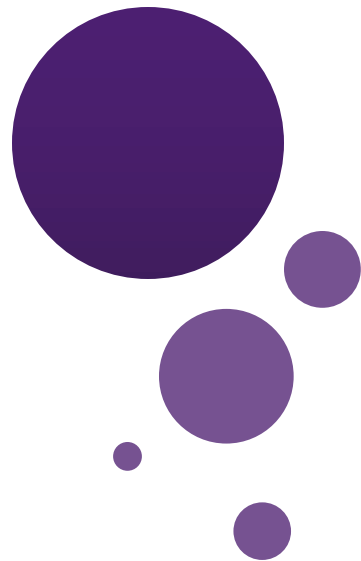




UNIVERSITY
AT ALBANY

State University of New York

Lecture 24: Recursive Algorithms and Basic Counting Rules



Dr. Chengjiang Long
Computer Vision Researcher at Kitware Inc.
Adjunct Professor at SUNY at Albany.
Email: clong2@albany.edu

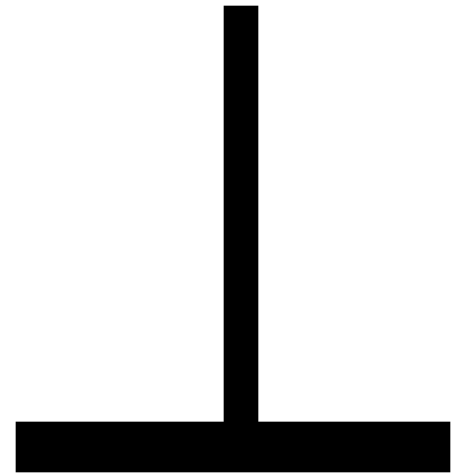
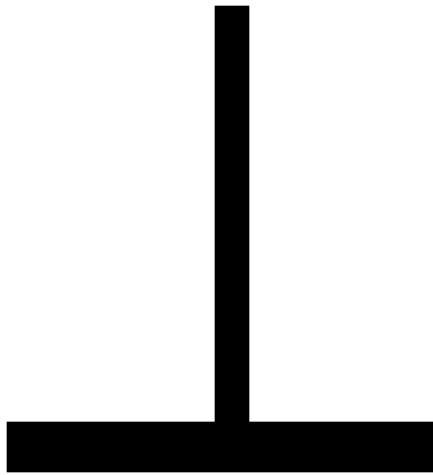
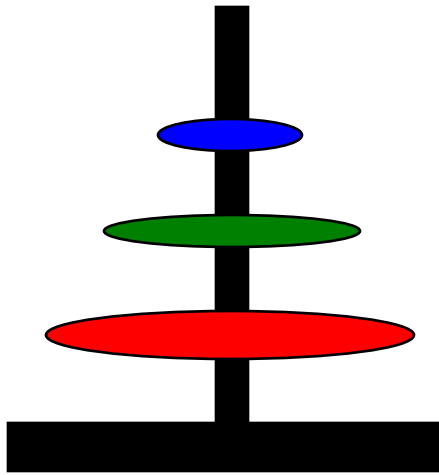
Outline

- Recursive Algorithms: Towers of Hanoi
- Basic Counting Rules
 - Sum Rule
 - Product Rule
 - Generalized Product Rule

Outline

- **Recursive Algorithms: Towers of Hanoi**
- Basic Counting Rules
 - Sum Rule
 - Product Rule
 - Generalized Product Rule

Towers of Hanoi (N=3)



Towers of Hanoi

- There are three pegs.
- 64 gold disks, with decreasing sizes, placed on the first peg.
- You need to move all of the disks from the first peg to the second peg.
- Larger disks cannot be placed on top of smaller disks.
- The third peg can be used to temporarily hold disks.

Towers of Hanoi

- The disks must be moved within one week. Assume one disk can be moved in 1 second. Is this possible?
- To create an algorithm to solve this problem, it is convenient to generalize the problem to the “N-disk” problem, where in our case $N = 64$.

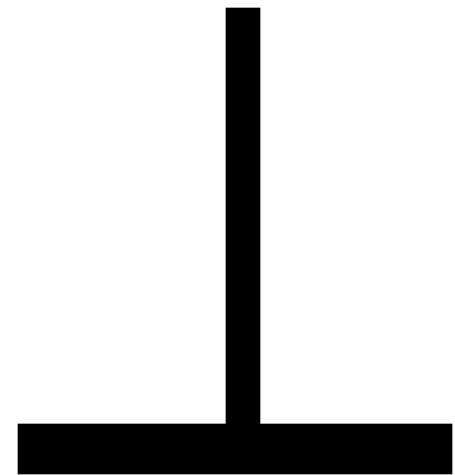
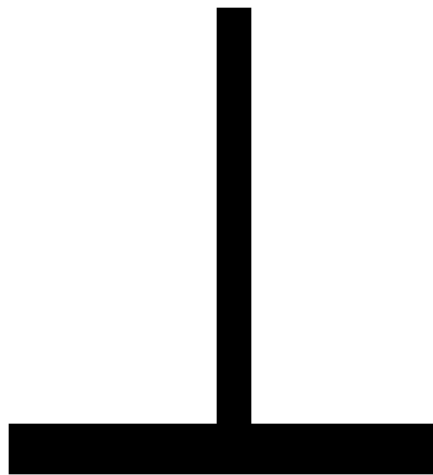
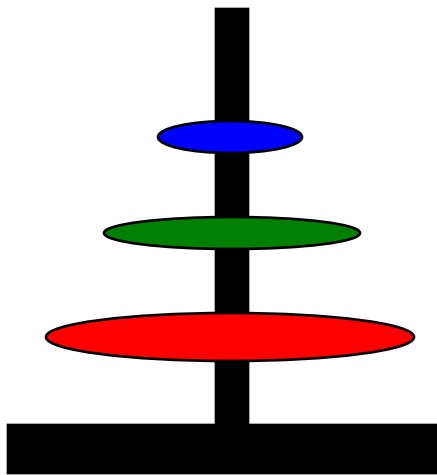
Towers of Hanoi

