4. 하노이탑(tower of hanoi)

하노이 탑은 **퍼즐**의 일종이다.(위키 백과 참조)

3개의 기둥과 이 기둥에 꽂을 수 있는 크기가 다양한 원판들이 있고,
퍼즐을 시작하기 전에는 한 기둥에 원판들이 작은 것이 위에 있도록 순서대로 쌓여 있다.

게임의 목적은 다음 두 가지 조건을 만족시키면서,
한 기둥에 꽂힌 원판들을 그 순서 그대로 다른 기둥으로 옮겨서 다시 쌓는 것이다.

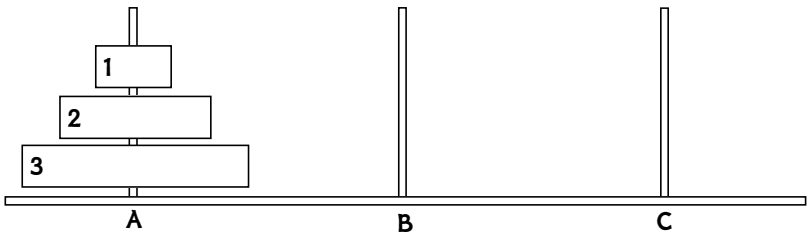
한 번에 하나의 원판만 옮길 수 있다.
큰 원판이 작은 원판 위에 있어서는 안 된다.

하노이탑 문제는 재귀호출을 이용하여 풀 수 있는 가장 유명한 예제 중의 하나이다.
그렇기 때문에 프로그래밍 수업에서 알고리즘 예제로 많이 사용한다.
일반적으로 원판이 n개 일 때, 2^n-1번의 이동으로 원판을 모두 옮길 수 있다.
2^n-1는 **메르센 수**라고 부른다.

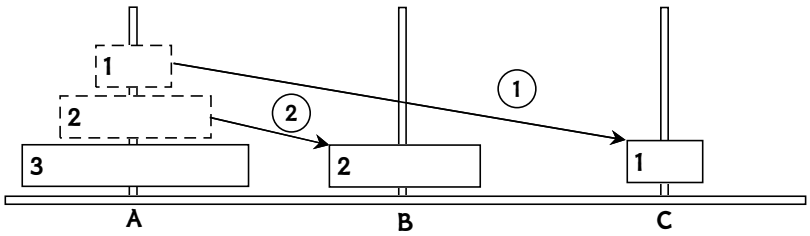
참고로, 64개의 원판을 옮기는 데 $2^n-1 \fallingdotseq 18446744073709551615$번을 움직여야 하고,
한번 옮기는 시간을 1초로 하면, 64개의 원판을 옮기는 데 5849억 4241만 7355년 걸린다.

하노이탑 재귀함수의 시간복잡도는 $O(2^n)$이다. 즉, **지수시간**이다.

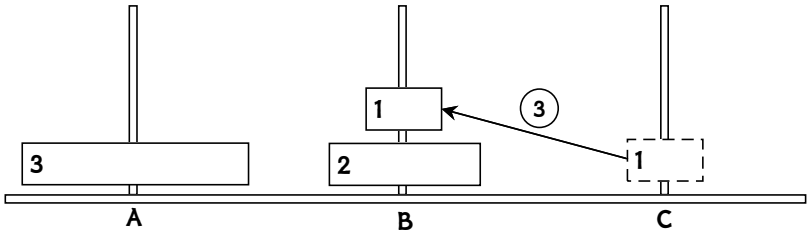
◈ 하노이 탑에서 원판이 3개인 경우 - 7번 이동이 필요

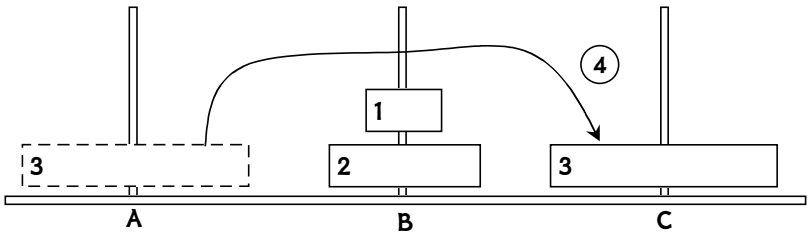


초기 상태

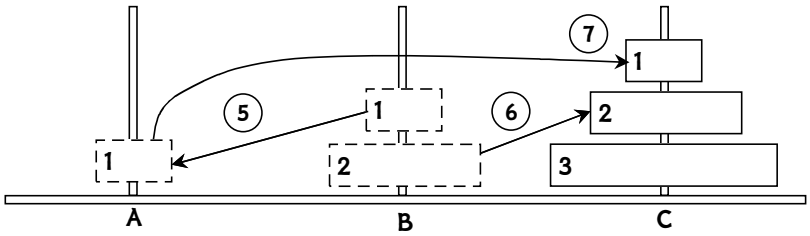


→ 원판 1을 기둥 B로, 원판 2를 기둥 C로 이동해도 된다.



