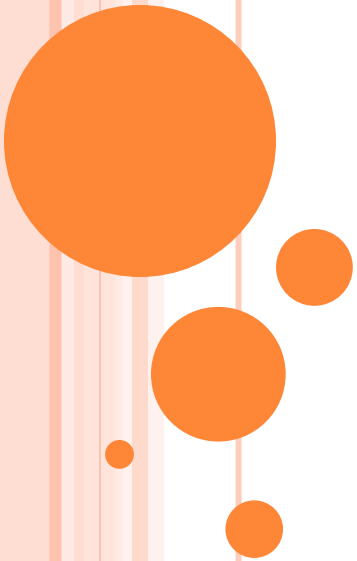


Recursion

Tanjina Helaly

Recursion

- The process in which a function calls itself

Factorial

- n! = n.(n-1).(n-2).....3.2.1
- 5! = 5.4.3.2.1
- 2! = 2.1
- 1! = 1
- 0! = ?
- <0 → factorial is only applicable for a number equal to or greater than 0.

Factorial

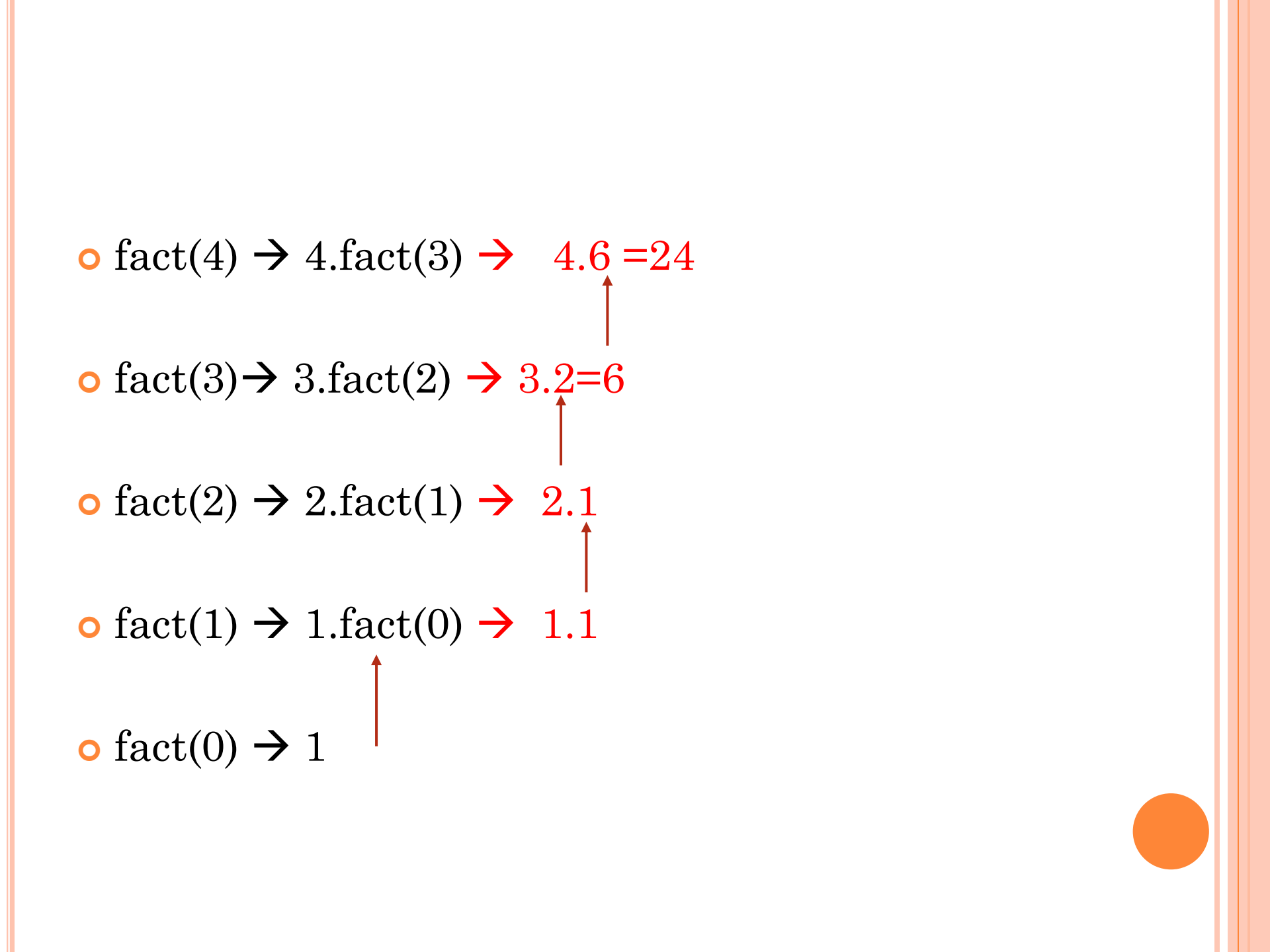
- Let's write a function fact(n) for n>=0
- Recursion
  - n! = n.(n-1).(n-2).....2.1
  - (n-1)! = (n-1).(n-2).....2.1
  - So, **n! = n.(n-1)! [recursive]**
  - Does it mean we can write fact(n) = n*fact(n-1) for all n?
  - When should we stop?
    - Try fact(1) or negative numbers
      - 1! = 1.0!
      - 0! = 0.(-1)! [0.anything = 0, also (-1)! Is not defined]