- How to solve it?
- **Think recursively!!!!**
- Suppose you could solve the problem for $n-1$ disks, i.e., you can move $(n-1)$ disks from one tower to another, without ever having a large disk on top of a smaller disk. How would you do it for n ?

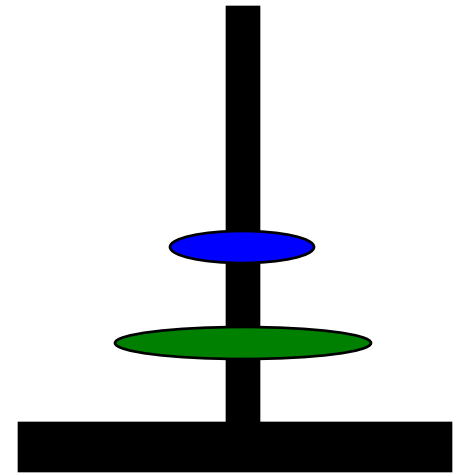
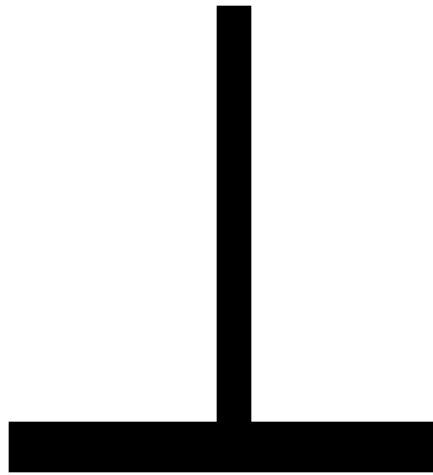
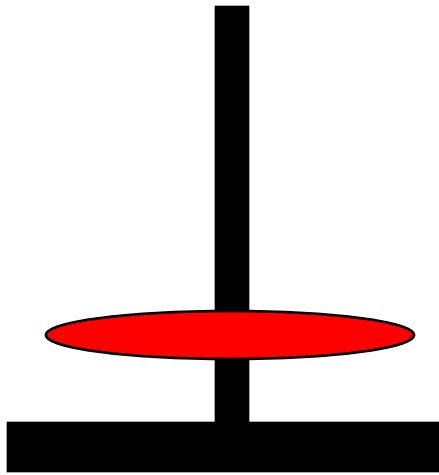
Towers of Hanoi

- Solution:
 1. Move top $(n-1)$ disks from tower 1 to tower 3 (you can do this by assumption – just pretend the largest ring is not there at all).
 2. Move largest ring from tower 1 to tower 2.
 3. Move top $(n-1)$ rings from tower 3 to tower 2 (again, you can do this by assumption).

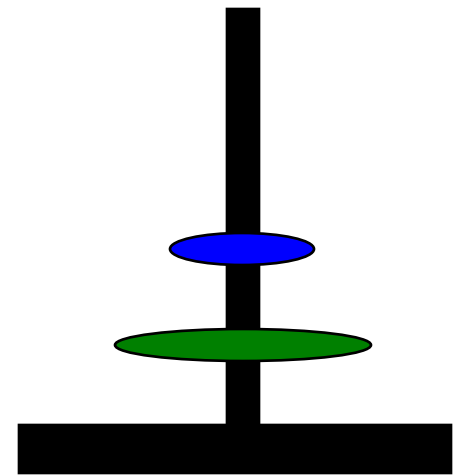
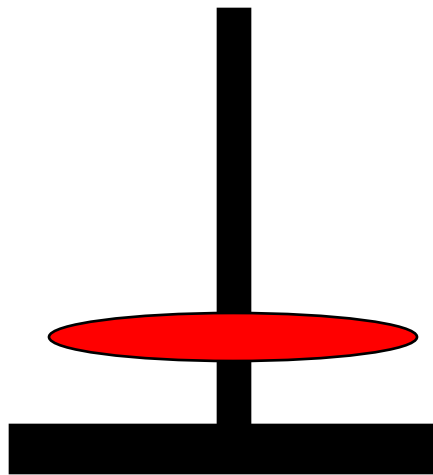
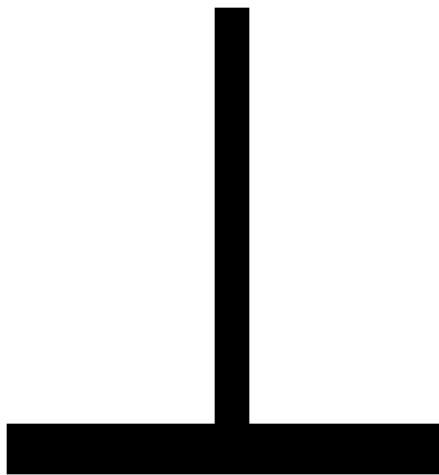
Recursive Solution



