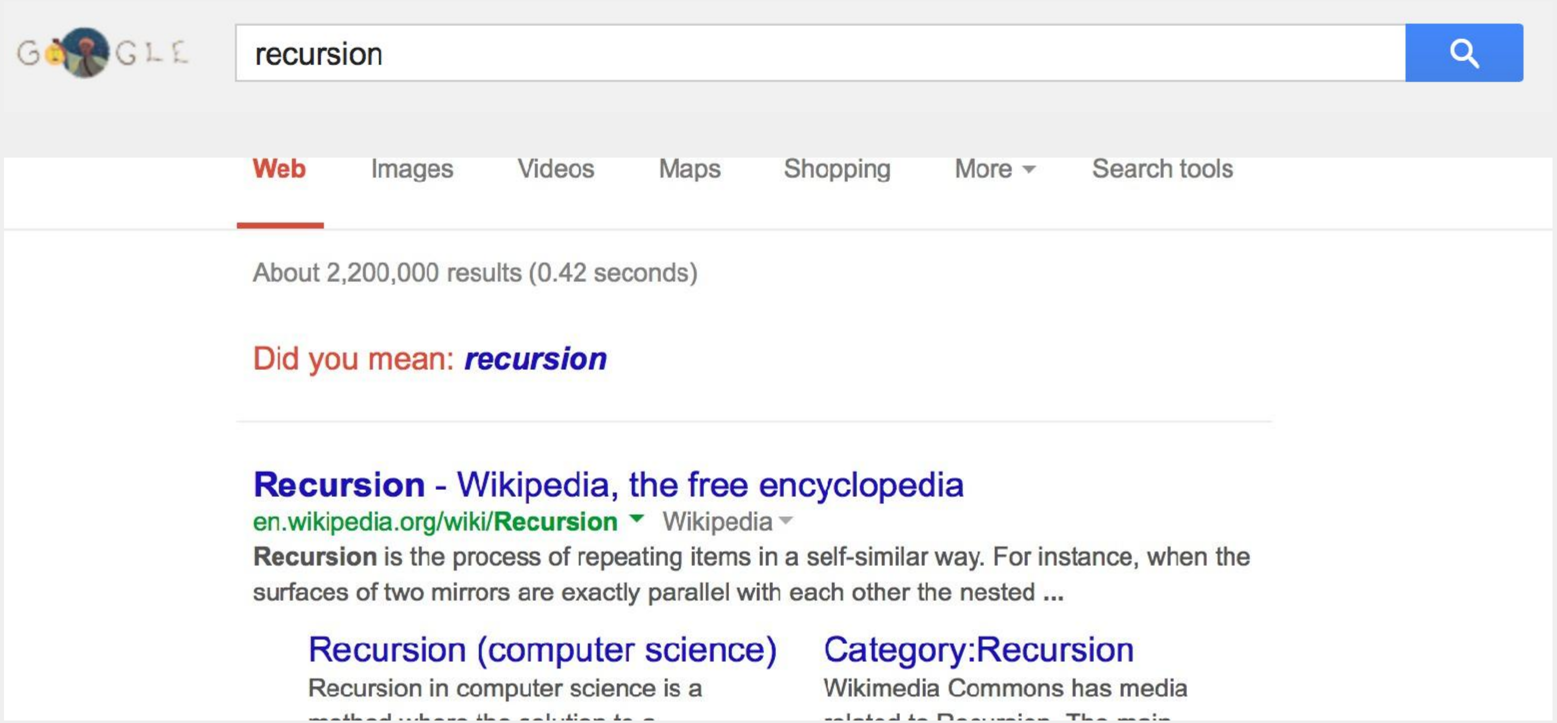
Data Structures and Algorithms

Khazhak Galstyan

Session 2: Recursion

What is Recursion?



recursion

Web Images Videos Maps Shopping More Search tools

About 2,200,000 results (0.42 seconds)

Did you mean: ***recursion***

Recursion - Wikipedia, the free encyclopedia
en.wikipedia.org/wiki/**Recursion** Wikipedia
Recursion is the process of repeating items in a self-similar way. For instance, when the surfaces of two mirrors are exactly parallel with each other the nested **...**

Recursion (computer science)
Recursion in computer science is a

Category:Recursion
Wikimedia Commons has media

What is Recursion?

A problem solving technique in which problems are solved by reducing them to smaller problems of the same form.

Motivations

Suppose you want to find all the files under a directory that contains a particular word. How do you solve this problem? There are several ways to solve this problem. An intuitive solution is to use recursion by searching the files in the subdirectories recursively.

Recursion In Real Life

How many students total are directly behind you in your "column" of the classroom?

Rules:

You can see only the people directly in front and behind you. So, you can't just look back and count. You are allowed to ask questions of the people in front / behind you.

How can we solve this problem recursively?

Recursion In Real Life

Answer:

1. The first person looks behind them, and sees if there is a person there. If not, the person responds "0".
2. If there is a person, repeat step 1, and wait for
   a response.
3. Once a person receives a response, they add 1 for the person behind them, and they respond to the person that asked them.

Three Musts of Recursion

1. Your code must have a case for all valid inputs

Three Musts of Recursion

1. Your code must have a case for all valid inputs

2. You must have a base case that makes no recursive calls

Three Musts of Recursion

1. Your code must have a case for all valid inputs
2. You must have a base case that makes no recursive calls
3. When you make a recursive call it should be to a simpler instance and make forward progress towards the base case.

Computing Factorial

```
factorial(0) = 1;
factorial(n) = n*factorial(n-1);
```

Computing Factorial

factorial(4)

```
factorial(0) = 1;
factorial(n) = n*factorial(n-1);
```

Computing Factorial

```
factorial(0) = 1;
factorial(n) = n*factorial(n-1);
```

factorial(4) = 4 * factorial(3)

Computing Factorial

factorial(0) = 1;

factorial(n) = n*factorial(n-1);

```
factorial(4) = 4 * factorial(3)
             = 4 * 3 * factorial(2)
```

Computing Factorial

factorial(0) = 1;
factorial(n) = n*factorial(n-1);

factorial(4) = 4 * factorial(3)
= 4 * 3 * factorial(2)
= 4 * 3 * (2 * factorial(1))

Computing Factorial

factorial(0) = 1;

factorial(n) = n*factorial(n-1);

factorial(4) = 4 * factorial(3)

= 4 * 3 * factorial(2)

= 4 * 3 * (2 * factorial(1))

= 4 * 3 * (2 * (1 * factorial(0)))

Computing Factorial

```
factorial(0) = 1;
factorial(n) = n*factorial(n-1);
```

```
factorial(4) = 4 * factorial(3)
             = 4 * 3 * factorial(2)
             = 4 * 3 * (2 * factorial(1))
             = 4 * 3 * ( 2 * (1 * factorial(0)))
             = 4 * 3 * ( 2 * ( 1 * 1)))
```

Computing Factorial

factorial(0) = 1;

factorial(n) = n*factorial(n-1);

```
factorial(4) = 4 * factorial(3)
             = 4 * 3 * factorial(2)
             = 4 * 3 * (2 * factorial(1))
             = 4 * 3 * ( 2 * (1 * factorial(0)))
             = 4 * 3 * ( 2 * ( 1 * 1)))
             = 4 * 3 * ( 2 * 1)
```

