Mergesort

$T(n)$ = # steps required on input of size n
↳ comparisons between items

$T(n) \le T(\frac{n}{2}) + T(\frac{n}{2}) \underbrace{+ cn}$
$T(1) = c_1$
$T(0) = c_0$

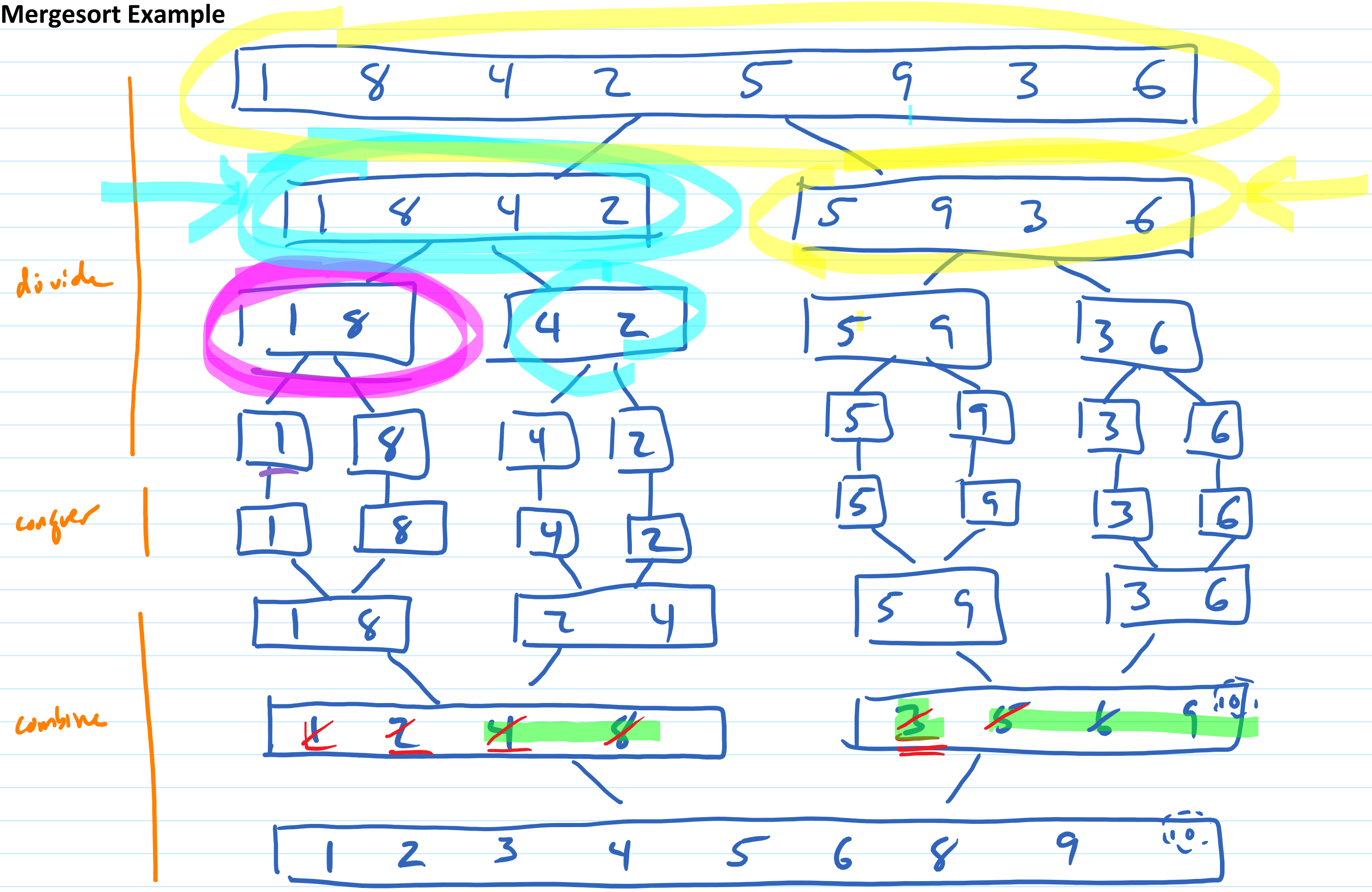
```
MERGE-SORT ( A )
    if len(A) < 2
        return a copy of A
    else
        L ← 1st half of A           ] divide list into two parts
        R ← rest of A
        L ← MERGESORT(L)            ] conquer the subproblems
        R ← MERGESORT(R)
f(n)    return MERGE(L,R)           ] combine results
            precondition: L, R are sorted

MERGE(L,R)
    A ← empty list
    i ← 0
    j ← 0
    while i < len(L) and j < len(R)     as long as elements left in both lists
        if L[i] ≤ R[j]                  compare what's 1st among remaining items
            append L[i] to A
            i ← i+1
        else
            append R[j] to A
            j ← j+1

    append L[i...len(L)-1] to A     } concatenate remaining items on
    append R[j...len(R)-1] to A     } non-empty list to result

    return A
```

