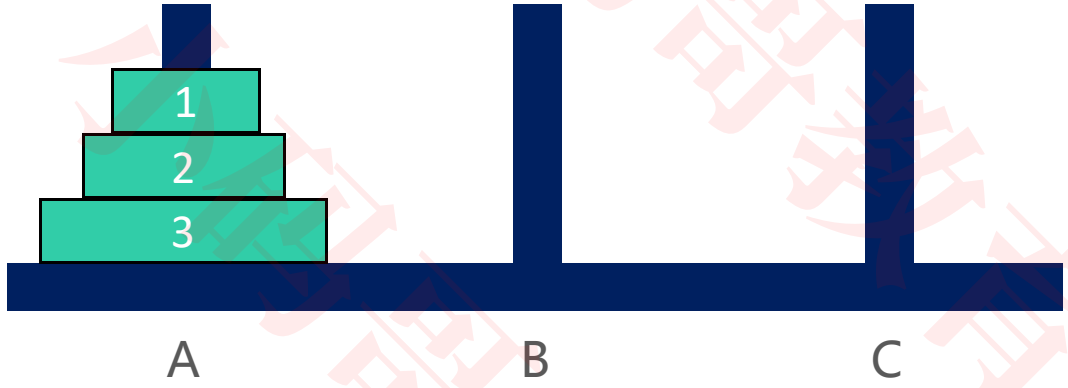
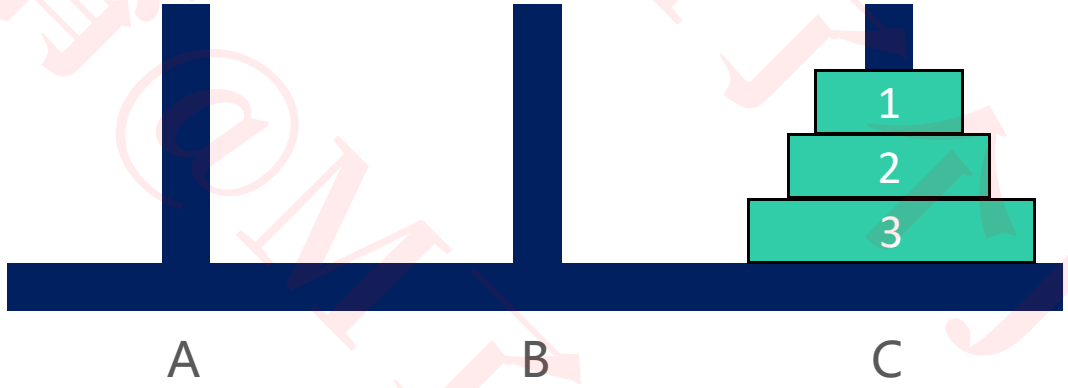
练习3 – 汉诺塔（Hanoi）

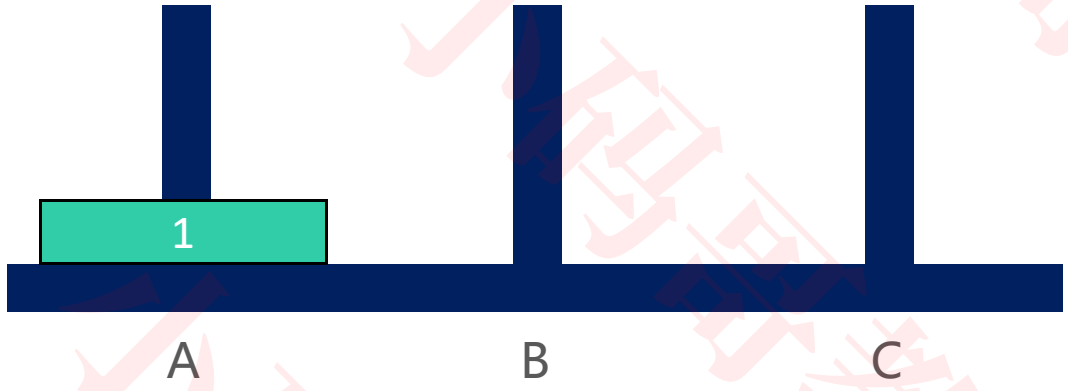
- 编程实现把 A 的 n 个盘子移动到 C（盘子编号是 [1, n] ）
  - 每次只能移动1个盘子
  - 大盘子只能放在小盘子下面