Computing Factorial

factorial(0) = 1;

factorial(n) = n*factorial(n-1);

```
factorial(4) = 4 * factorial(3)
             = 4 * 3 * factorial(2)
             = 4 * 3 * (2 * factorial(1))
             = 4 * 3 * ( 2 * (1 * factorial(0)))
             = 4 * 3 * ( 2 * ( 1 * 1)))
             = 4 * 3 * ( 2 * 1)
             = 4 * 3 * 2
```

Computing Factorial

```
factorial(4) = 4 * factorial(3)
             = 4 * 3 * factorial(2)
             = 4 * 3 * (2 * factorial(1))
             = 4 * 3 * ( 2 * (1 * factorial(0)))
             = 4 * 3 * ( 2 * ( 1 * 1)))
             = 4 * 3 * ( 2 * 1)
             = 4 * 3 * 2
             = 4 * 6
```

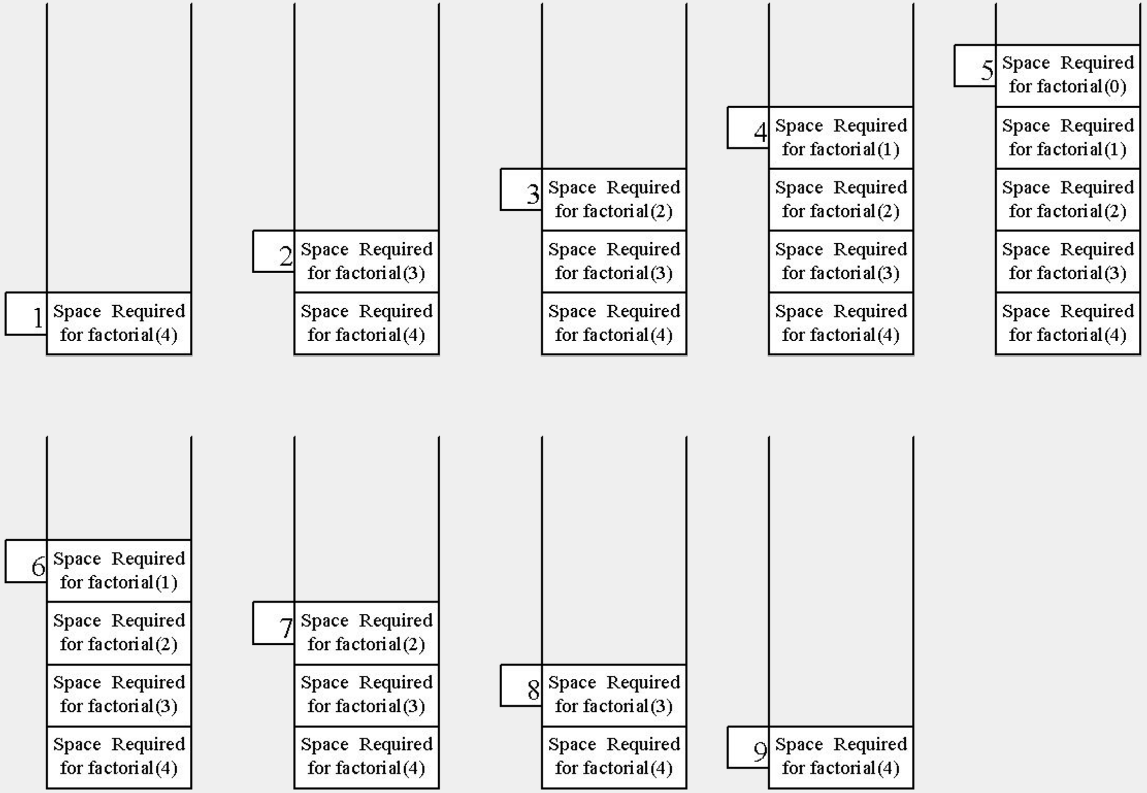
```
factorial(0) = 1;
factorial(n) = n*factorial(n-1);
```

Computing Factorial

```
factorial(4) = 4 * factorial(3)
             = 4 * 3 * factorial(2)
             = 4 * 3 * (2 * factorial(1))
             = 4 * 3 * ( 2 * (1 * factorial(0)))
             = 4 * 3 * ( 2 * ( 1 * 1)))
             = 4 * 3 * ( 2 * 1)
             = 4 * 3 * 2
             = 4 * 6
             = 24
```

```
factorial(0) = 1;
factorial(n) = n*factorial(n-1);
```

factorial(4) Stack Trace



1
Space Required for factorial(4)
2
Space Required for factorial(3)
Space Required for factorial(4)
3
Space Required for factorial(2)
Space Required for factorial(3)
Space Required for factorial(4)
4
Space Required for factorial(1)
Space Required for factorial(2)
Space Required for factorial(3)
Space Required for factorial(4)
5
Space Required for factorial(0)
Space Required for factorial(1)
Space Required for factorial(2)
Space Required for factorial(3)
Space Required for factorial(4)
6
Space Required for factorial(1)
Space Required for factorial(2)
Space Required for factorial(3)
Space Required for factorial(4)
7
Space Required for factorial(2)
Space Required for factorial(3)
Space Required for factorial(4)
8
Space Required for factorial(3)
Space Required for factorial(4)
9
Space Required for factorial(4)

Fibonacci Numbers

```
Finonacci series: 0 1 1 2 3 5 8 13 21 34 55 89…
         indices: 0 1 2 3 4 5 6 7  8  9  10 11
```

fib(0) = 0;

fib(1) = 1;

fib(index) = fib(index -1) + fib(index -2); index >=2

fib(3) = fib(2) + fib(1) = (fib(1) + fib(0)) + fib(1) = (1 + 0) +fib(1) = 1 + fib(1) = 1 + 1 = 2

Examples!

The power() function:

Write a recursive function that takes in a number (x) and an exponent (n) and returns the result of x^n

Examples!

Recursion is about solving a small piece of a large problem.

- What is 69743 in binary?
  - Do we know anything about its representation in binary?
- Case analysis:
  - What is/are easy numbers to print in binary?
- Can we express a larger number in terms of a smaller number(s)?

Examples!

Suppose we are examining some arbitrary integer N.

N's binary representation is 100101

- (N / 2)'s binary representation is _____
- (N % 2)'s binary representation is _____

Examples!

Suppose we are examining some arbitrary integer N.

N's binary representation is 100101

- (N / 2)'s binary representation is 10010
- (N % 2)'s binary representation is _____

Examples!

