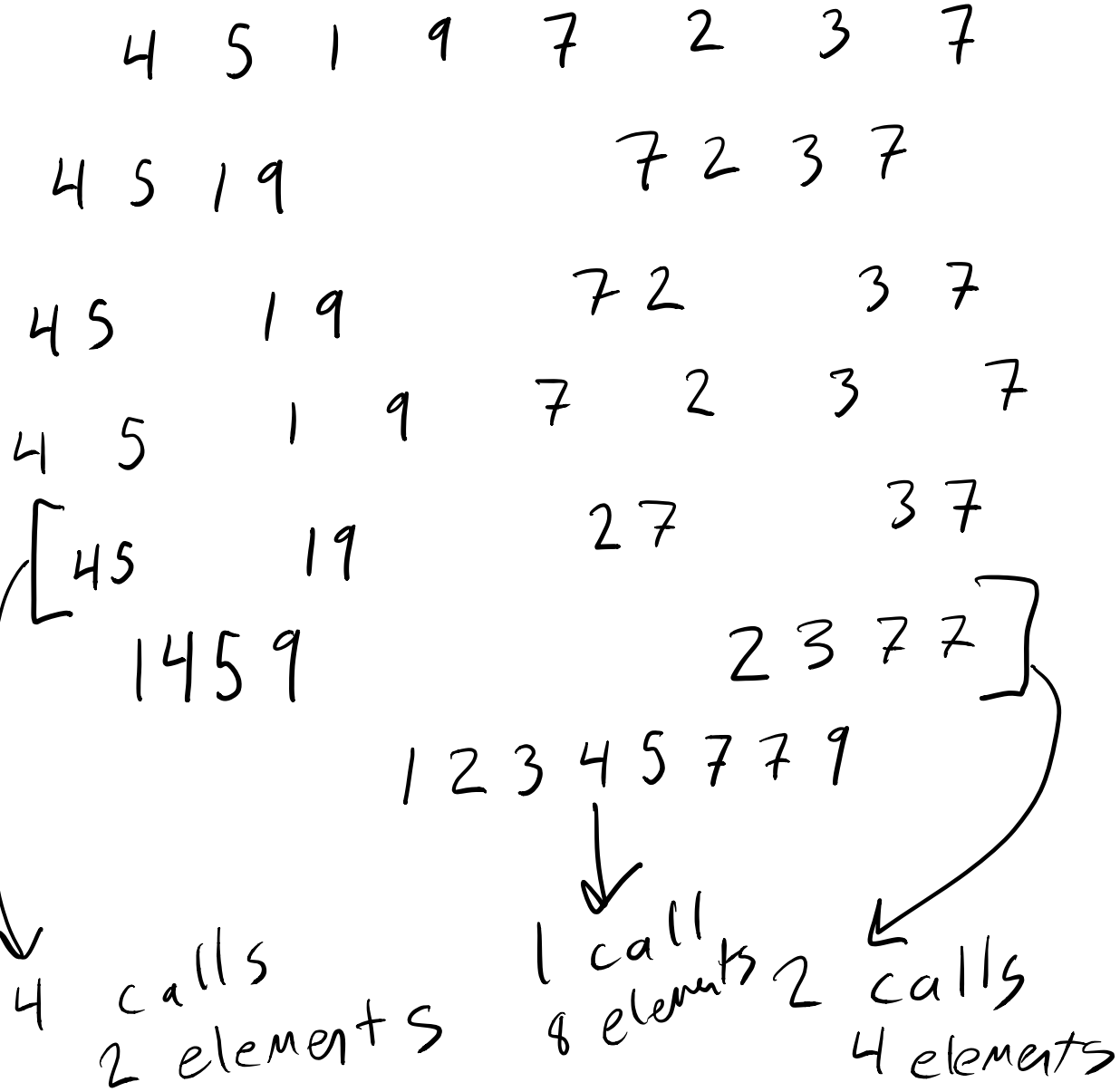
Merge Sort

Split array in half

Merge sort each half

Merge the halves → linear O(n)