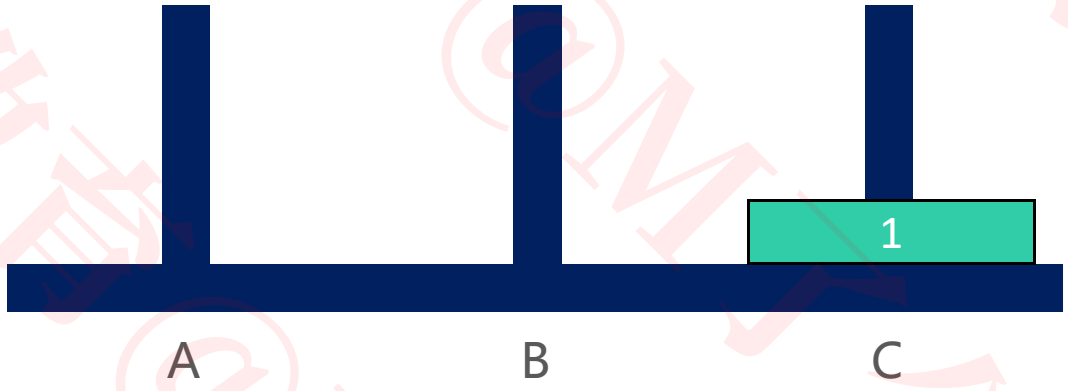




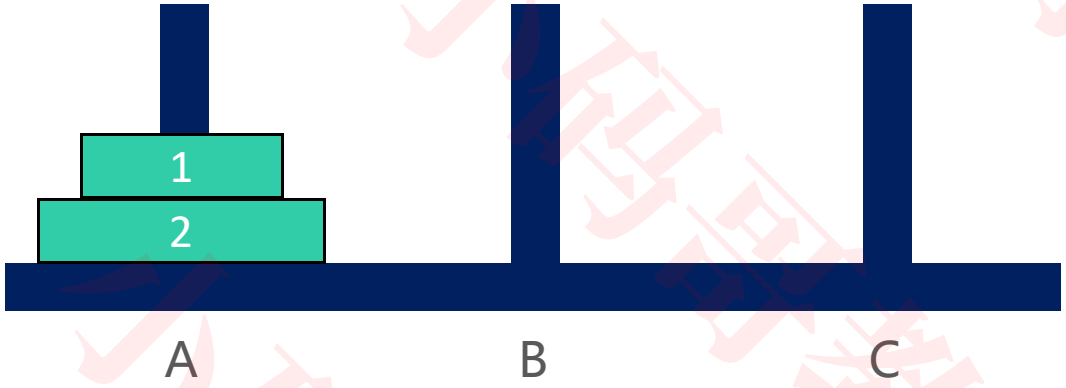
小码哥教育 SEEMYGO

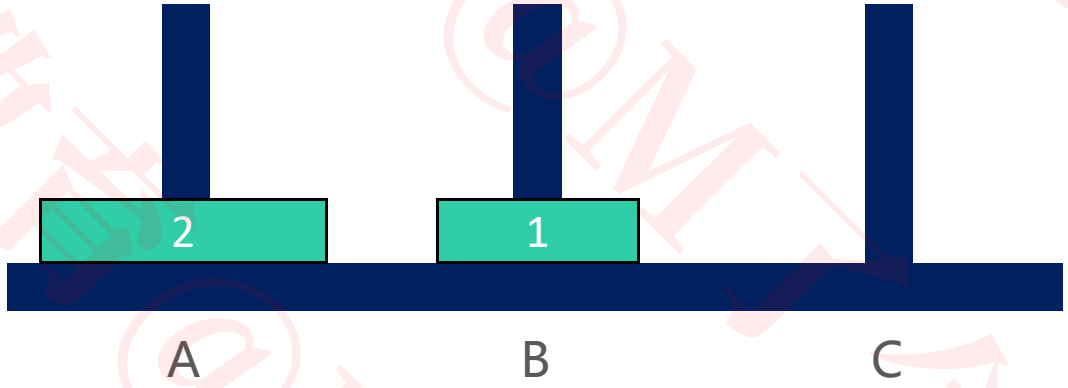
1个盘子