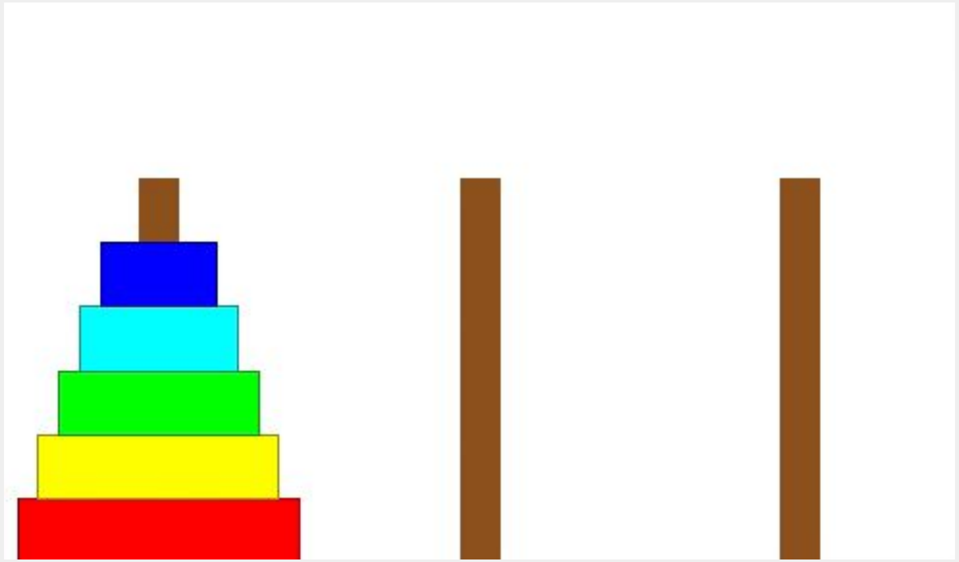
Suppose we are examining some arbitrary integer N.

N's binary representation is 100101

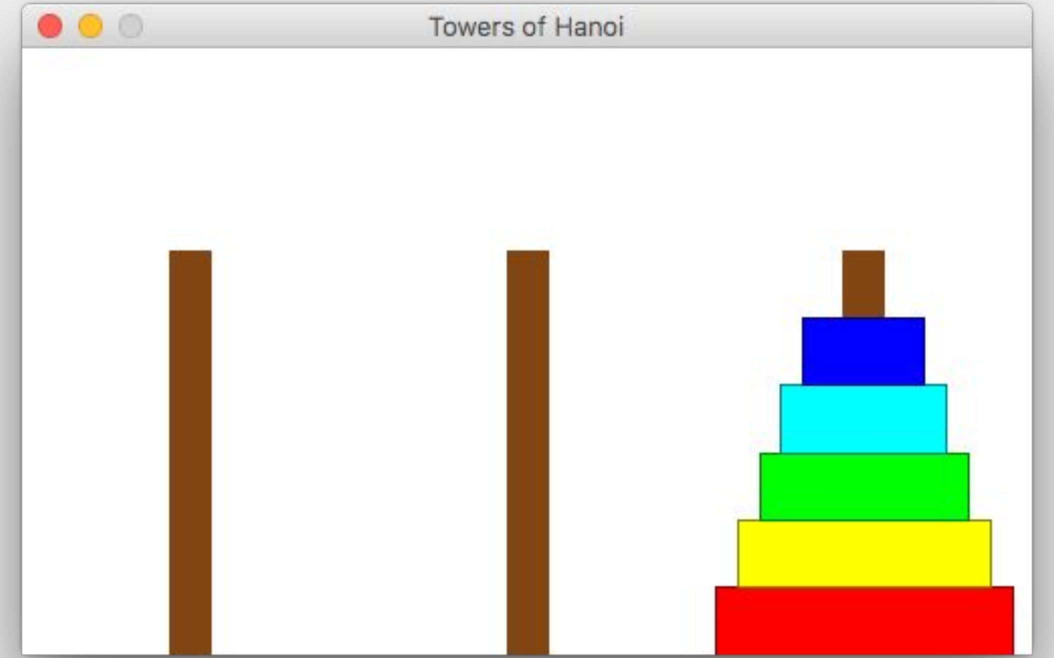
- (N / 2)'s binary representation is 10010
- (N % 2)'s binary representation is 1

What can we infer from this relationship?

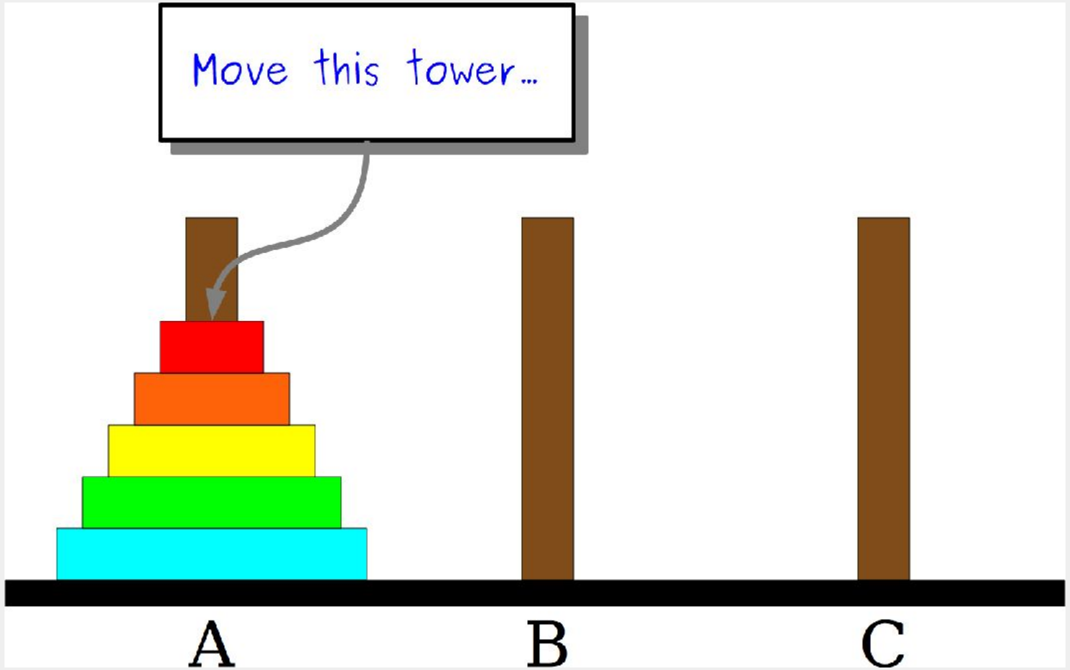
The Towers of Hanoi Puzzle



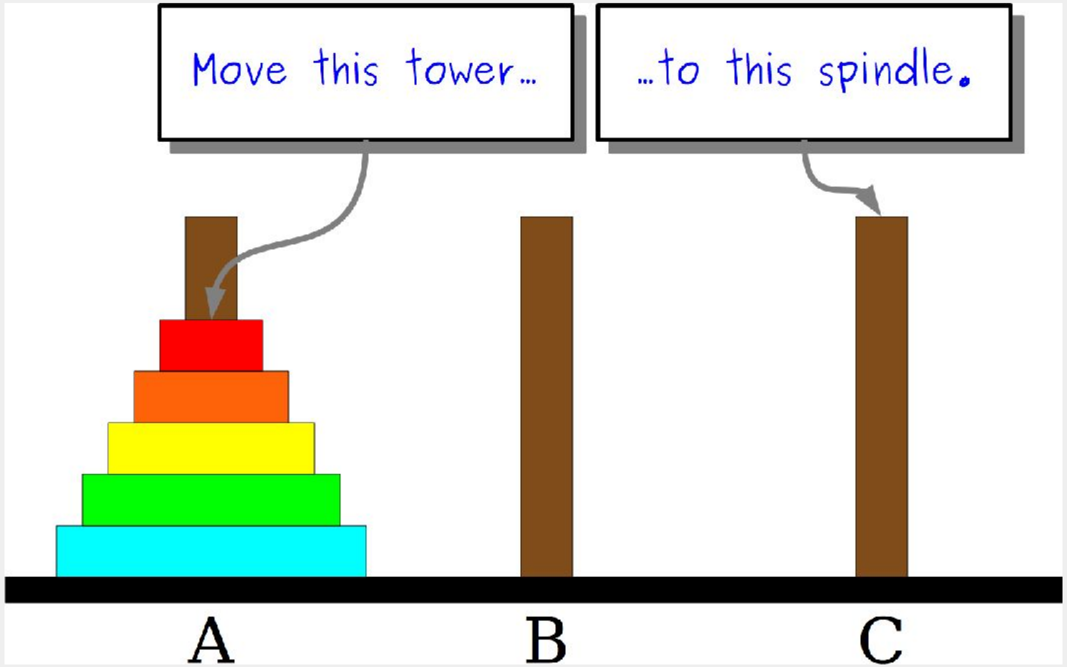
By the end of today, we will be able to write this program, and you may talk about the algorithm in section