example 3 recursions deep

$\lceil \log_2 n \rceil$

$(c \cdot n)(\log_2 n)$

$O(n \log n)$

vs

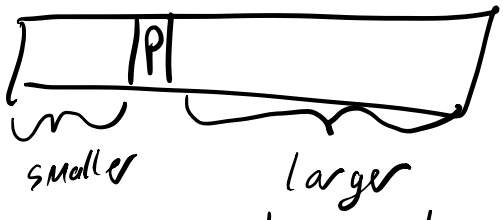
$O(n^2)$

Quick Sort

pick some pivot element

two sub-arrays

- everything less than pivot
- everything larger than pivot

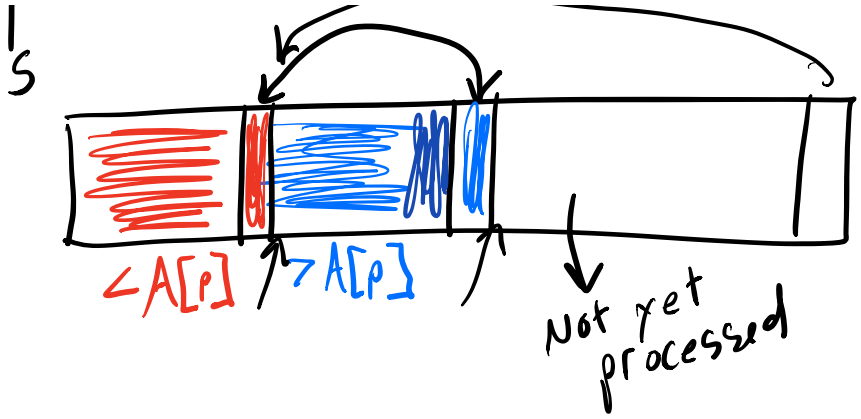


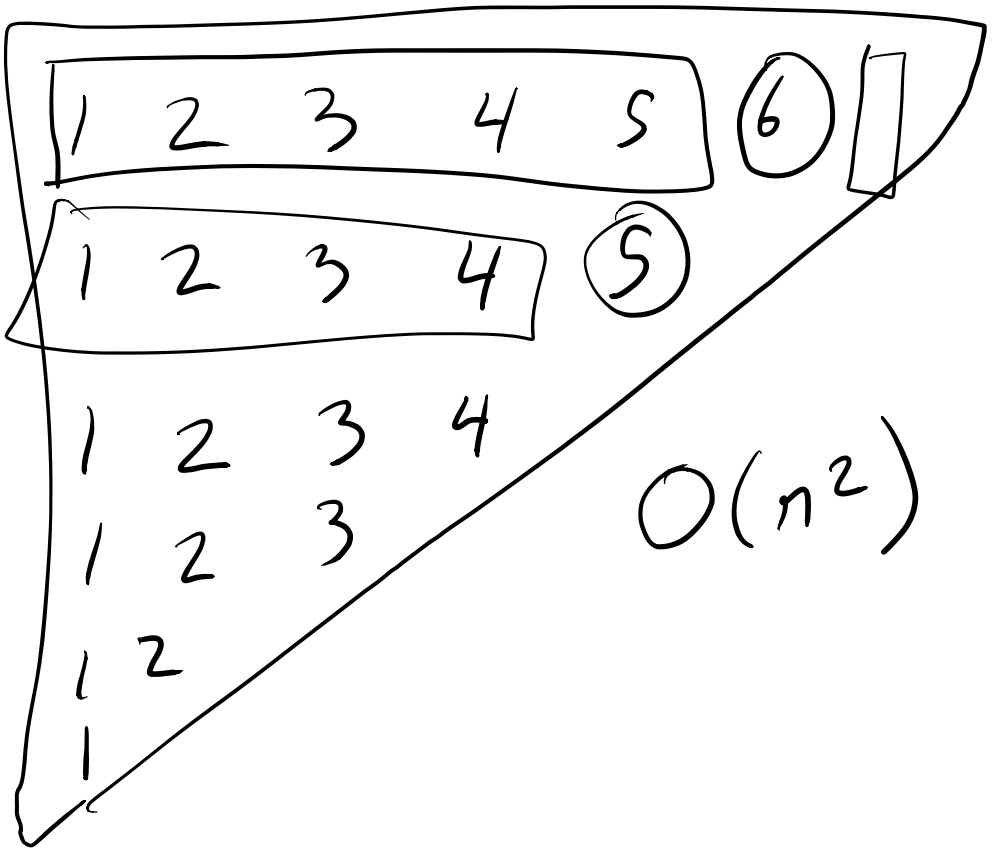
quick sort sub-arrays

```
Function QuickSort(A, start, end)
    if start == end:
        return                    // base case

    p = partition(A, start, end)
    QuickSort(A, start, p-1)
    QuickSort(A, p+1, end)
end Function
```

```
Function Partition(A, start, end):
    s = ~~0~~ start
    l = ~~0~~ start
    pivot = A[end]
    while l < end:
        if A[l] < pivot:
            swap A[l], A[s]
            s++;
        end if
        l++;
    end while
    swap A[s], A[end]
    retur s
end Function
```





Randomize!

Gambler's sort:

flip a coin

if heads: magically sort in $O(1)$ time

if tails: flip again

$$\frac{1}{2} \cdot 1 + \frac{1}{4} \cdot 2 + \frac{1}{8} \cdot 3 + \cdots$$

$$\sum_{i=1}^{\infty} \left(\frac{1}{2}\right)^i \cdot i = 2$$