$T(16) \le T(8) + T(8) + c \cdot 16$
$\le T(4) + T(4) + c \cdot 8$
$+ T(4) + T(4) + c \cdot 8$
$\le \vdots$

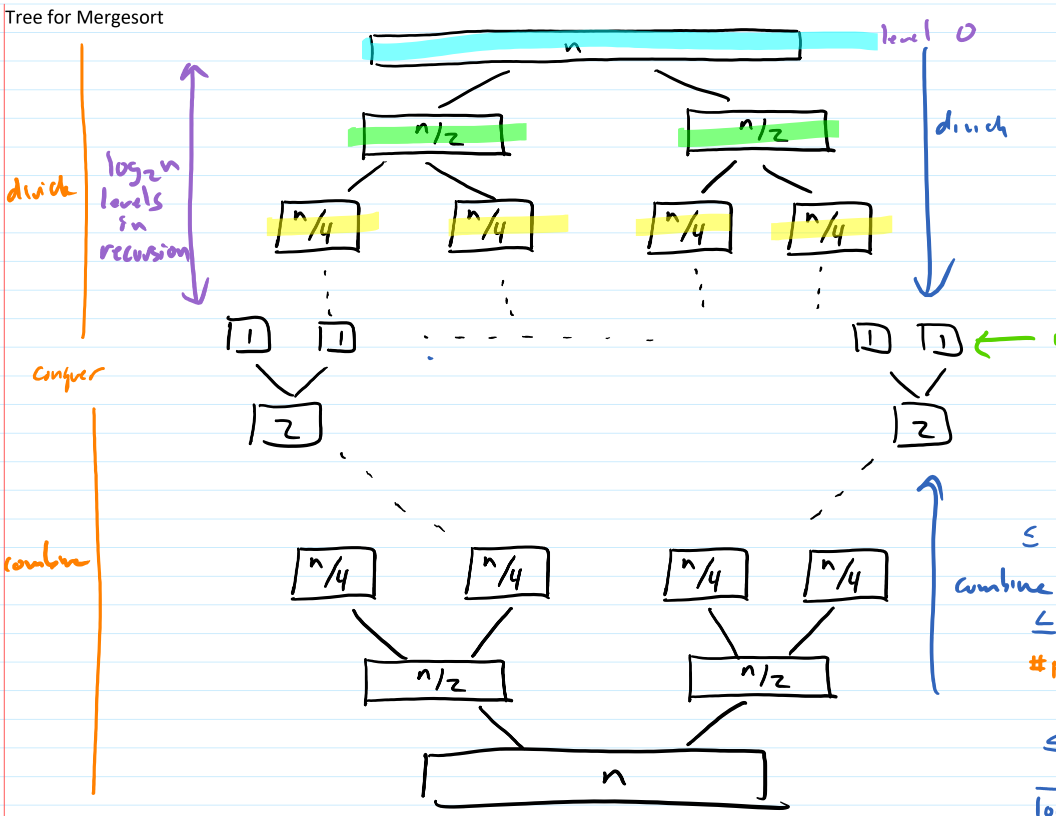
Divide-and-Conquer

divide
conquer
combine

Mergesort Example



Tree for Mergesort



← n subproblems c_1 work each (base cases)

so $c_1 \cdot n$ total (dominated by divide / combine)

$\leq \underline{4}(\underline{c \cdot \frac{n}{4}}) = \underline{c\, n}$

$\vdots$

$\leq \underline{2} \cdot \underline{c \cdot \frac{n}{2}} = cn$

\# problems ↖ divide + combine steps

$\leq \underline{c\, n}$

$$\sum_{i=0}^{\log_2 n - 1} cn = \log_2 n \cdot cn$$

$\Theta(n \log n)$

Suppose $T(1) = c_1, \ldots, T(m) = c_m$; $T(n) \leq 2 \cdot T(\frac{n}{2}) + cn$ for $n > m$

Find c s.t. a) $T(n) \leq c \cdot n \log_2 n$ for all base cases
b) divide / combine steps work in $\leq cn$ time