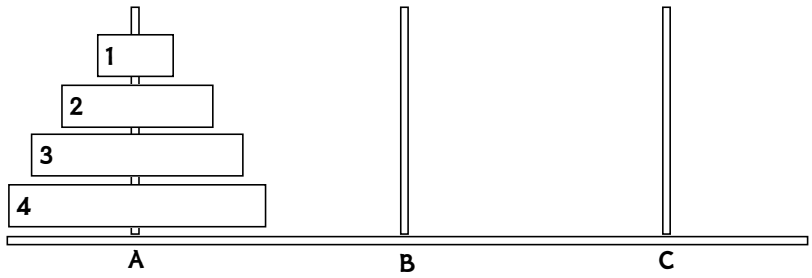


가장 큰 원판 이동

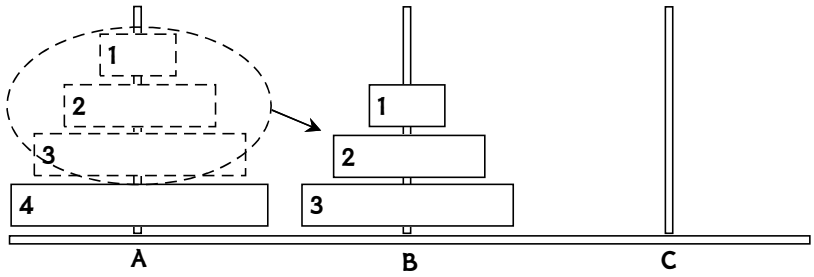


이동 완료

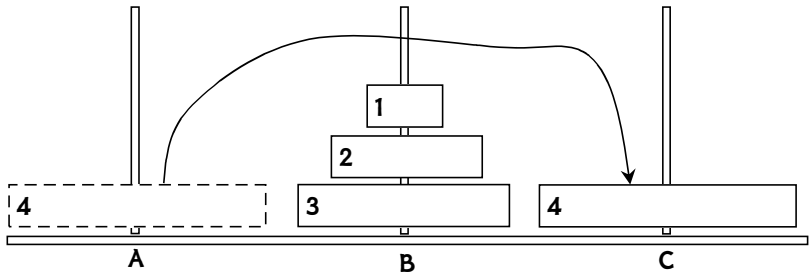
◈ 하노이 탑에서 원판이 4개인 경우 – 15번 이동이 필요



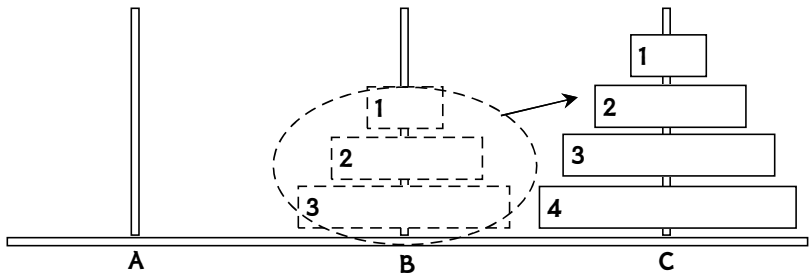
초기 상태



7번 이동이 필요



가장 큰 원판 이동



7번 이동이 필요

• 가장 큰 원판이 이동되는 시점이 전체 이동횟수에서 항상 중간이 된다.(8번째 이동)
• 이유는 가장 큰 원판 이동을 제외한 3개의 원판 이동횟수가 같으므로
• 이런 원리를 이해하면, 주어진 문제에서 답을 쉽게 찾을 수 있다.(같은 패턴 반복)

```
// 하노이탑 알고리즘을 파이썬으로 구현
def hanoi(n, A, B, C):
    if n==1:                    # 원반이 1개이면 그냥 목적지로 옮기면 됨
        print('원반 {0} : 기둥 {1}에서 {2}로 이동'.format(n, A, C))
    else:
        hanoi(n-1, A, C, B)    # n-1개의 원반을 A에서 B로 이동
        print('원반 {0} : 기둥 {1}에서 {2}로 이동'.format(n, A, C))    # 가장 큰 원반 이동
        hanoi(n-1, B, A, C)    # n-1개의 원반을 B에서 C로 이동
hanoi(3, 'A', 'B', 'C')            # 기둥 "A"에 있는 원반 3개를 기둥 "C"로 이동
```

↓

↓출력

↓

```
원반 1 : 기둥 A에서 C로 이동
원반 2 : 기둥 A에서 B로 이동
원반 1 : 기둥 C에서 B로 이동
원반 3 : 기둥 A에서 C로 이동
원반 1 : 기둥 B에서 A로 이동
원반 2 : 기둥 B에서 C로 이동
원반 1 : 기둥 A에서 C로 이동
```

```
// 하노이탑 알고리즘에서 핵심 부분
hanoi(n-1, A, C, B)        # n-1개의 원반을 A에서 B로 이동
print('원반 {0} : 기둥 {1}에서 {2}로 이동'.format(n, A, C))      # 가장 큰 원반 이동
hanoi(n-1, B, A, C)        # n-1개의 원반을 B에서 C로 이동
```

↓

```
hanoi(n-1, A, C, B)        # A, C, B 순서인데
print('원반 {0} : 기둥 {1}에서 {2}로 이동'.format(n, A, C))      # 가장 큰 원반 이동
hanoi(n-1, B, A, C)        # B, A, C 순서임
```