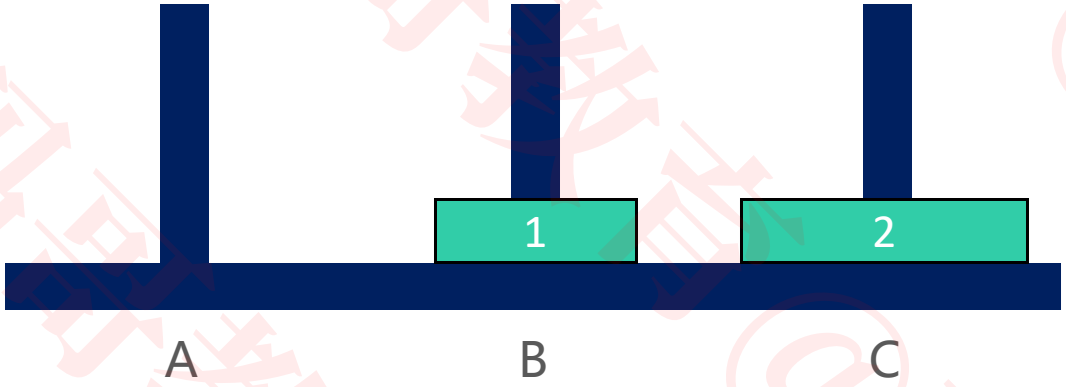


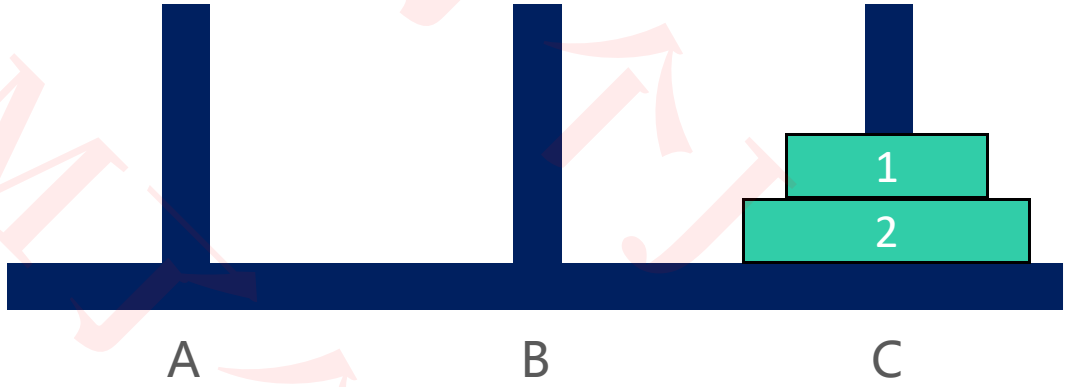


2个盘子



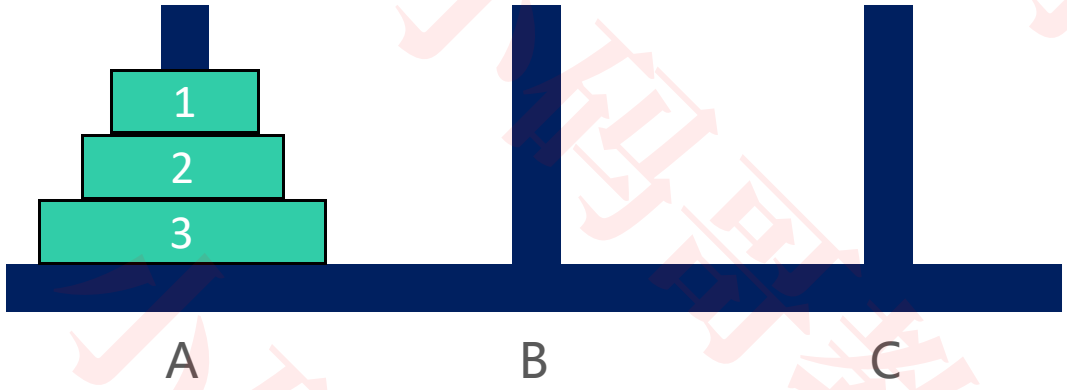


