Lecture 20: Recursion
recurrere = to run back (L)

Recursively defined sequence: the k-th term is defined in terms of 1 or more of the preceding terms; one or more initial terms must be given

\[ a_k = 2a_{k-1} + 3 \quad a_0 = 1 \]

\[ a_k = (a_{k-2})^2 \quad a_0 = 2 \quad a_1 = 3 \]

ex. 1  Fibonacci sequence (Rabbits)

<table>
<thead>
<tr>
<th>Month</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td># Pairs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ex. 2  The number of individual disk-moves needed to solve the Towers of Hanoi puzzle

Let \( m_k \) be the total number of individual disk-moves if the puzzle has k disks.
ex. 3  Compound interest after $k$ years at rate $r$

Let $A_k$ be the total amount after $k$ years.

ex. 4  The number of strings in $\Sigma^k$ where $\Sigma = \{0, 1\}$ that do not contain the pattern “11”

Hint: How can such strings start?

ex. 5  The number of ways of choosing $k$ objects out of a set of $n$ objects $S_{n,k}$

Hint: Focus on 1 point
**ex. 6** How many ways can you partition a set with \( n \) elements into \( r \) subsets? \( S_{n,r} \)

Easy cases:

\[
\begin{align*}
S_{1,1} & \\
S_{2,1} & \quad S_{2,2} \\
S_{n,1} & \quad S_{n,n}
\end{align*}
\]

Hint: Pick a point *. In each partition, either \{*\} is a set or * is part of another set.

---

**Solving recurrence relations**

Solving a recurrence relation means

**Methods:**

A. Iteration

* Idea: Substitute until you see a pattern, then make an intelligent guess. (The guess can be proved by induction.)*

ex: The compound interest recurrence relation
ex: the Towers of Hanoi puzzle. Let $T(n)$ be the number of disk-moves to solve the puzzle with $n$ disks.

Object: Move the entire stack of disks from peg A to peg C
Rules: Move 1 disk at a time.
   Place no larger disk atop a smaller disk.
   You may use peg B as a helper or "auxiliary" peg.

$T(1) =$
$T(2) =$
$T(3) =$