$$\begin{aligned} A_n &= \text{amount after } n \text{ years} \\ A_0 &= 100 \\ A_n &= A_{n-1} + \text{interest} \\ &= A_{n-1} + 10\% \text{ of } A_{n-1} \\ &= A_{n-1} + 0.1A_{n-1} \\ &= 1.1A_{n-1} = (1.1)^2 A_{n-2} = (1.1)^3 A_{n-3} = \cdots \\ &= (1.1)^n A_0 \\ A_{10} &= 1.1^{10}(\$100) \approx \$260 \end{aligned}$$

Example: Consider the recurrence relation

$$a_n = 2a_{n-1} - a_{n-2}, \quad \text{where } a_0 = 5 \text{ and } a_1 = 5$$

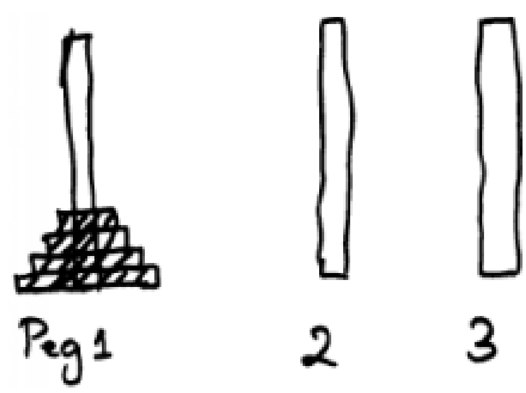
- $a_2 = 2a_1 - a_0 = 2 \cdot 5 - 5 = 5$
- $a_3 = 2a_2 - a_1 = 5$
- $a_n = 5$

Example: Consider the same relation with different initial conditions

$$a_n = 2a_{n-1} - a_{n-2}, \quad \text{where } a_0 = 5 \text{ and } a_1 = 10$$

- $a_2 = 2a_1 - a_0 = 2 \cdot 10 - 5 = 15$
- $a_3 = 2 \cdot 15 - 10 = 20$
- $a_4 = 2 \cdot 20 - 15 = 25$
- $a_n = 5(n + 1)$

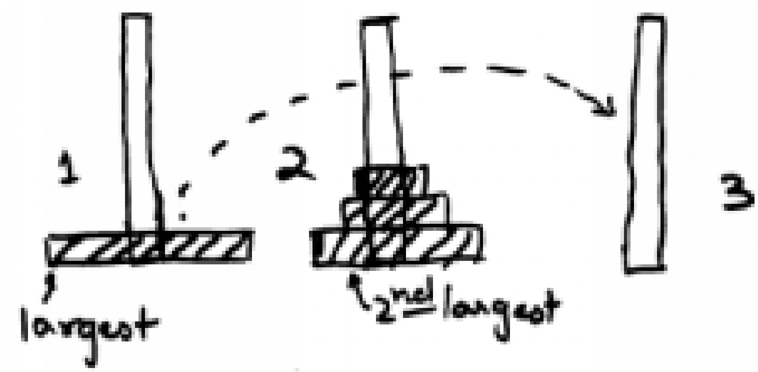
Example: Towers of Hanoi



We want to move n disks from Peg 1 onto Peg 3, one disk at a time. The catch is that a larger disk can never be placed on top of a smaller disk. How many moves does it take?

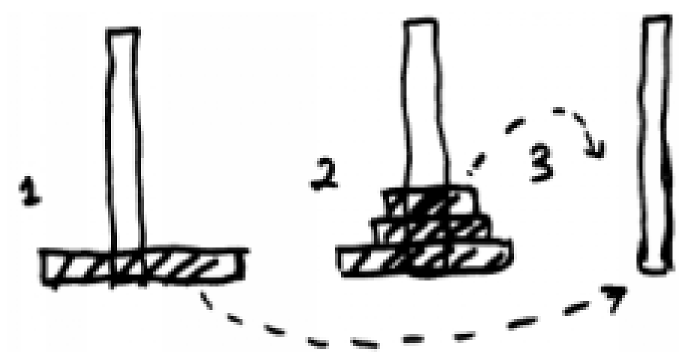
- Let's say the number of moves required is H_n.
- If there were only one disk, it would take one move, so $H_1 = 1$.

- If we knew how to move the first $n-1$ disks to Peg 2, then we could move the largest disk to Peg 3, and move all the $n-1$ smaller disks from Peg 2 onto the largest disk on Peg 3.



- So our algorithm is:

  1. Recursively move $n-1$ (all but the largest) from Peg 1 to Peg 2. This takes H_{n-1} steps.
  2. Move the largest disk from Peg 1 to Peg 3.
  3. Recursively move $n-1$ from Peg 2 to Peg 3. This takes another H_{n-1} steps.



- Adding these up, we get

$$
\begin{aligned}
H_n &= \#\text{ of moves for } n \text{ disks} \\
&= 2H_{n-1}+1 \\
&= 2(2H_{n-2}+1)+1 = 4H_{n-2}+2+1 \\
&= 4(2H_{n-3}+1)+2+1 = 8H_{n-3}+4+2+1 \\
&= 2^{n-1}H_1 + 2^{n-2} + 2^{n-3} + \cdots + 2 + 1 \\
&= 2^n - 1
\end{aligned}
$$

Example: What is the number of strings of length n with no consecutive zeroes?

- a_n = Desired # of strings = Those that end with 1 plus those that end with 0.
- In any n-bit string ending in 1, if there are two consecutive zeroes, the last bit isn't one of them.

  $$\sqcup \sqcup \cdots \sqcup \sqcup \sqcup 1$$

  So the number of n-bit strings that end with 1 and have no consecutive zeroes is the same as the number of $(n-1)$-bit strings with no consecutive zeroes, i.e. a_{n-1}.
- If an n-bit string ends with 0, then the second-to-last bit must be a 1.

  $$\sqcup \sqcup \cdots \sqcup \sqcup 0\,1$$

  So the number of n-bit strings that end with 0 and have no consecutive zeroes is the same as the number of $(n-2)$-bit strings with no consecutive zeroes, i.e. a_{n-2}.
- $a_n = a_{n-1} + a_{n-2}$, with $a_1 = 2$ and $a_2 = 3$.
- $a_n = F_{n+1}$ (the $n+1$st Fibonacci number)

Example: "Enumerating certain types of codewords" A valid codeword of length n is a binary string of n bits with an even number of 1s. What is the number of codewords of length n?

- Let a_n be the number of valid codewords of length n.
- There are two ways of obtaining a valid code word of length n:
  - Case I: If the word ends with a 1, then the first $n-1$ bits must be an invalid codeword of length $n-1$. The number of invalid codewords of length $n-1$ is $2^{n-1} - a_{n-1}$.
  - Case II: If the word ends with a 0, then the first $n-1$ bits must be a valid codeword of length $n-1$. There are a_{n-1} such words.
- So the total number of valid codewords is $a_n = (2^{n-1} - a_{n-1}) + a_{n-1} = 2^{n-1}$