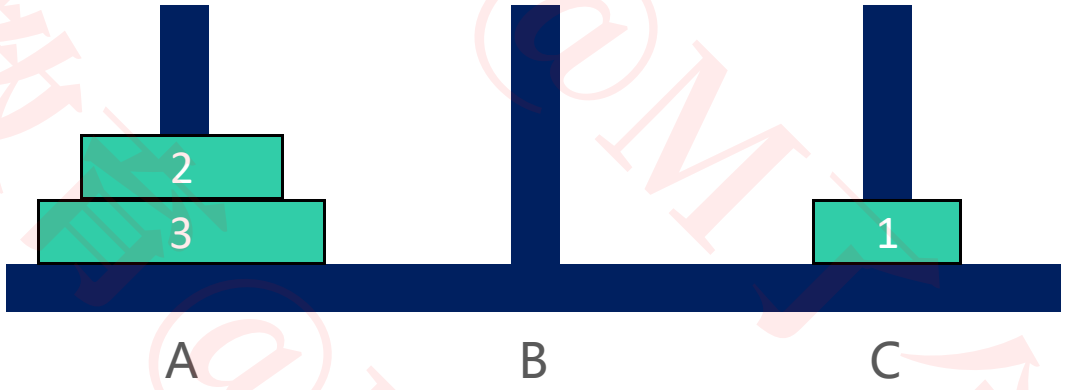


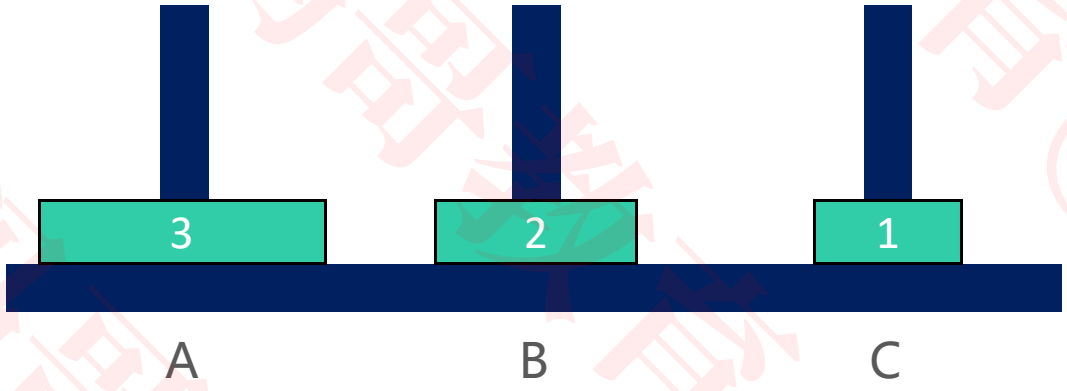


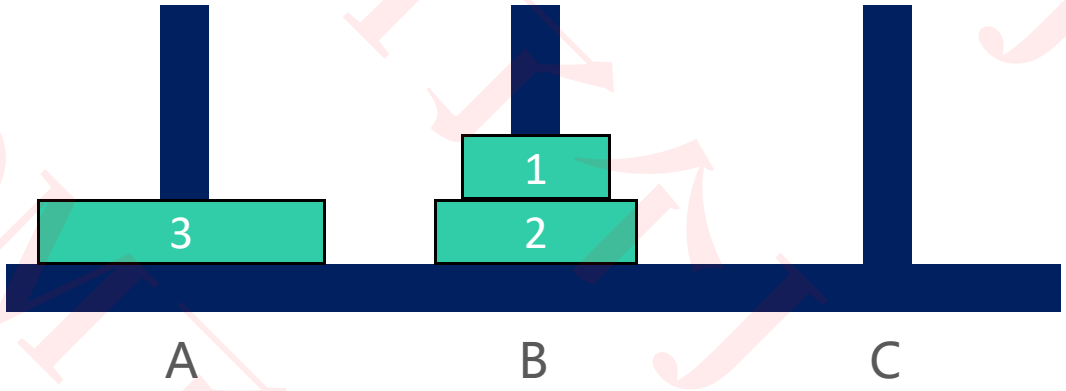
小码哥教育 SEEMYGO

3个盘子