Recursion equation of Factorial

$$n! = \begin{cases} n.(n-1)! & if\ n \geq 1 \\ 1 & if\ n = 0 \end{cases}$$

$$fact(n) = \begin{cases} n.fact(n-1) & if\ n \geq 1 \\ 1 & if\ n = 0 \end{cases}$$

- fact(4) → 4.fact(3) → 4.6 =24
- fact(3)→ 3.fact(2) → 3.2=6
- fact(2) → 2.fact(1) → 2.1
- fact(1) → 1.fact(0) → 1.1
- fact(0) → 1



Some recursion examples

- Power $a^n = a.a^{n-1}$

```
pow(a, n)
        if n =0 return 1;
        return a*pow(a,n-1);
```

- Arithmetic series = 1+2+3+…(n-1)+n
  - AS(n) = n + AS(n-1)

```
ArithSum(n)
        if n =0 return 0;
        return n+ArithSum(n-1);
```

Some recursion examples

- Finding an element
  - FE(Arr, n, val)

    if n == 0 return -1;

    if Arr[n] == val return n;

    return FE(n-1);

```
Fib(n)
    Res = Fin(n-1)+Fib(n-2)
    if n== 0
       res = 1;
    else
       res = n*Fact(n-1)
    return res; // when will this line execute?
```

F(3)→F(2)→F(1)→ 3.2.F(1)→3.2.1.F(0)

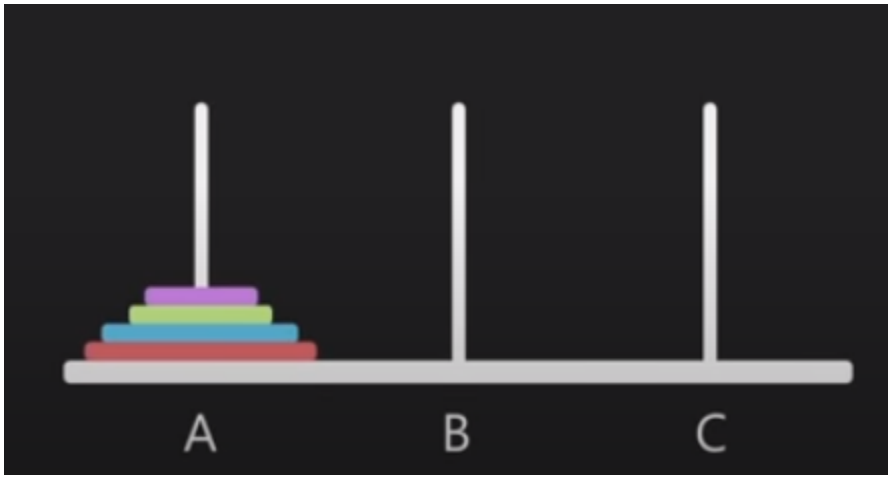
Tower of Hanoi

Tower of Hanoi

- The **Tower of Hanoi** (also called the **Tower of Brahma** or **Lucas' Tower** and sometimes pluralized as **Towers**) is a mathematical game or puzzle.
- It consists of **three rods** and a **number of disks** of different sizes, which can slide onto any rod.
- The puzzle starts with the disks in a neat stack in **ascending** order of **size** on one rod, the **smallest at the top**, thus making a **conical** shape.