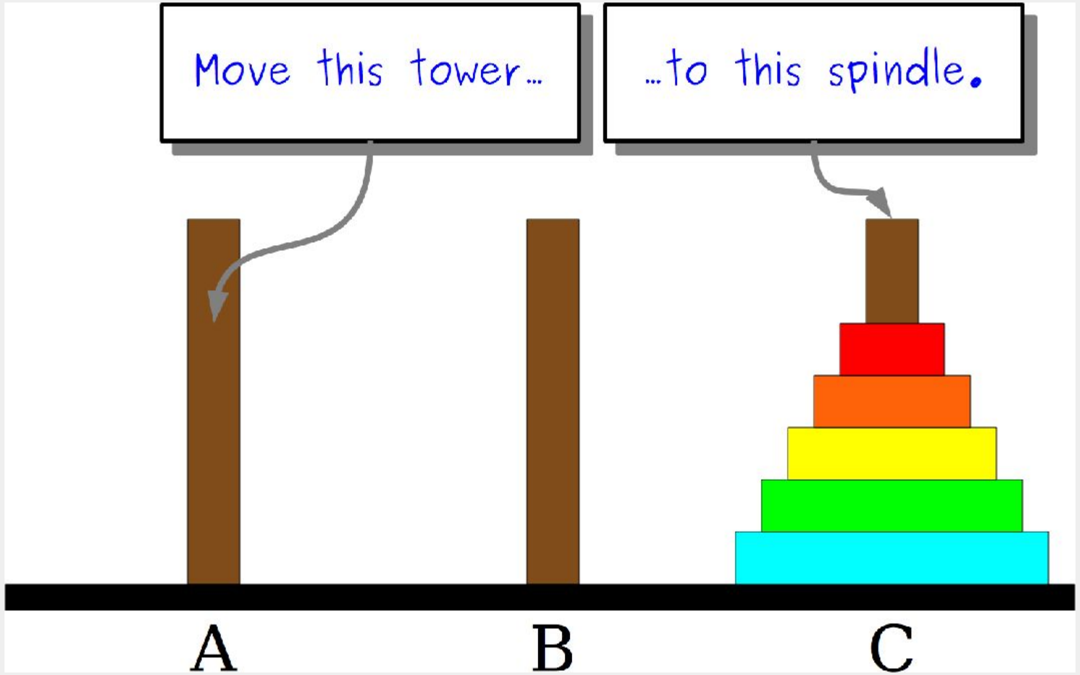
Here is the way the game is played



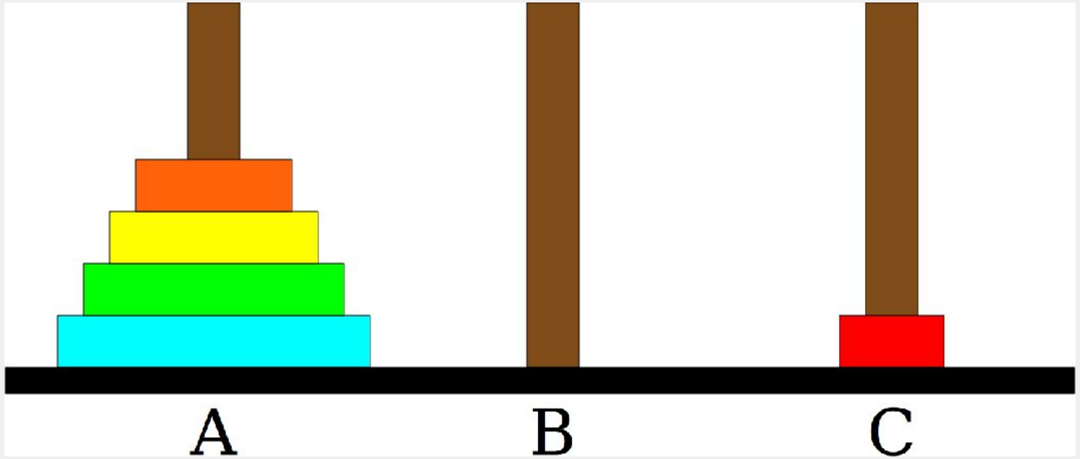
Here is the way the game is played



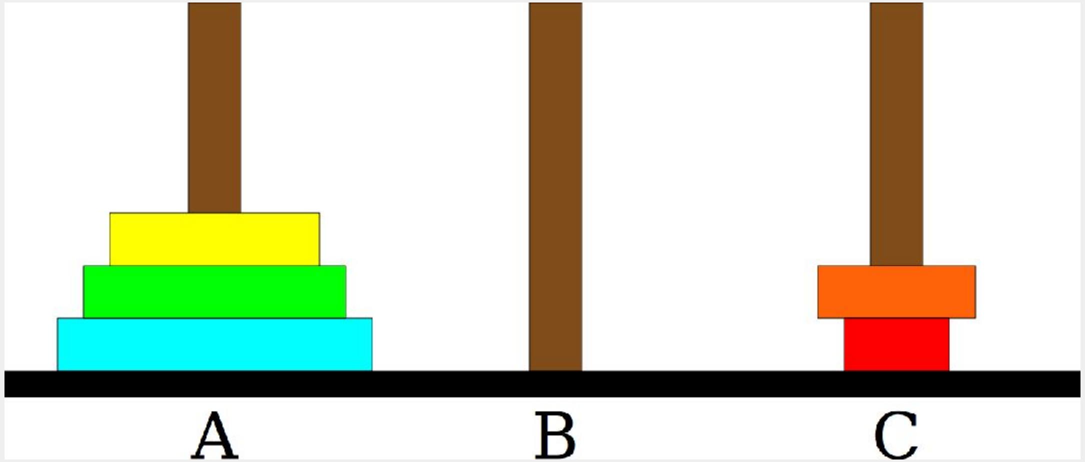
Here is the way the game is played