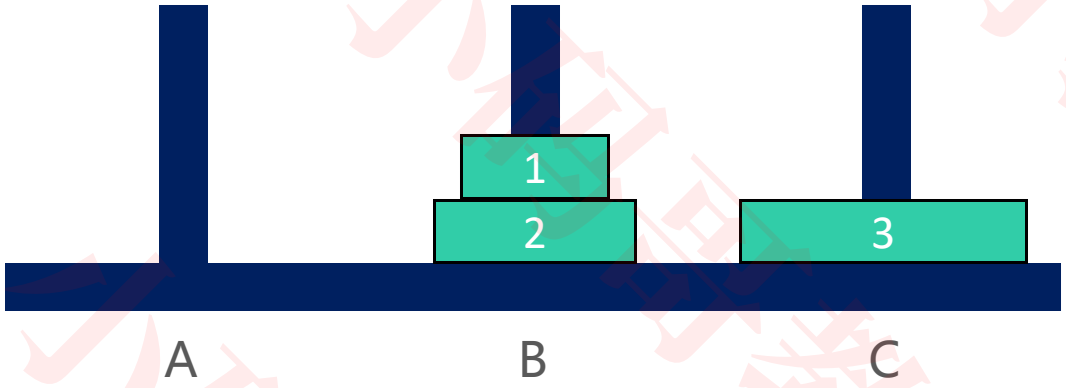


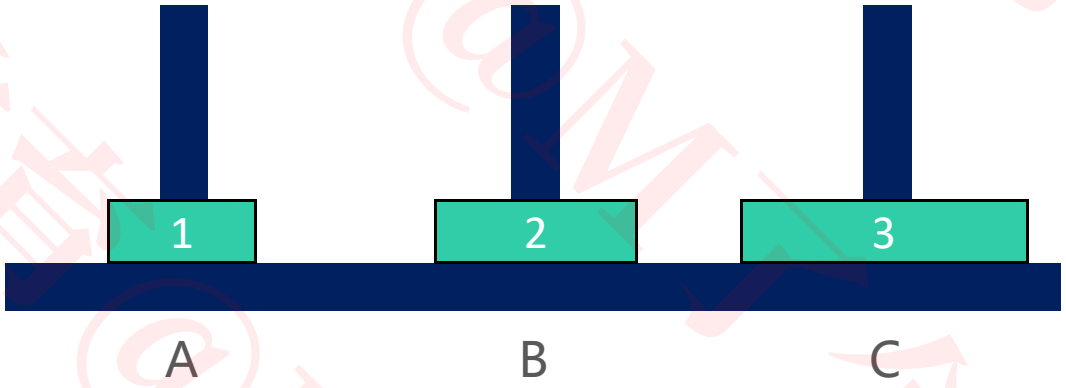


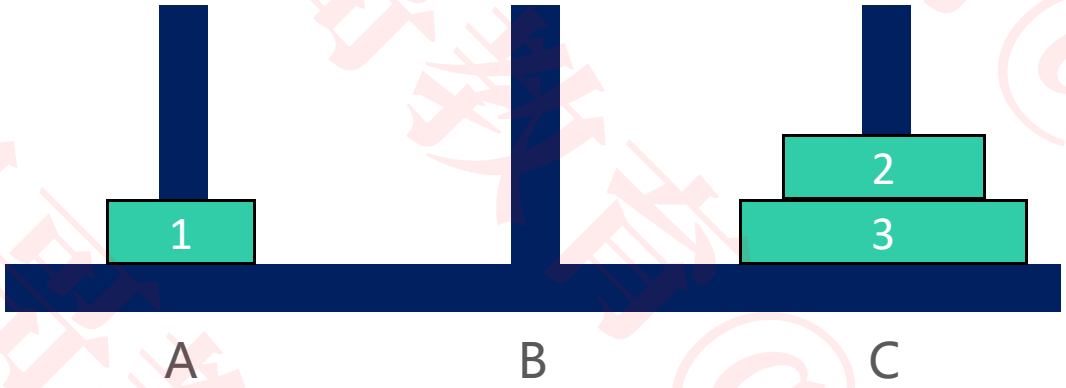


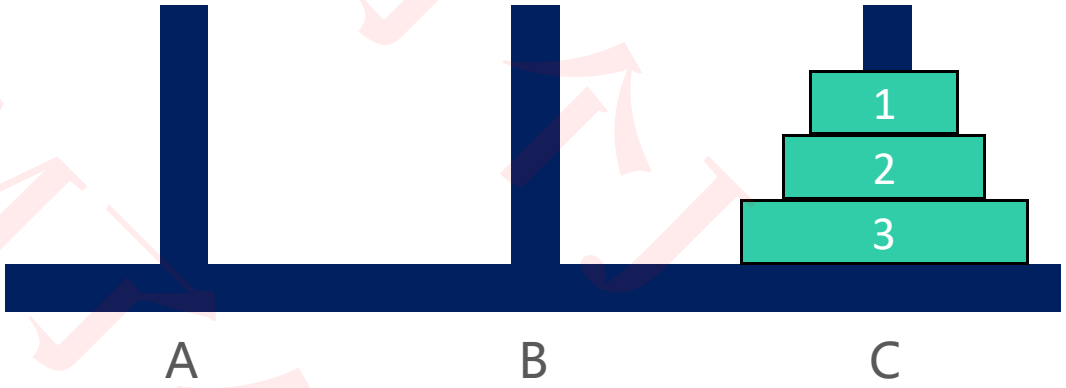