Source: Wikipedia

Tower of Hanoi

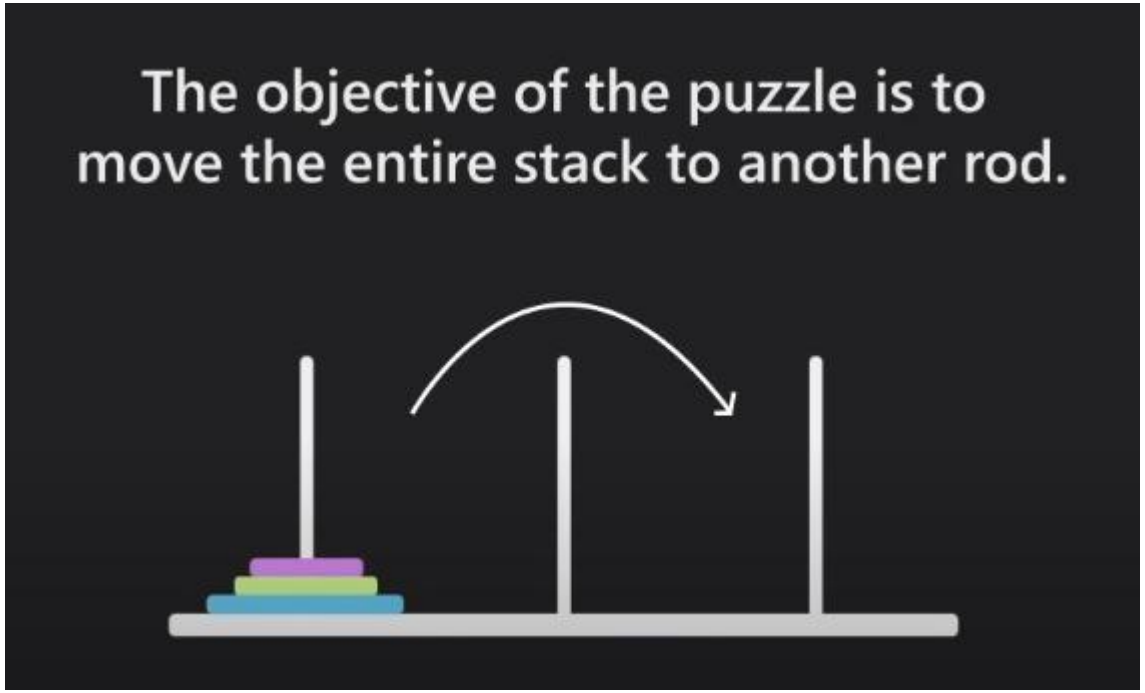


Tower of Hanoi

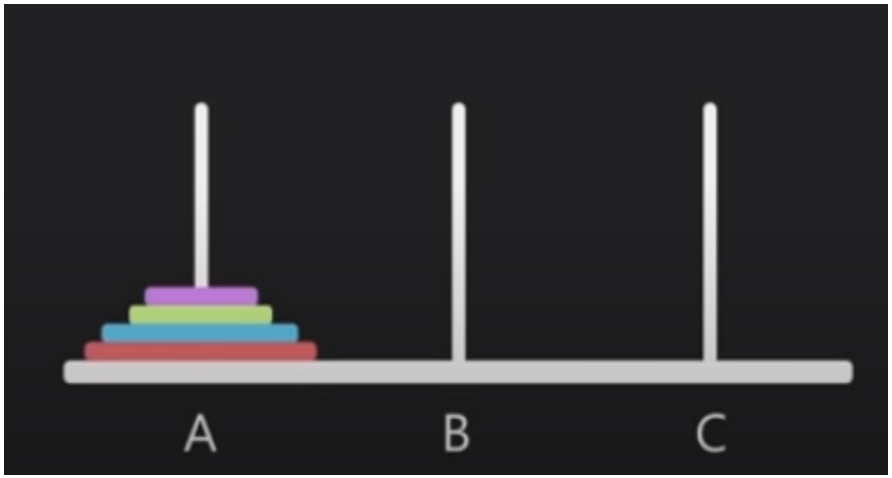
- The **objective** of the puzzle is to move the entire stack to another rod, obeying the following simple **rules**:
  - **Only one disk** can be moved at a time.
  - Each move consists of taking the **upper disk** from one of the stacks and placing it on top of another stack or on an empty rod.
  - **No larger disk may be placed on top of a smaller disk**.