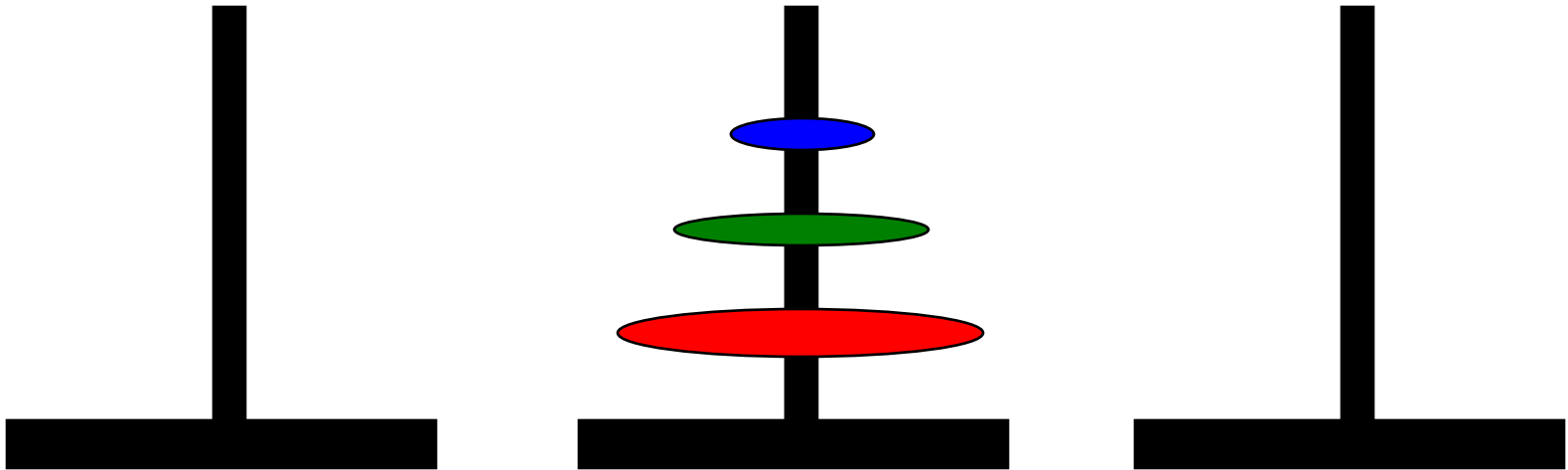
Recursive Solution



Recursive Solution



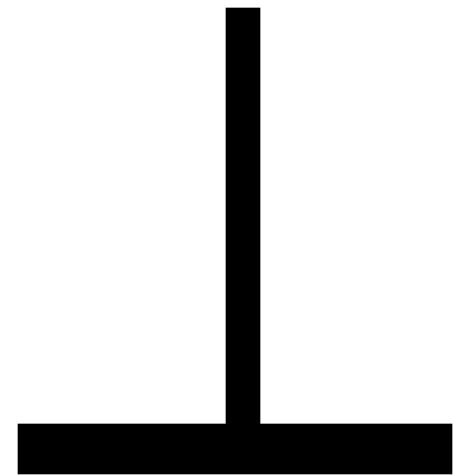
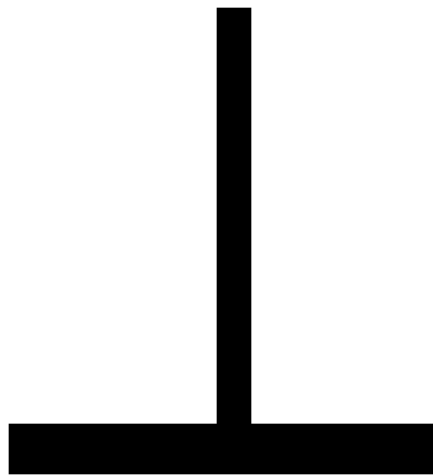
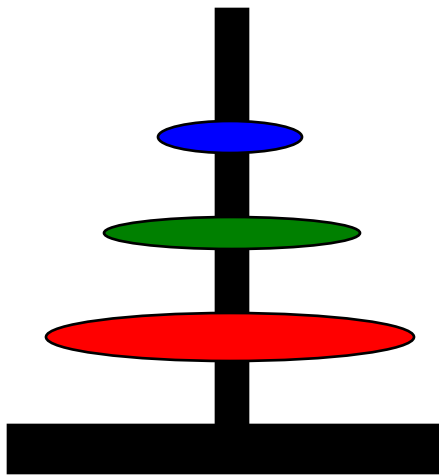
Recursive Solution



