Towers of Hanoi

Goal: Move all disks from A to C

Rules:

- Only one disk may move at a time
- no larger disk may be stacked on a smaller disk

Insight: First move top-3 disks from A to B (recursively) and then move bottom disk A → C

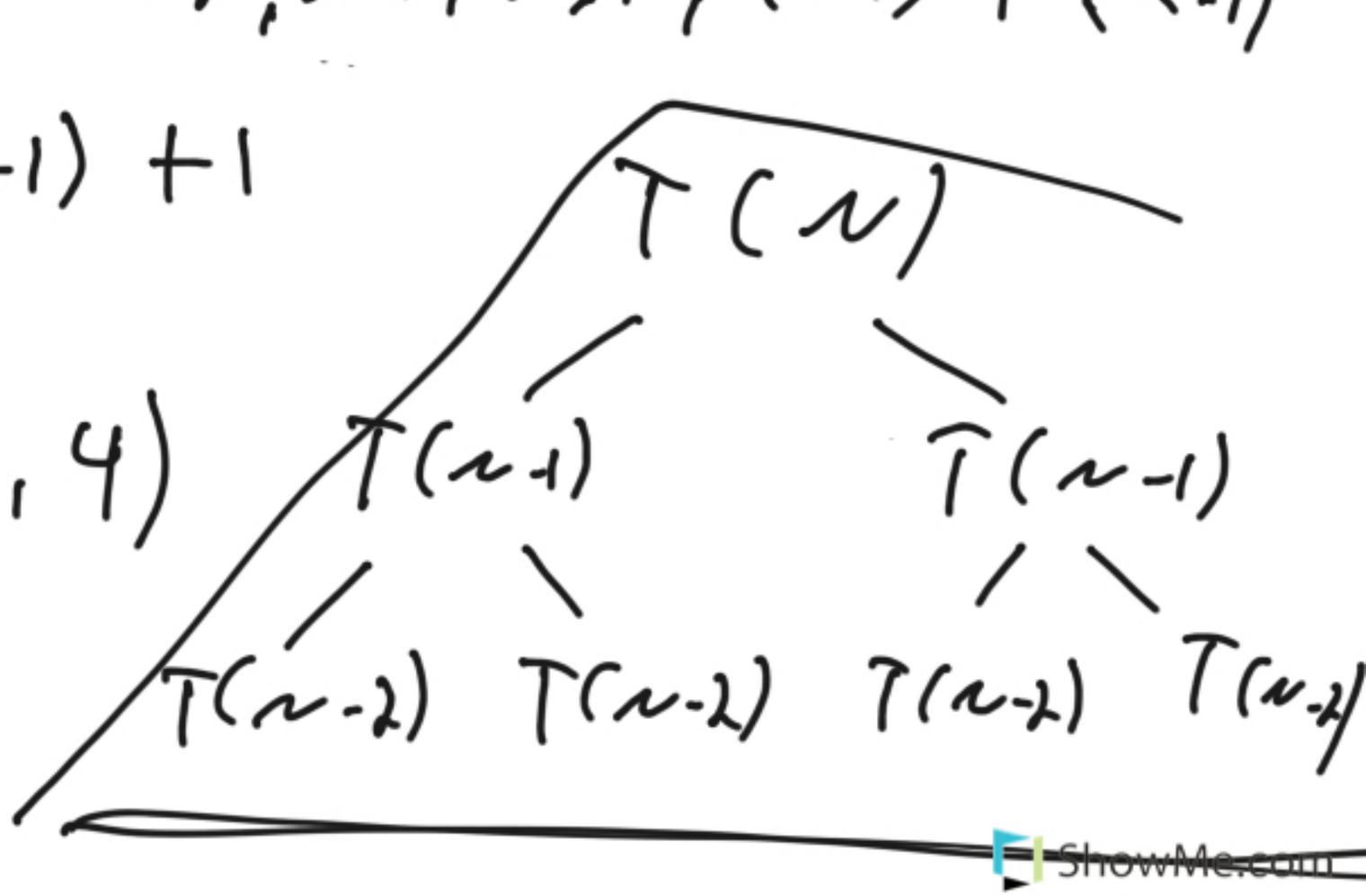
Hanoi (src, buffer, dest, k):