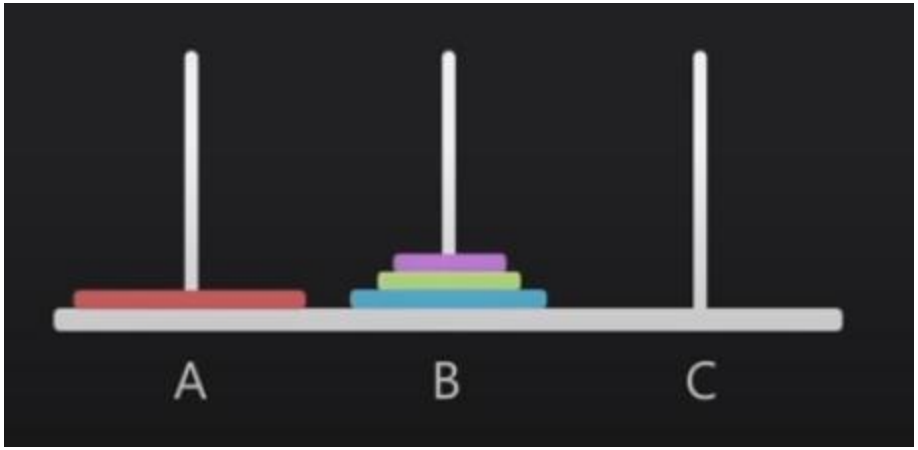
Source: Wikipedia

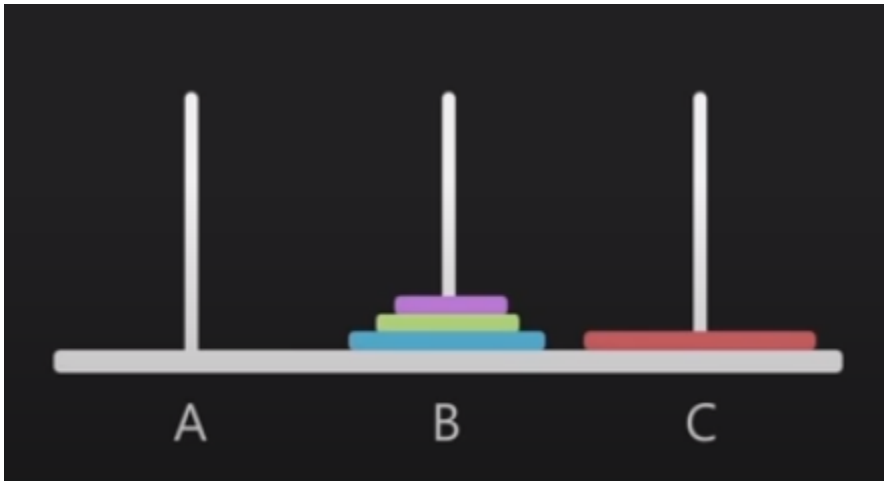
Tower of Hanoi

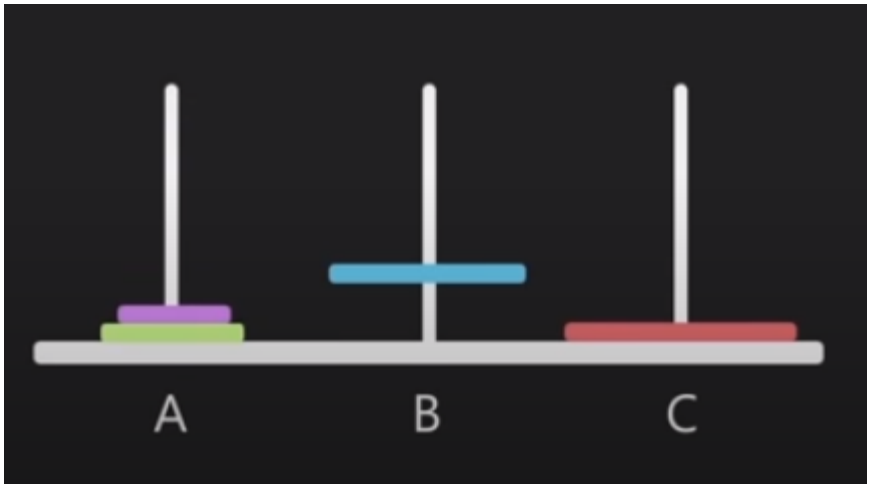


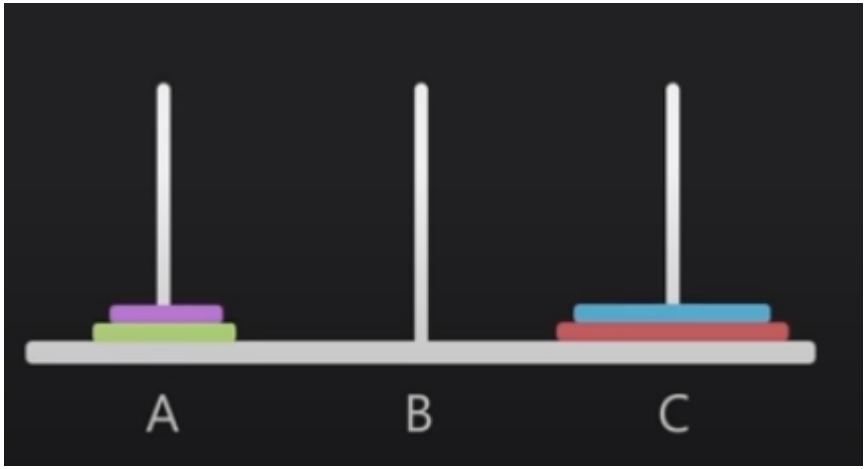
Let's Try with 4 Discs

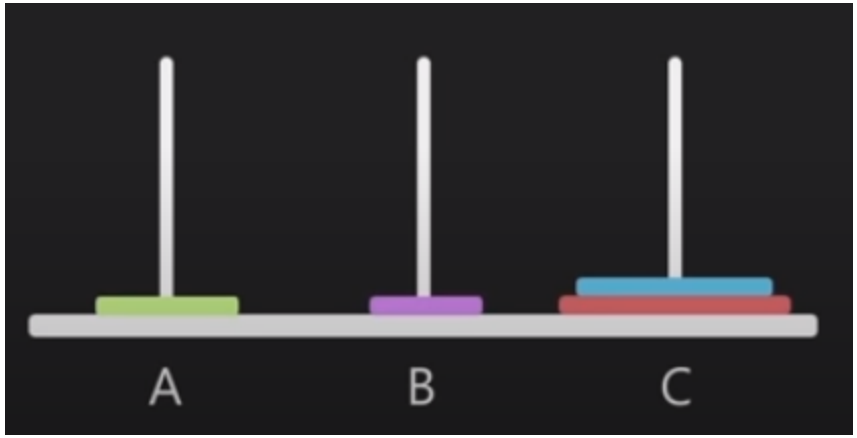