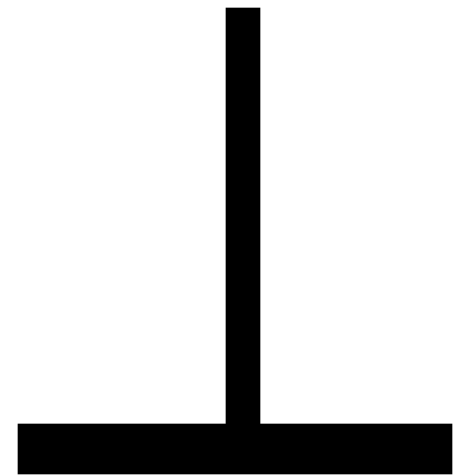
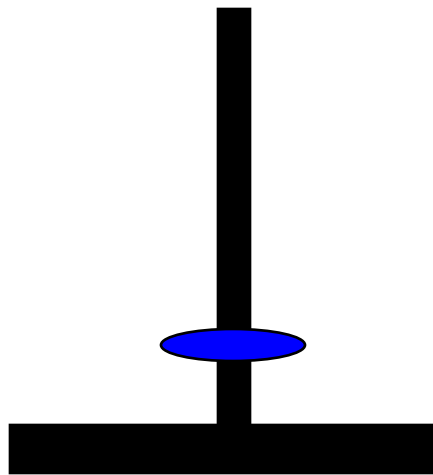
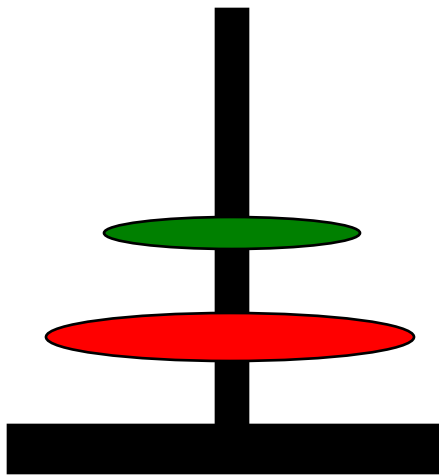
Towers of Hanoi

- **Procedure** *TowerHanoi* (n, a, b, c : n, x, y, z integers, $1 \leq a \leq 3, 1 \leq b \leq 3, 1 \leq c \leq 3$)
- **if** $n = 1$ **then**
- $move(a, b)$
- **else**
- **begin**
- *TowerHanoi*($n-1, a, c, b$)
- $move(a, b)$;
- *TowerHanoi* ($n-1, c, b, a$);
- **end**

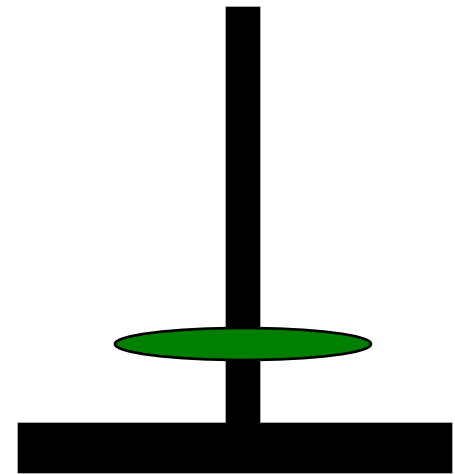
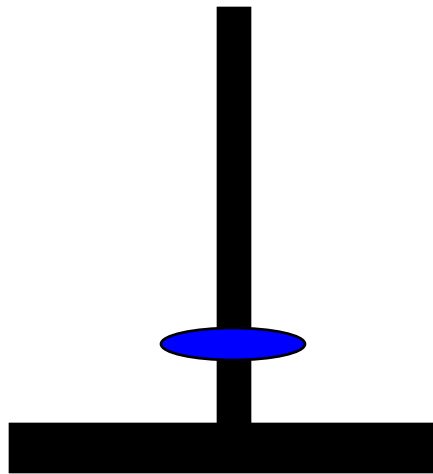
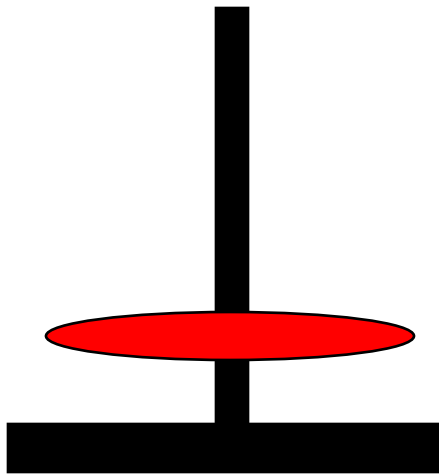
Towers of Hanoi



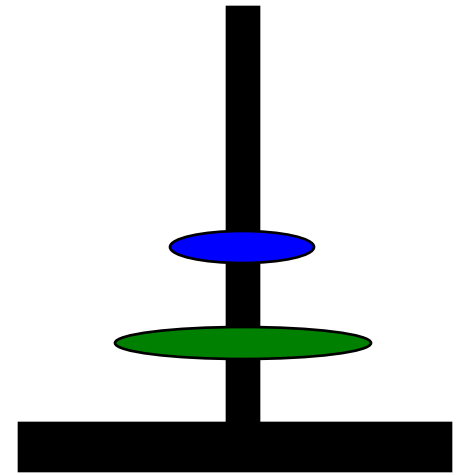
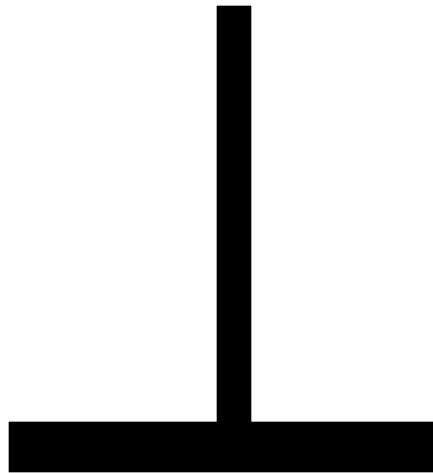
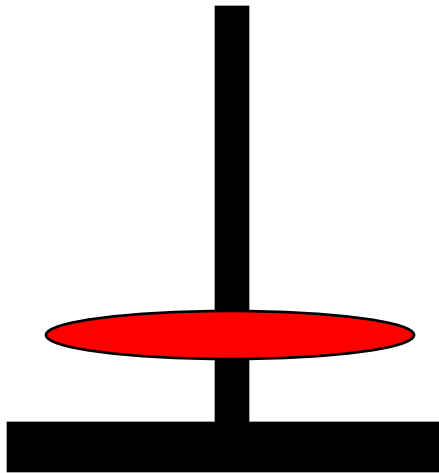
Towers of Hanoi