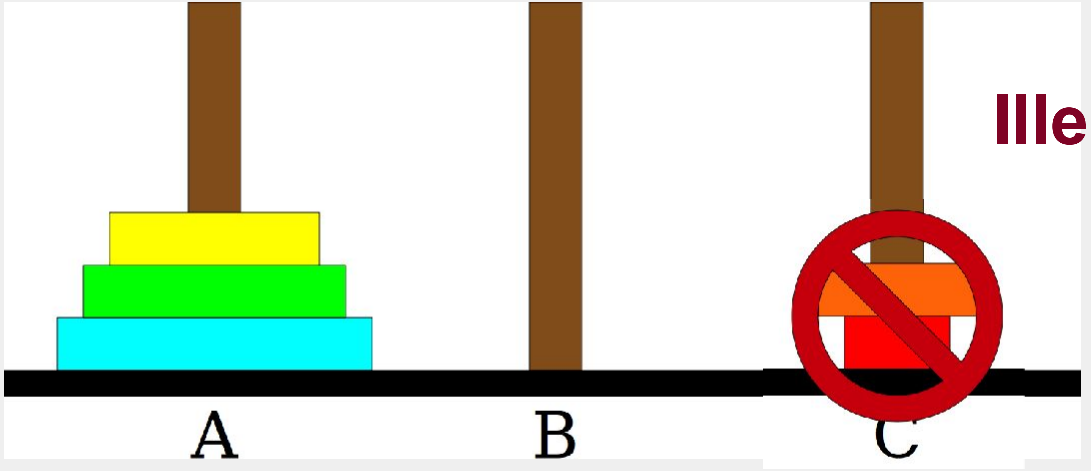
Here is the way the game is played



Here is the way the game is played

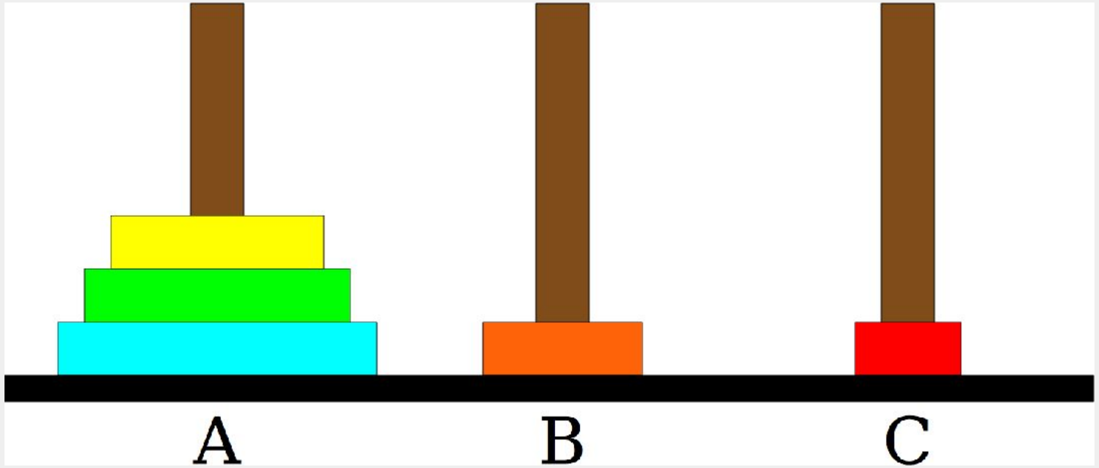


Here is the way the game is played

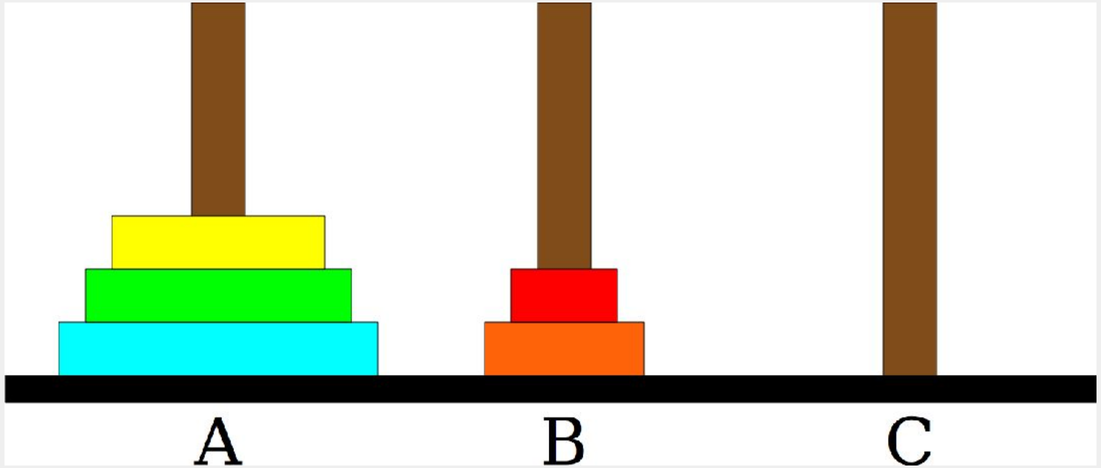


Illegal move!

