Merge Problem

Input: $a_1, \ldots, a_n$ $b_1, \ldots, b_m$

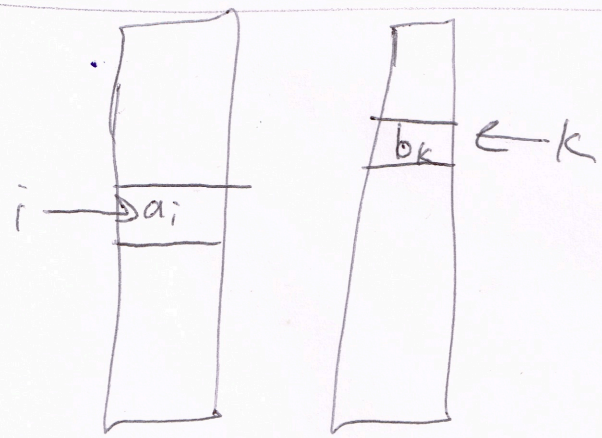
Sorted

Output: m+n numbers in sorted order

Ex in: {1, 3, 5, 7} {2, 4, 6, 8}

out {1, 2, 3, 4, 5, 6, 7, 8}

Goal: O(n) time algo (O(m+n))



Initialize $i \leftarrow 1$

$k \leftarrow 1$

loops

```
if (a_i < b_k)
    add a_i to the output
        array
    i++;
else
    add b_k to the output
        array
    k++;
if (i > n)
    Copy the rest of b
    to the output
    return;
if (k > m)
    Copy the rest of a
    to the output
    return
```

MergeSort(a, n)

$O(1)$ — if $n=1$ return a_1
if $n=2$ return $(\min(a_1, a_2), \max(a_1, a_2))$

else $(n>2)$

$O(n)$ — $a_L \leftarrow a_1, \ldots, a_{\lfloor \frac{n}{2} \rfloor}$
$a_R \leftarrow a_{\lfloor \frac{n}{2} \rfloor + 1}, \ldots, a_n$

$O(n)$ — return Merge((MergeSort(a_L, $\lfloor \frac{n}{2} \rfloor$), MergeSort($a_R$, $n - \lfloor \frac{n}{2} \rfloor$)))

$$O(1) + O(n) + O(n) = O(n)$$

Correctness of Merge Sort

- for recursive algorithms, correctness "typically" by induction on n.

Base case for induction is the base of the recursion.

Base case: $n=1$: sorted: one element is always sorted.
$n=2$: returns (min, max) which must be sorted.

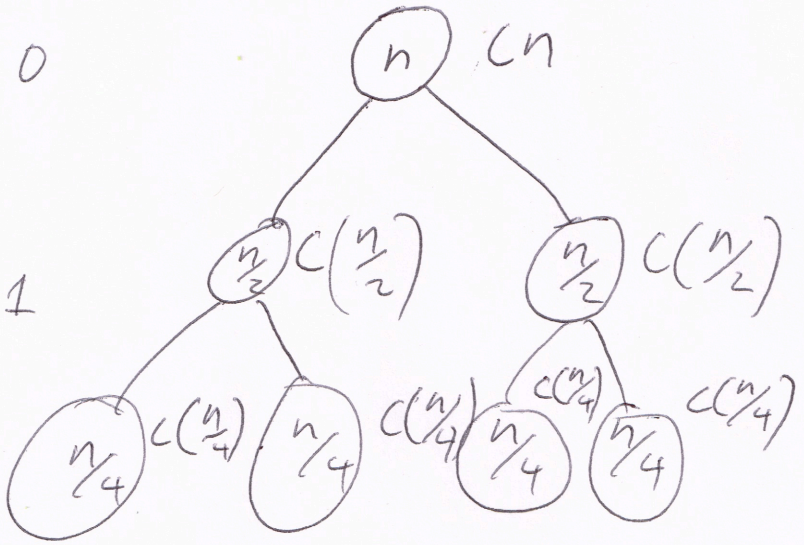
~~recursion~~: induction step: assume true for ~~k~~ all i ~~s.t. 1 ≤ i ≤ k~~ s.t. $1 \leq i \leq k$
prove true for $k+1$.

True since Merge( ) is correct.
this is a shortcut. Merge should be proven correct.

Runtime of mergesort

$T(n) \leq c \quad n \leq 2$ (for base case, runtime is $O(1)$)

$T(n) \leq cn + 2T(n/2)$



Level 0 — cn

Level 1 — cn

Level 2 — cn

Level i — $c(n/2^i)$ each, 2^i of them — $2^i c(n/2^i) = cn$

Level $\log n$ — ① ① ① · · · ①ish

$n = 1$ Base case

$\frac{n}{2^i} = 1 \Rightarrow n = 2^i \Rightarrow \log n = i$

$\log n$ levels with cn time each,

$cn \log n$ time $\equiv O(n \log n)$