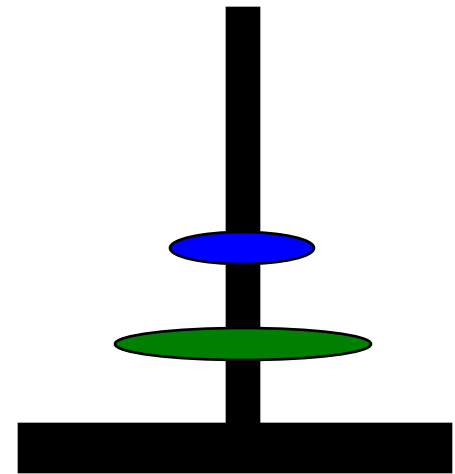
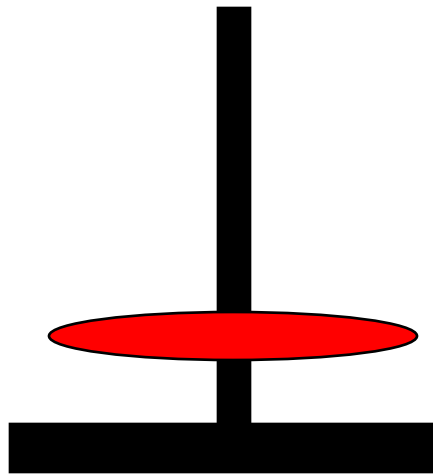
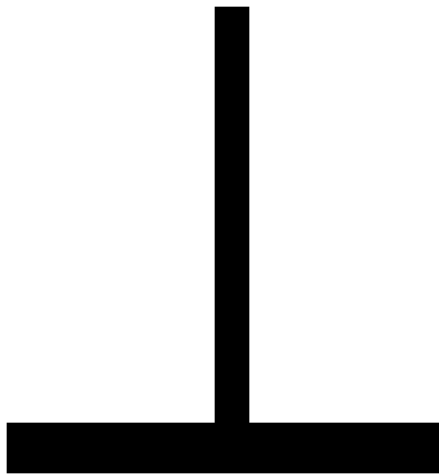
Towers of Hanoi



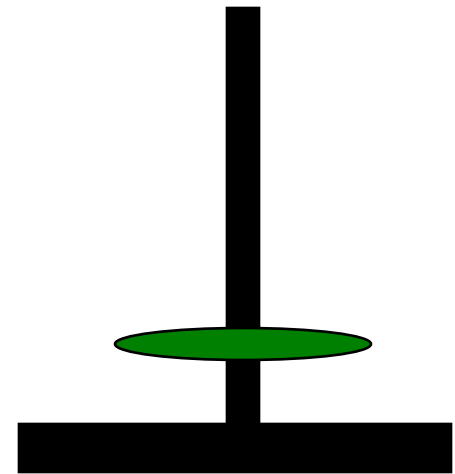
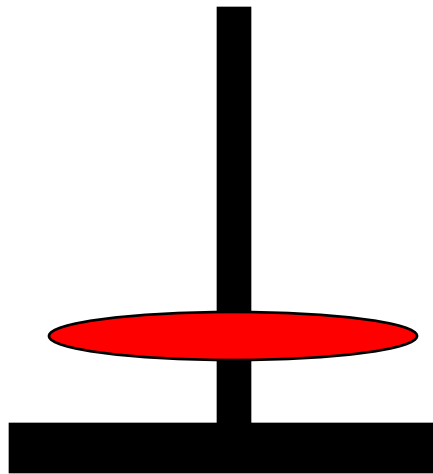
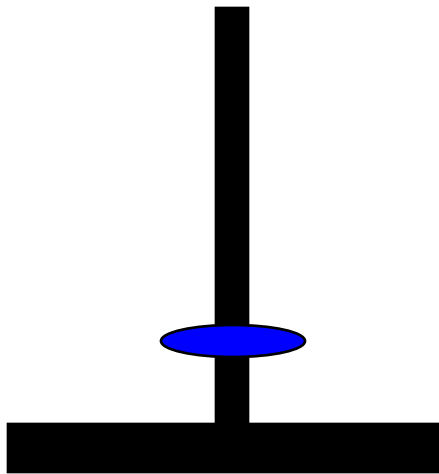
Towers of Hanoi