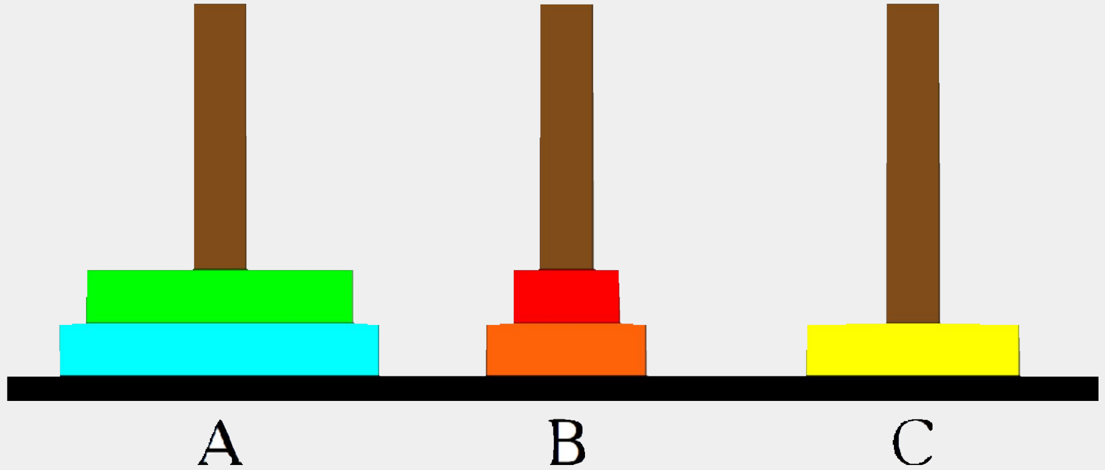
Here is the way the game is played



Here is the way the game is played

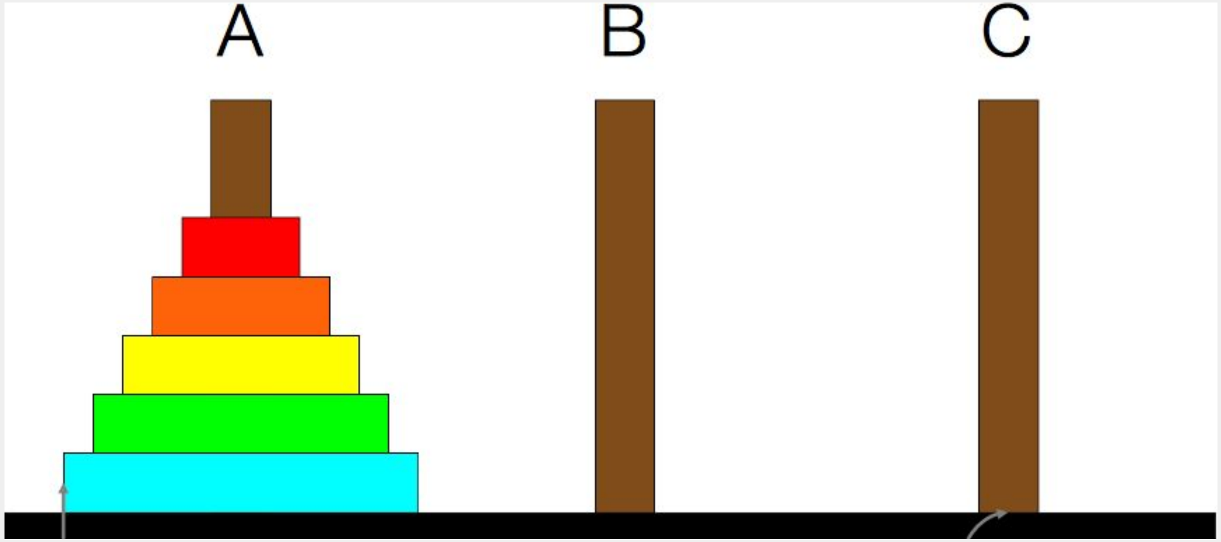


Here is the way the game is played

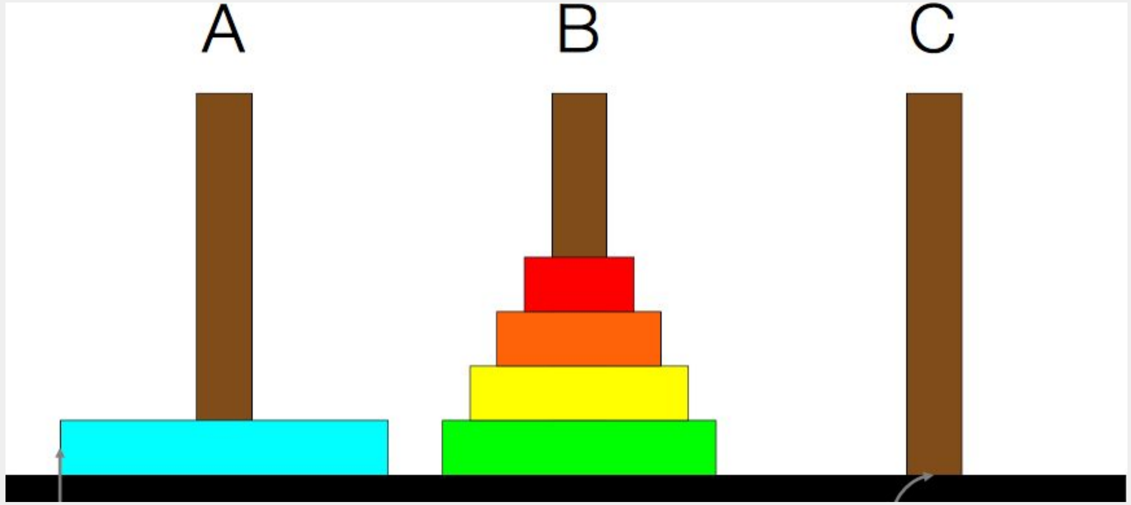


etc.

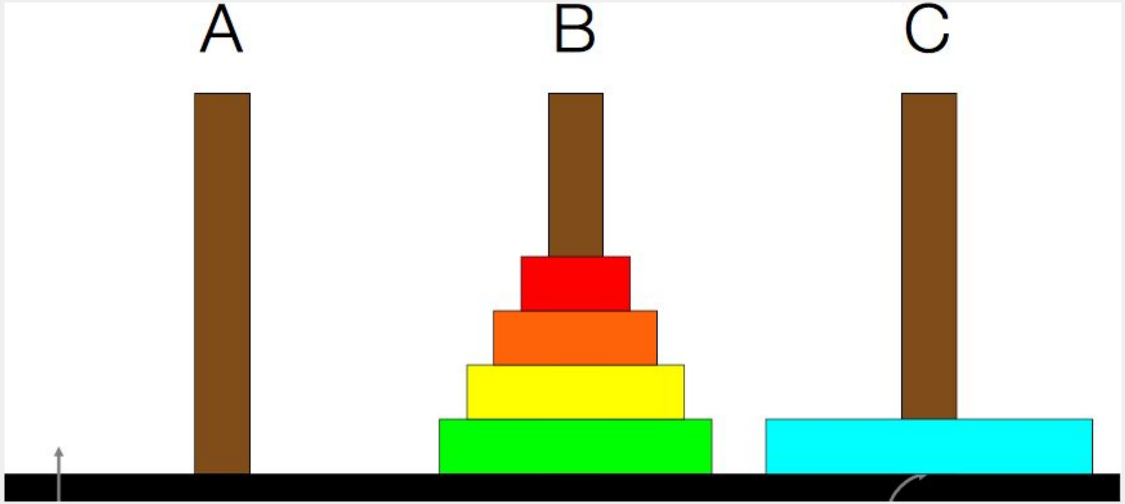
This is a hard problem to solve iteratively, but can be done recursively (though the recursive insight is not trivial to figure out)