小码哥教育 SEEMYGO

3个盘子









汉诺塔 – 思路

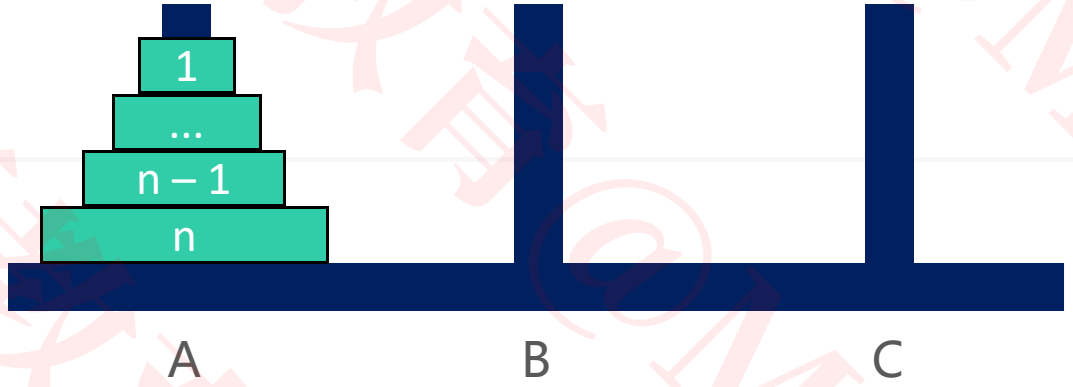
- 其实分 2 种情况讨论即可
- 当 n == 1时，直接将盘子从 A 移动到 C
- 当 n > 1时，可以拆分成3大步骤