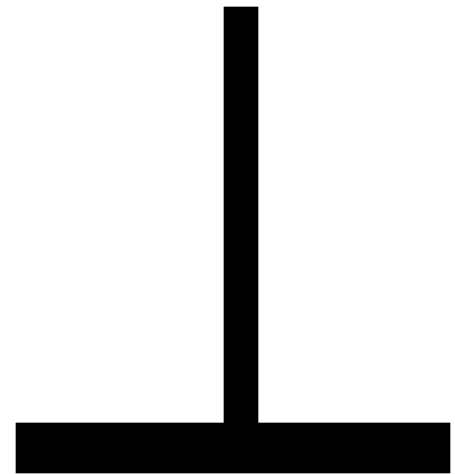
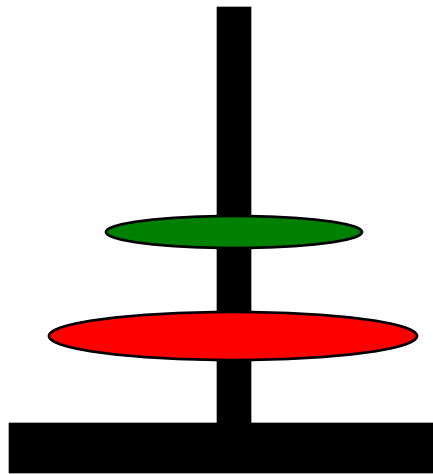
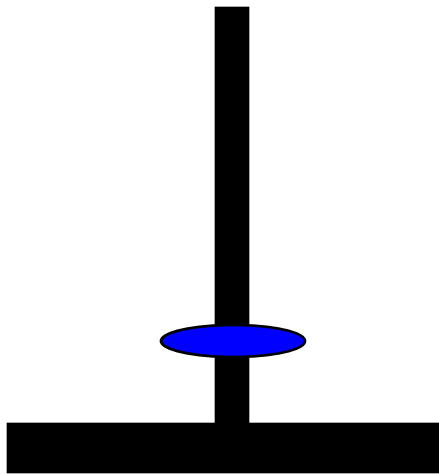
Towers of Hanoi



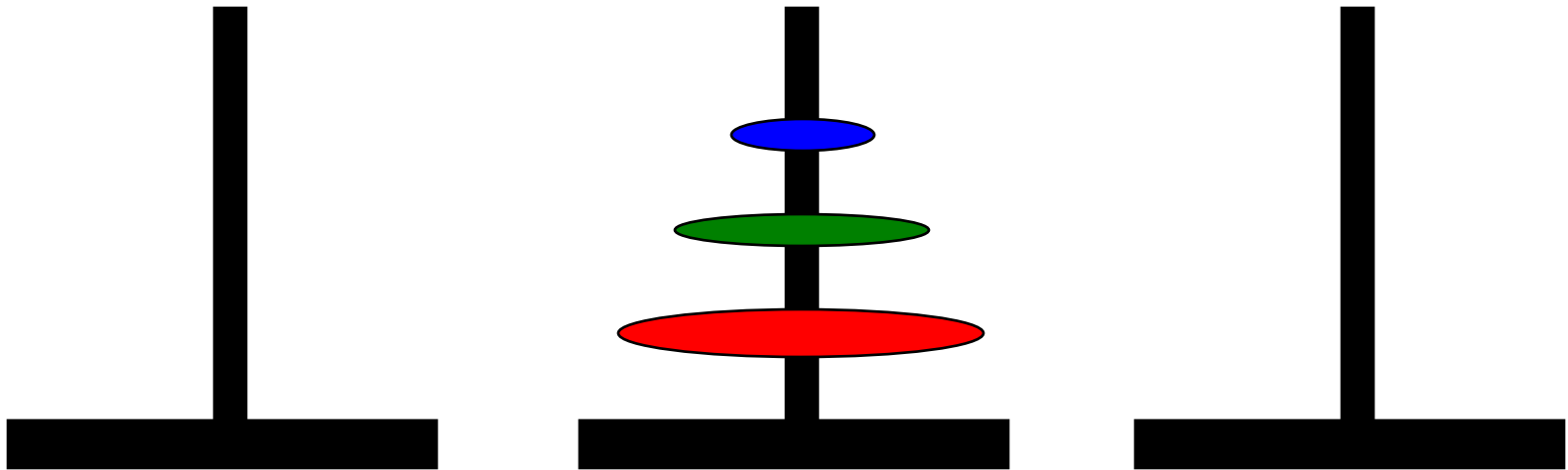
Towers of Hanoi



Towers of Hanoi



Towers of Hanoi



Analysis of Towers of Hanoi

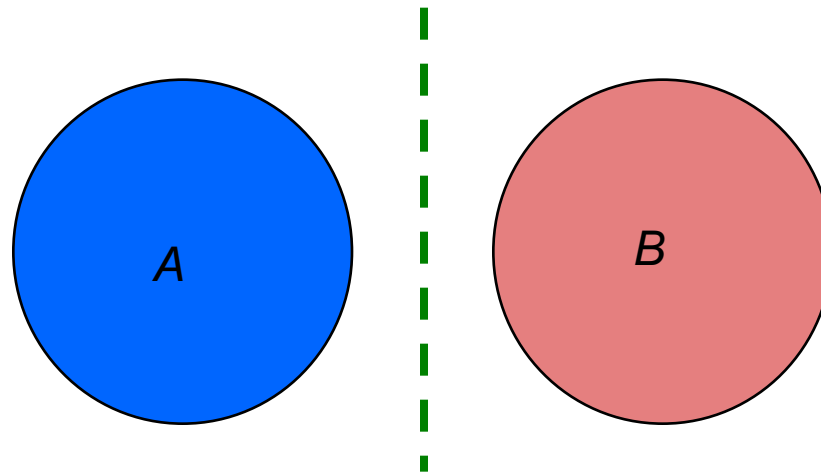
- Hypothesis --- it takes $2^n - 1$ moves to perform TowerHanoi(n, a, b, c) for all positive n .
- Proof:
- Basis: $P(1)$ – we can do it using move(a, b) i.e., $2^1 - 1 = 1$
- Inductive Hypothesis: $P(n)$ - it takes $2^n - 1$ moves to perform TowerHanoi(n, a, b, c)
- Inductive Step: In order to perform TowerHanoi($n+1, a, b, c$)
- We do: TowerHanoi(n, a, c, b), move(a, c), and TowerHanoi(n, c, b, a);
- Assuming the IH this all takes $2^n - 1 + 1 + 2^n - 1 = 2 \times 2^n - 1 = 2^{(n+1)} - 1$