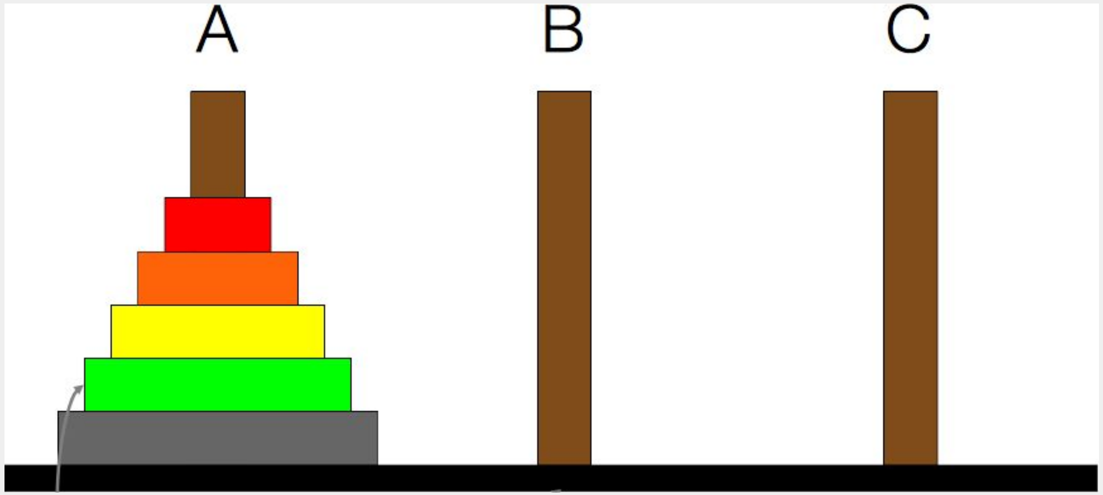
How to solve the Towers of Hanoi



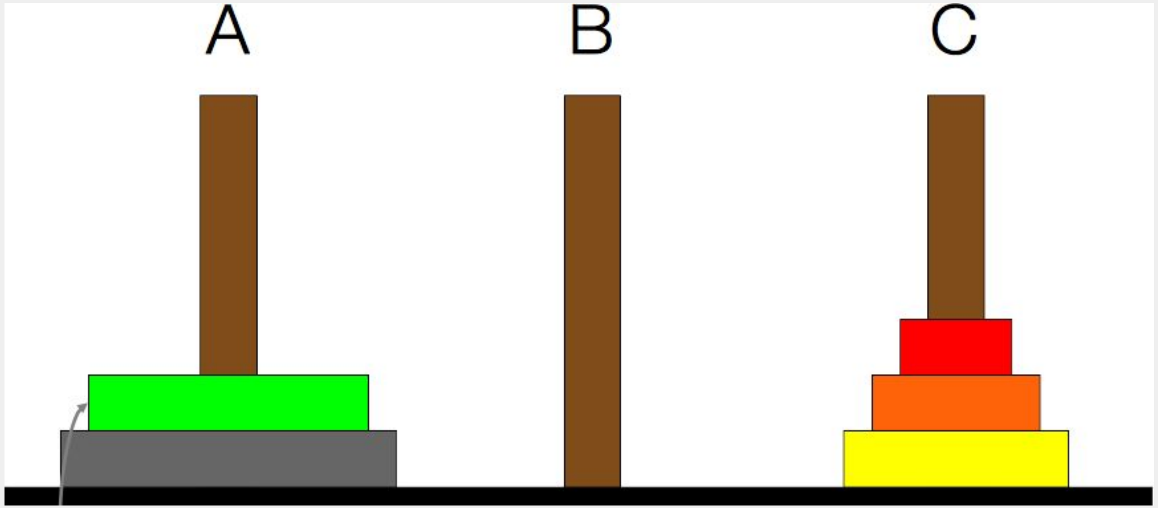
How to solve the Towers of Hanoi



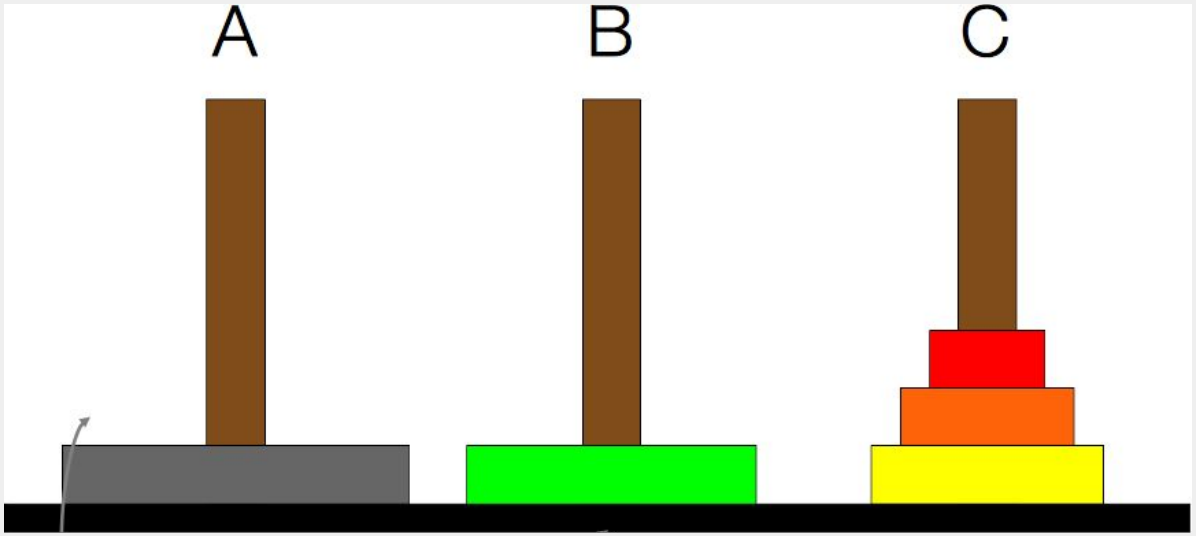
How to solve the Towers of Hanoi



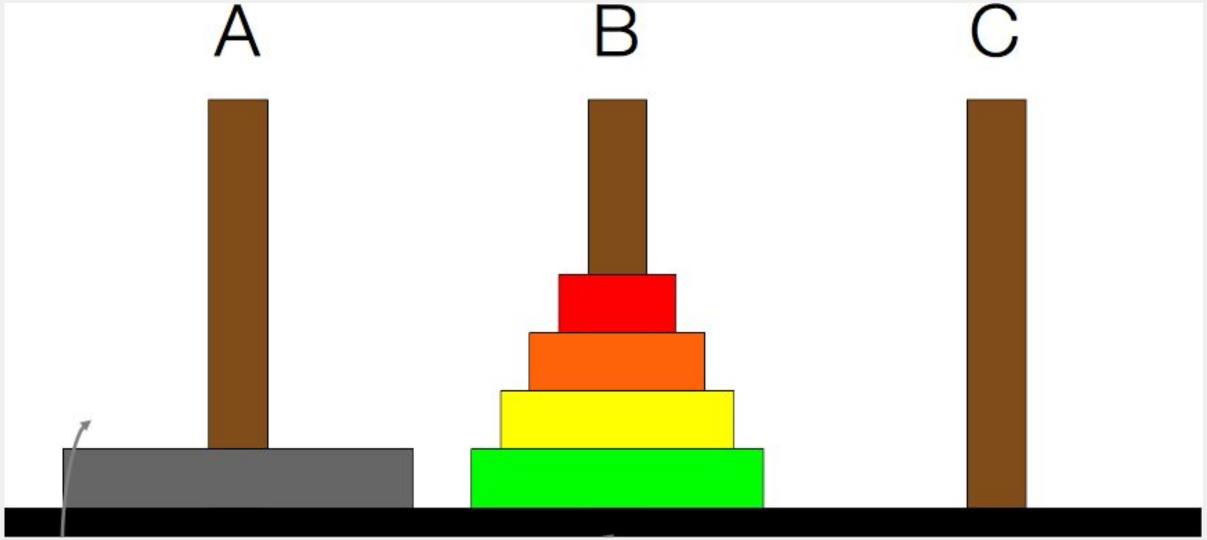
How to solve the Towers of Hanoi



How to solve the Towers of Hanoi



How to solve the Towers of Hanoi



How to solve the Towers of Hanoi

We need to find a very simple case that we can solve directly in order for the recursion to work.