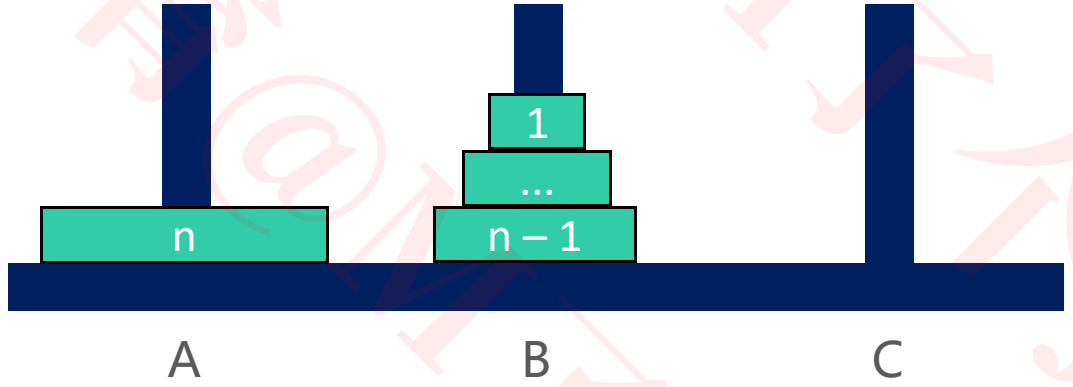
① 将 n – 1 个盘子从 A 移动到 B

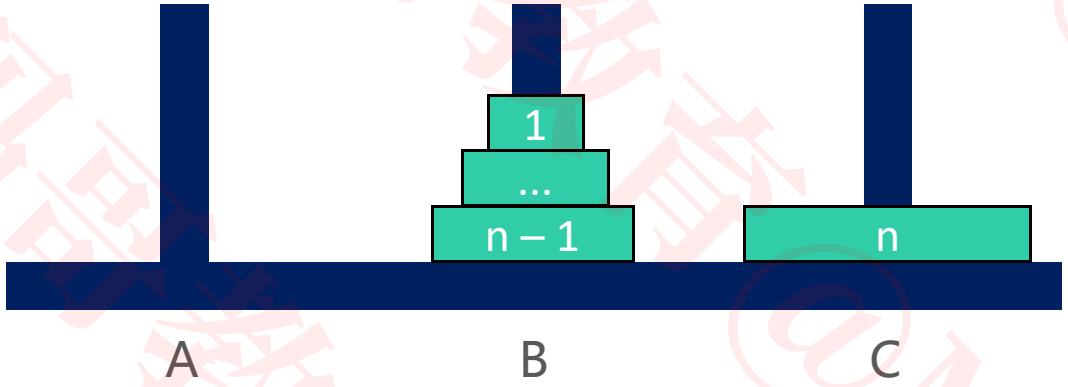
② 将编号为 n 的盘子从 A 移动到 C

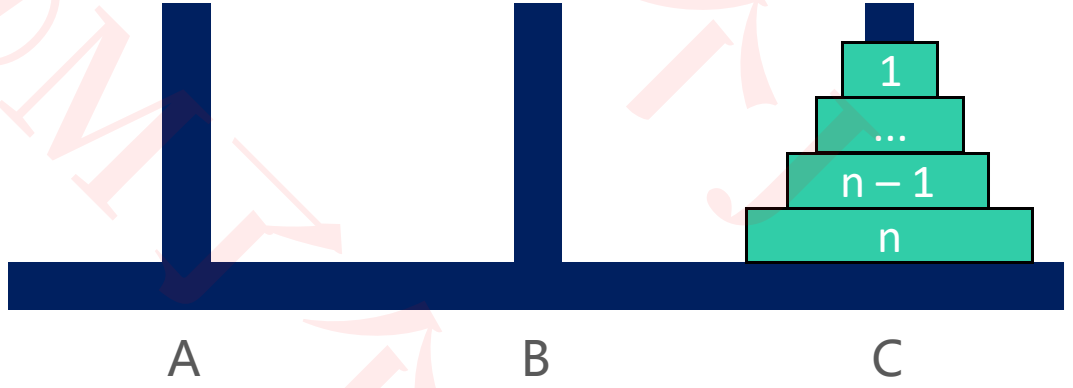
③ 将 n – 1 个盘子从 B 移动到 C

✓ 步骤 ① ③ 明显是个递归调用









汉诺塔 – 实现

```java
/**
 * 将第 i 号盘子从 from 移动到 to
 */
void move(int i, String from, String to) {
    System.out.println(i + "号盘子: " + from + "->" + to);
}
```

```java
/**
 * 将 n 个盘子从 p1 移动到 p3
 */
void hanoi(int n, String p1, String p2, String p3) {
    if (n <= 1) {
        move(n, p1, p3);
        return;
    }
    hanoi(n - 1, p1, p3, p2);
    move(n, p1, p3);
    hanoi(n - 1, p2, p1, p3);
}
```

- $T(n) = 2 * T(n-1) + O(1)$
  - 因此时间复杂度是：$O(2^n)$
- 空间复杂度：$O(n)$