- step 1: Hanoi (src, dest, buffer, k-1) $T(k-1)$
- step 2: Move disk from src to dest
- step 3: Hanoi (buffer, src, dest, k-1) $T(k-1)$

$T(n) = 2 \cdot T(n-1) + 1$

Hanoi (A, B, C, 4)

$O(2^n)$

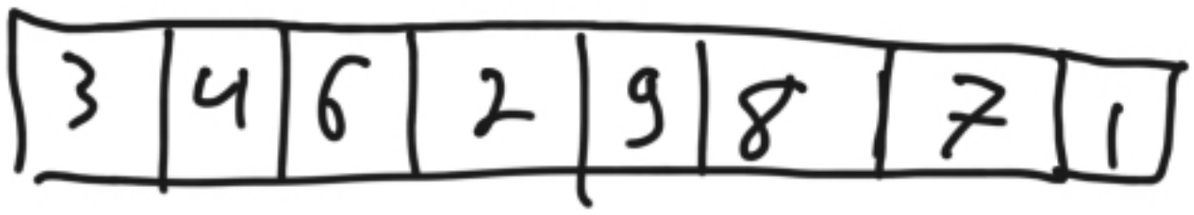


Merge Sort

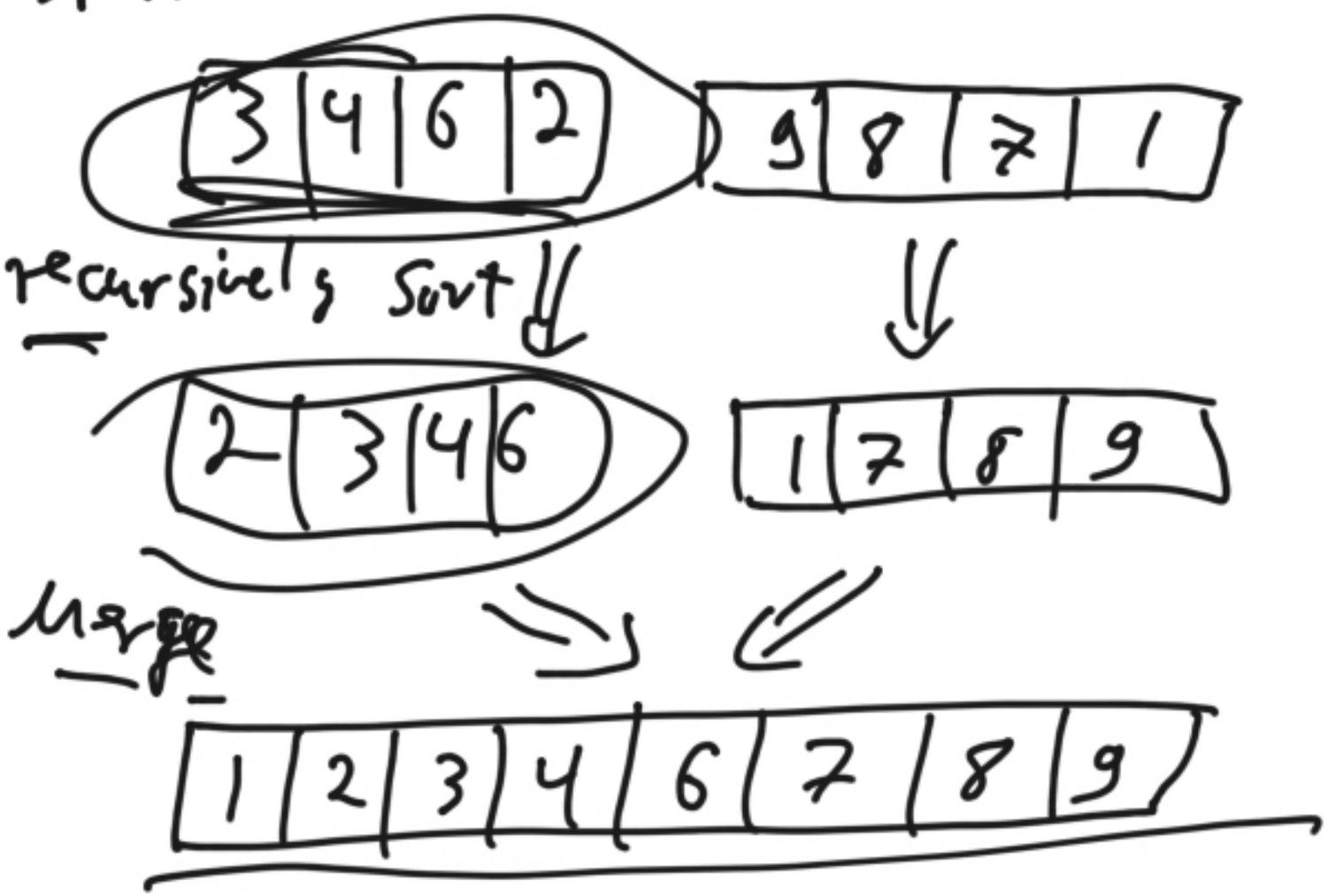
$O(n \log n)$

Stable ✓

requires $O(n)$ additional space

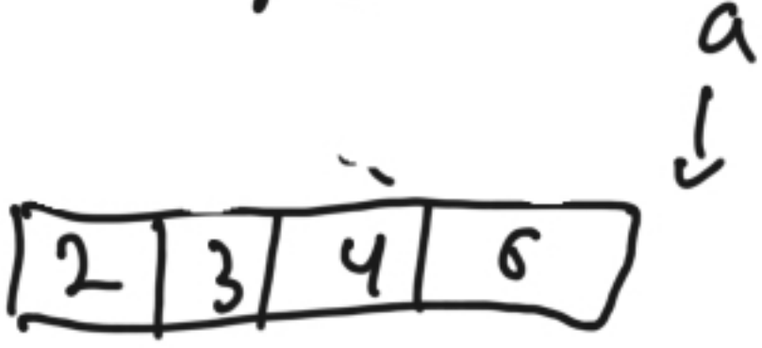


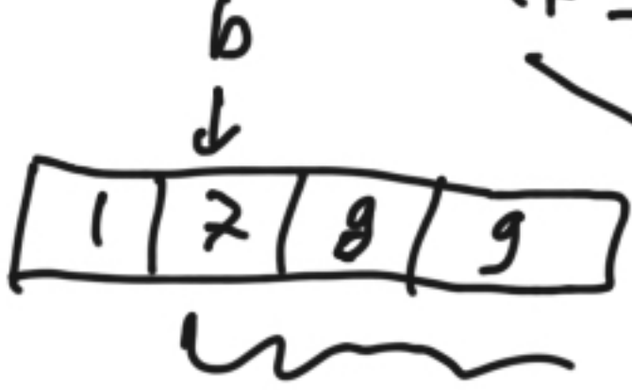
split

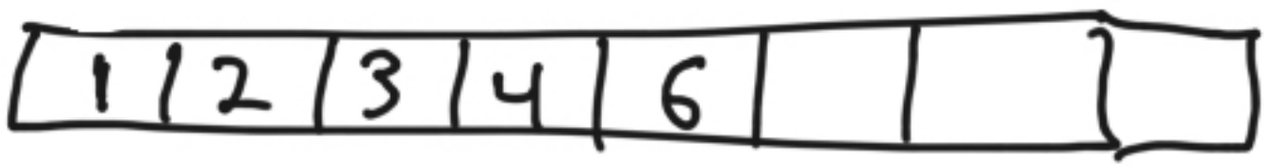


O(n)

Merge








```
while a < len(left) and b < len(right):
    if left[a] <= right[b]:
        result.append(left[a])
        a += 1
    else:
        result.append(right[b])
        b += 1
```

```
while a < len(left):
    result.append(left[a])
    a += 1
while b < len(right):
    result.append(right[b])
    b += 1
```

$$T(n) = 2T\left(\frac{n}{2}\right) + n$$

$$= 2 \cdot \left(2 \cdot T\left(\frac{n}{4}\right) + \frac{n}{2}\right) + n$$

$$= 4\,T\left(\frac{n}{4}\right) + n + n$$

$$= 2^k T\left(\frac{n}{2^k}\right) + k \cdot n \quad \left| \begin{array}{l} \text{assume} \\ k = \log_2 n \end{array} \right.$$

$$= 2^{\log_2 n} T\left(\frac{n}{2^{\log_2 n}}\right) + \log_2 n \cdot n$$

$$= n\,T(1) + \log_2 n \cdot n$$

$$= n + n \log_2 n = O(n \log n)$$