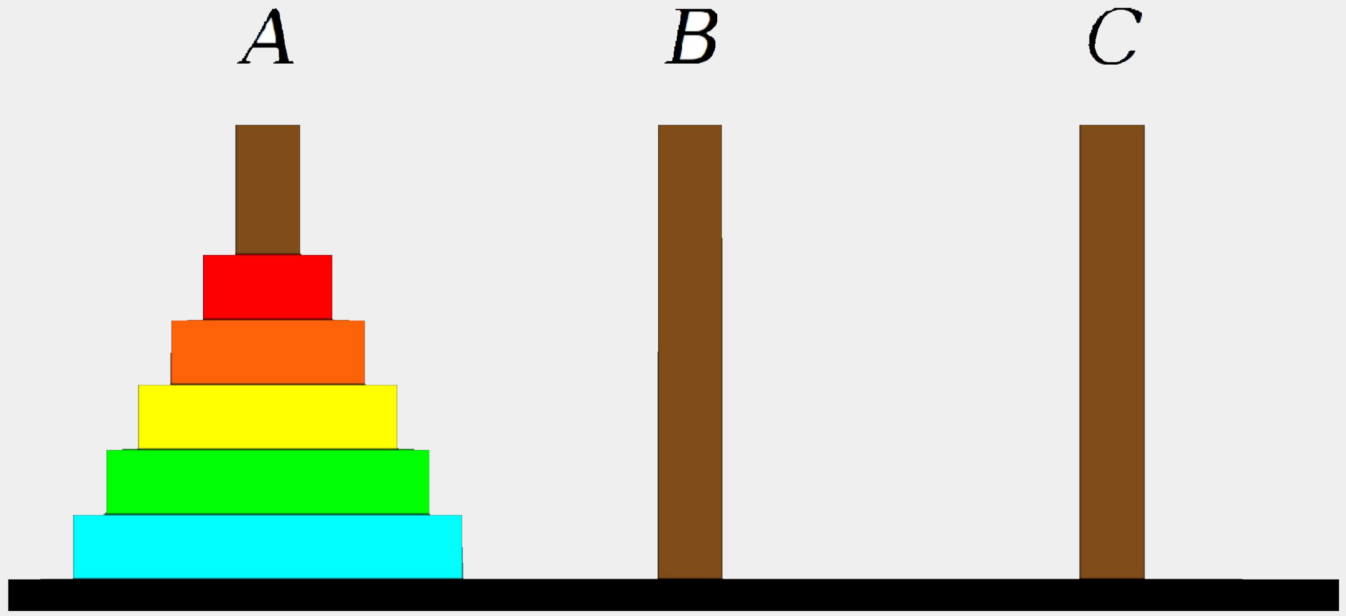
If the tower has size one, we can just move that single disk from the source to the destination.

If the tower has more than one, we have to use the auxiliary spindle.

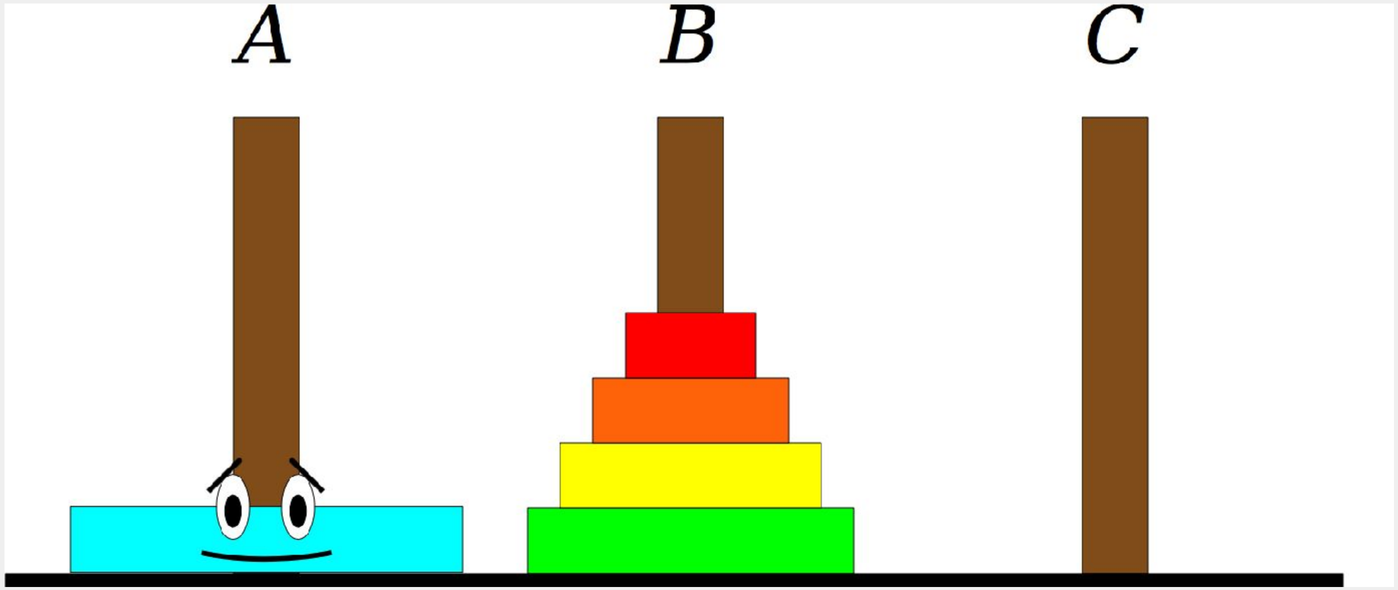
How to solve the Towers of Hanoi

We can break the entire process down into very simple steps -- not necessarily easy to think of steps, but simple ones!

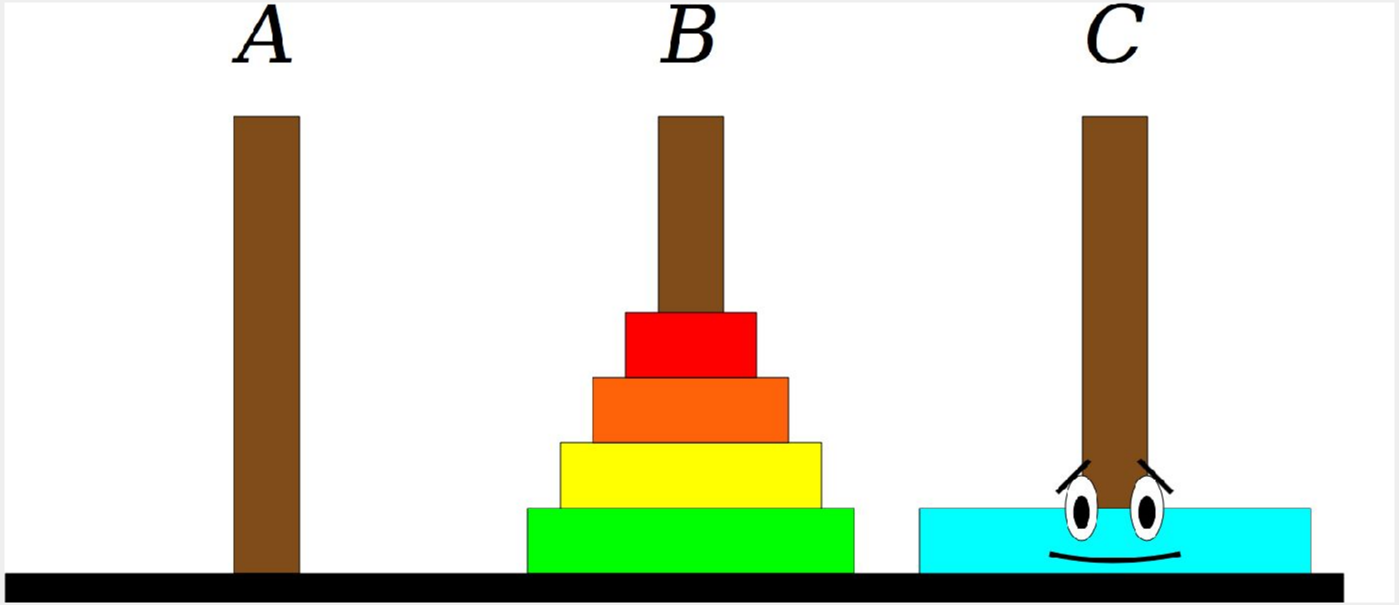
How to solve the Towers of Hanoi



How to solve the Towers of Hanoi



How to solve the Towers of Hanoi



How to solve the Towers of Hanoi