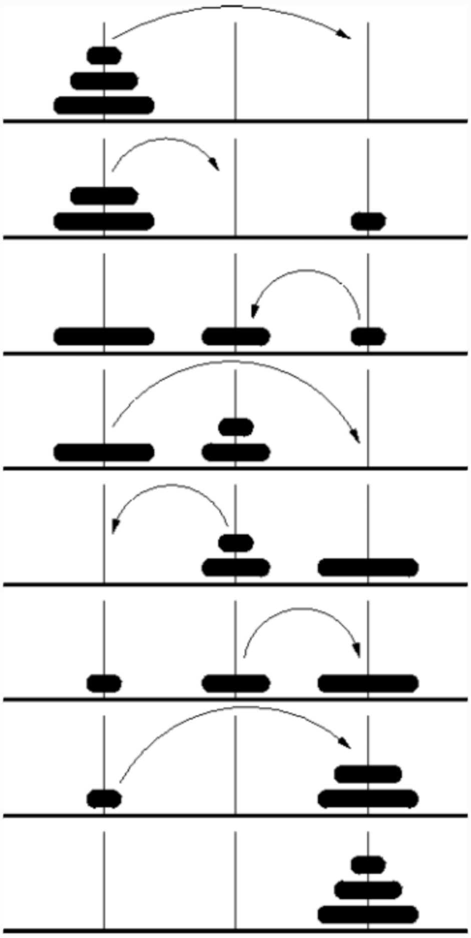
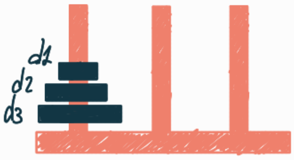
The Towers of Hanoi problem



Rules for Towers of Hanoi. The goal of the puzzle is to move all the disks from the leftmost peg to the rightmost peg, adhering to the following rules: Move only **one disk at a time**. A **larger disk may not be placed on top of a smaller disk**.

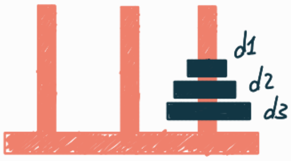
```
(define (domain hanoi-domain)
    (:requirements :strips )
    (:predicates
        (disk ?x) → is the x object a disk ?
        (clear ?x) → nothing is on top of the disk x
        (on ?x ?y) → is x on top of y
        (smaller ?x ?y)
    )
                                   → we can solve this problem with the single move action
    (:action move : move the disk from a location to another location
      :parameters (?d ?from ?to)
      :precondition (and (
          ;input check → is it a disk ?
          (disk ?d) (location ?from)(location ?to)
          (smaller ?d ?to) disk should be smaller of target position
          (on ?d ?from) disk should be on top of the pile
          (clear ?d) nothing is on top of the disk you are moving
          (clear ?to) ending position is also clear (?)
      )
      :effect (and (
            (clear ?from)
            (on ?d ?to)
            (not (on ?d ?from))
            (not (clear ?to))
      )
    )
)
```

```
(define (problem hanoi_problem)
        (:domain hanoi_domain)
        (:objects d1 d2 d3
                  p1 p2 p3
        )
        (:init
            (disk d1)(disk d2)(disk d3)
            (on d1 d2)(on d2 d3)
            (on d1 p1)(on d2 p2)(on d3 p1)
            (clear d1)(clear p2)(clear p3)
```



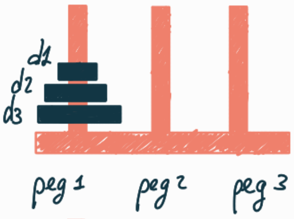
pegs are always bigger than the disks, so it is always possible to move a disk on an empty peg

```
            (smaller d1 d2)(smaller d2 d3)(smaller d1 d3)
            (smaller d1 p1)(smaller d2 p1)(smaller d3 p1)
            (smaller d1 p2)(smaller d2 p2)(smaller d3 p2)
            (smaller d1 p3)(smaller d2 p3)(smaller d3 p3)
        )
        (:goal (and
            (on d1 d2)(on d2 d3)
            (on d3 p3)
        )
)
```



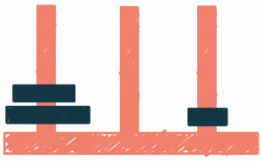
Plan found :

init

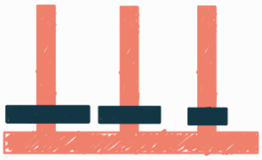


from to

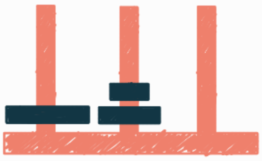
move d1 d2 peg3



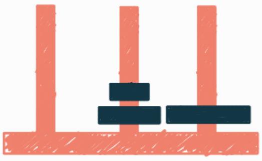
move d2 d3 peg2



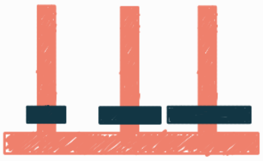
move d1 peg3 d2



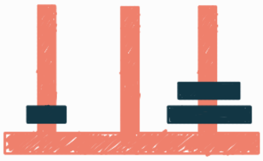
move d3 peg1 peg3



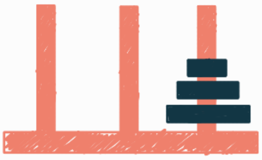
move d1 d2 peg1



move d2 peg2 d3



move d1 peg1 d2



goal