Outline

- Recursive Algorithms: Towers of Hanoi
- **Basic Counting Rules**
 - Sum Rule
 - Product Rule
 - Generalized Product Rule

Sum Rule

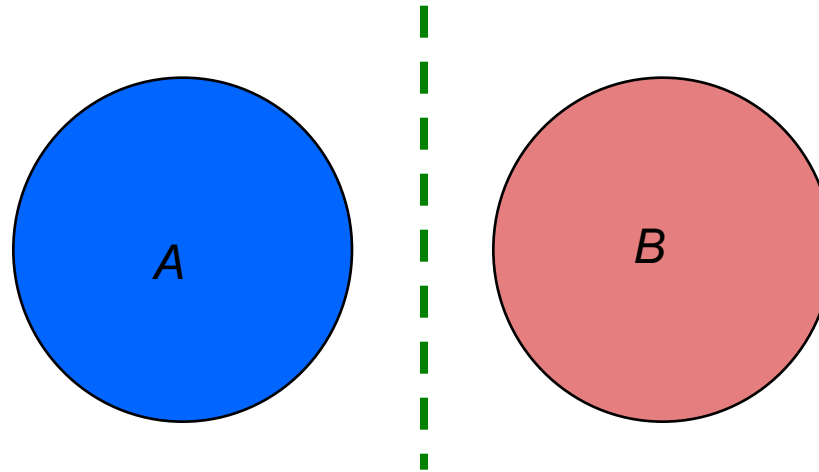
$|S|$: the number of elements in a set S .



If sets A and B are **disjoint**, then

$$|A \cup B| = |A| + |B|$$

Sum Rule



If sets A and B are **disjoint**, then

$$|A \cup B| = |A| + |B|$$

- Class has 43 women, 54 men, so total enrollment = $43 + 54 = 97$
- 26 lower case letters, 26 upper case letters, and 10 digits, so total characters = $26+26+10 = 62$

Product Rule

Recall that, given two sets A and B , the Cartesian product

$$A \times B = \{(a, b) \mid a \in A \text{ and } b \in B\}$$

Fact: If $|A| = n$ and $|B| = m$, then $|A \times B| = mn$.

$$A = \{a, b, c, d\}, \quad B = \{1, 2, 3\}$$

$$\begin{aligned} A \times B = \{ & (a, 1), (a, 2), (a, 3), \\ & (b, 1), (b, 2), (b, 3), \\ & (c, 1), (c, 2), (c, 3), \\ & (d, 1), (d, 2), (d, 3) \} \end{aligned}$$

Example: If there are 4 men and 3 women, there are

$$4 \times 3 = 12 \text{ possible married couples.}$$

Product Rule

Fact: If $|A| = n$ and $|B| = m$, then $|A \times B| = mn$.

In general let $A = \{a_1, a_2, a_3, \dots, a_m\}$ and $B = \{b_1, b_2, \dots, b_n\}$.

We can arrange the elements into a table as follows.

$$\begin{aligned} A \times B = & \{(a_1, b_1), (a_1, b_2), \dots, (a_1, b_n), \\ & (a_2, b_1), (a_2, b_2), \dots, (a_2, b_n), \\ & (a_3, b_1), (a_3, b_2), \dots, (a_3, b_n), \\ & \dots \\ & (a_m, b_1), (a_m, b_2), \dots, (a_m, b_n), \} \end{aligned}$$

There are m rows, and each row has n elements,
and so there are a total of mn elements.

Product Rule

Fact: $|A_1 \times A_2 \times \dots \times A_k| = |A_1| \times |A_2| \times \dots \times |A_k|$.

The formal proof uses mathematical induction.

But the proof idea is not difficult.

We think of $A_1 \times A_2 \times \dots \times A_k$ as $(\dots((A_1 \times A_2) \times A_3) \dots \times A_k)$.

That is, we first construct $A_1 \times A_2$, and it is a set of size $|A_1| \times |A_2|$.

Then, we construct $(A_1 \times A_2) \times A_3$, the product of $A_1 \times A_2$ and A_3 ,

and it is a set of size $(|A_1| \times |A_2|) \times |A_3|$ by the product rule on two sets.

Repeating the argument we can see that $|A_1 \times A_2 \times \dots \times A_k| = |A_1| \times |A_2| \times \dots \times |A_k|$.

Example: Counting Strings

What is the number of 10-bit strings?

Let $B = \{0, 1\}$.

The set of 2-bit strings is just $B \times B$.

The set of 10-bit strings is just $B \times B \times B \times B \times B \times B \times B \times B \times B \times B$, denoted by B^{10} .

By the product rule, $|B \times B| = |B| \times |B| = 2 \times 2 = 4$, and

$$|B^{10}| = |B| \times |B| \times |B| \times |B| \times |B| \times |B| \times |B| \times |B| \times |B| \times |B| = |B|^{10} = 2^{10} = 1024.$$

Example: IP Addresses

What is the number of IP addresses?