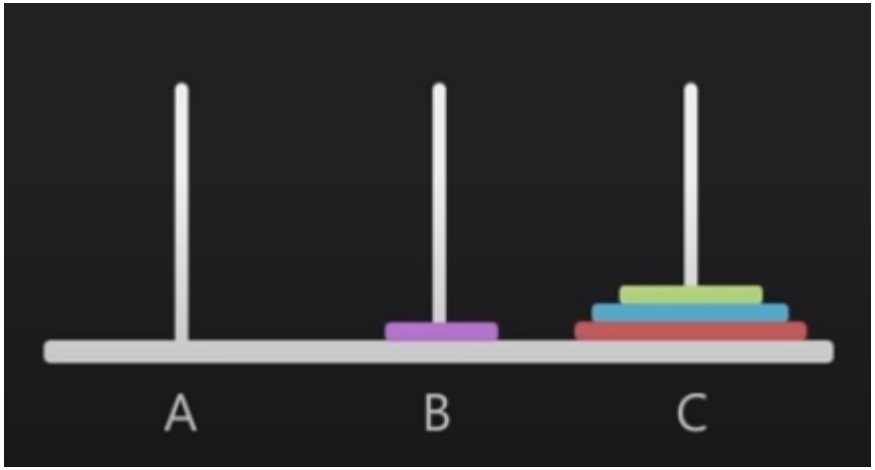

A
B
C


A
B
C


A
B
C


A
B
C


A
B
C


A
B
C

Algorithm

```
ToH(n, S, D, A)
        if (n==1)
                Move n to D
        ToH(n-1, S, A, D)
        Move nth Disc to D
        ToH(n-1, A, D, S)
```