An IP address is of the form 192.168.0.123.

There are four numbers, each is between 0 and 255.

Let $B = \{0, 1, \dots, 255\}$.

Then the set of IP addresses is just B^4 .

By the product rule, $|B^4| = |B|^4 = 256^4 = 4294967296$.

Example: Product Rule

In general we have:

The number of length- n strings from an *alphabet* of size m is m^n .

That is, $|B^n| = |B|^n$.

e.g. the number of length- n binary strings is 2^n

the number of length- n strings formed by capital letters is 26^n

Example: Counting Passwords

How many passwords satisfy the following requirements?

- between 6 & 8 characters long
- starts with a letter
- case sensitive
- other characters: digits or letters

First we define the set of letters and the set of digits.

$$L = \{a, b, \dots, z, A, B, \dots, Z\}$$

$$D = \{0, 1, \dots, 9\}$$

Example: Counting Passwords

$$L ::= \{a, b, \dots, z, A, B, \dots, Z\}$$

$$D ::= \{0, 1, \dots, 9\}$$

We first count the number of passwords with a specific length.

Let P_n be the set of passwords with length n .

$$\begin{aligned} P_6 &= L \times (L \cup D) \times (L \cup D) \times (L \cup D) \times (L \cup D) \times (L \cup D) \\ &= L \times (L \cup D)^5 \end{aligned}$$

$$\begin{aligned} P_n &::= \text{length } n \text{ passwords} \\ &= L \times (L \cup D)^{n-1} \end{aligned}$$

Example: Counting Passwords

$$\left| L \times (L \cup D)^{n-1} \right| = |L| \cdot |L \cup D|^{n-1} \quad \text{by product rule}$$

$$= |L| \cdot (|L| + |D|)^{n-1} \quad \text{by sum rule}$$

$$= 52 \cdot 62^{n-1}$$

The set of Passwords:

$$P = P_6 \cup P_7 \cup P_8$$

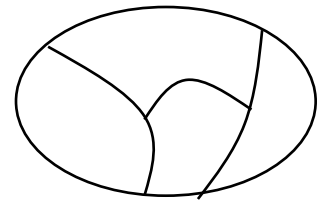
$$|P| = |P_6| + |P_7| + |P_8|$$

$$= 52 \cdot 62^5 + 52 \cdot 62^6 + 52 \cdot 62^7$$

$$= 186125210680448$$

$$\approx 19 \cdot 10^{14}$$

counting by partitioning



This is a common technique.

Divide the set into disjoint subsets.

Count each subset and add the answers.

At Least One Seven

How many # 4-digit numbers with at least one 7?

Method 1: count by *1st occurrence of 7*:

The set of 4-digit numbers with 7 in the first digit.

The set of 4-digit numbers with 7 in the second digit, but the first digit is not 7, and so on.

$$7xxx + o7xx + oo7x + ooo7$$

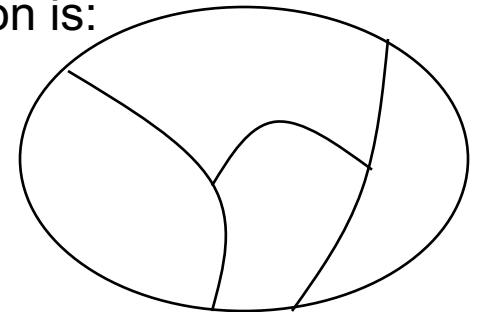
where x represents any digit from 1 to 10, while o represent any digit from 1 to 10 except 7.

Clearly, each number containing at least one 7 is in one of the above sets, and these sets are disjoint.

Therefore, the answer to the question is:

$$10^3 + 9 \cdot 10^2 + 9^2 \cdot 10 + 9^3 = 3439$$

(counting by partitioning)



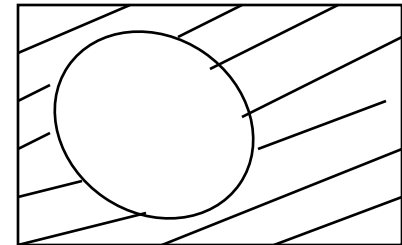
At Least One Seven

How many # 4-digit numbers with at least one 7?

Method 2:

$$\begin{aligned} |\text{4-digit numbers with at least one 7}| &= \\ |\text{4-digit numbers}| - |\text{those with no 7s}| \\ &= 10^4 - 9^4 \\ &= 3439 \end{aligned}$$

(counting the complement)



Counting the complement is a useful technique.

Defective Dollars



A dollar is defective if some digit appears more than once in the 6-digit serial number.

How common are nondefective dollars?

Defective Dollars

How common are nondefective dollars?

10 possible choices for the first digit,
9 possible choices for the second digit, and so on...

So, there are $10 \times 9 \times 8 \times 7 \times 6 \times 5$
= 151200 serial number with all its digit different

There are totally $10^6 = 1000000$ serial numbers.

So, only about 15% of dollars are nondefective.

Generalized Product Rule

Q a set of length- k sequences. If there are:

n_1 possible 1st elements in sequences,

n_2 possible 2nd elements for each first entry,

n_3 possible 3rd elements for each 1st & 2nd,

...

then, $|Q| = n_1 \cdot n_2 \cdot n_3 \cdot \dots \cdot n_k$

Next class

- Topic: The Pigeonhole Principle
- Pre-class reading: Chap 6.2