Suppose $n \geq m$ and $T(k) \leq c \cdot k \log_2 k$ for $k = 1, \ldots, n-1$

Then $T(n) \leq 2 \cdot T(\frac{n}{2}) + c \cdot n$

$\leq 2 \cdot c \frac{n}{2} \log_2 \frac{n}{2} + cn$

$= cn (\log_2 n - 1) + cn$

$= cn \log_2 n - cn + cn$

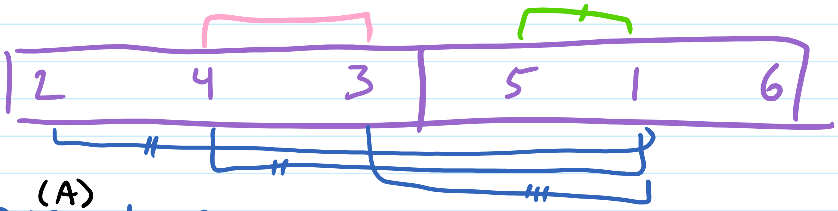
Counting Inversions

Rank the following

	Orioles	Mariners	Padres	Red Sox	Yankees	
Jim	1	2	3	4	5	
Clarence	4	2	3	1	5	5 inversions
Kat	4	3	2	5	1	7 inversions

for 2 orderings, count # inversions

$\Theta(n^2)$ by brute force check every pair



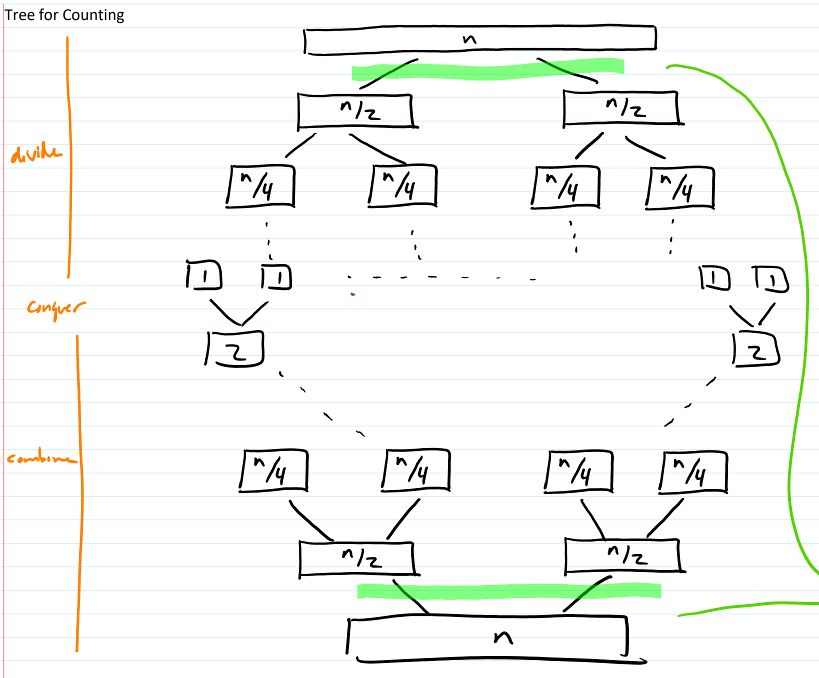
```
COUNT (A)
    if len(A)<2 return 0
    L ← 1st half of A
    R ← 2nd half of A

    countL ← COUNT (L)
    countR ← COUNT (R)
    countM ← COUNT-ACROSS(L,R) *  ← $\frac{n^2}{4}$ pairs to check by brute force
    return        countL+countR+countM
```

w/ brute-force (bad) COUNT-ACROSS

$T(n)$ = # pairs examined on input of size n

$= \underline{2 \cdot T(\frac{n}{2})} + \underline{c \cdot n^2}$

Tree for Counting



assuming straightforward count inversions between L, R

level i: 2^i subproblems of size $\frac{n}{2^i}$

combine step $c\cdot\left(\frac{n}{2^i}\right)^2$ each

$$2^i \cdot c\left(\frac{n}{2^i}\right)^2 = \frac{c\cdot n^2}{2^i}$$

⋮

$$\le \underline{4}\cdot c\left(\frac{n}{\underline{4}}\right)^2 = \frac{1}{4}cn^2$$

$$\le \underline{2}\cdot \underline{c\left(\frac{n}{2}\right)^2} = \frac{1}{2}cn^2$$

$$\le \underline{cn^2}$$

$$T(n) \le \sum_{i=0}^{\log_2 n-1} \left(\frac{1}{2}\right)^i \cdot cn^2$$

$$\le cn^2 \cdot \sum_{i=0}^{\infty} \left(\frac{1}{2}\right)^i$$

$$\le 2c\cdot n^2$$

$\Theta(n^2)$ (since we also have $T(n) \ge$ work @ 1st level $= cn^2$)

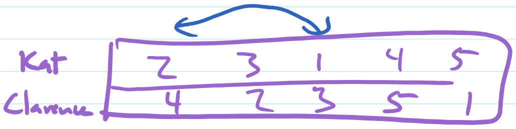
Counting Inversions

```
COUNT_AND_SORT(A)

    L ← 1st half of A
    R ← 2nd half of A

    L, countL ← COUNT_AND_SORT(L)
    R, countR ← COUNT_AND_SORT(R)

    M, countM ← MERGE_AND_COUNT(L, R)

    return M, countL + countR + countM
```

2nd person's pref list in order

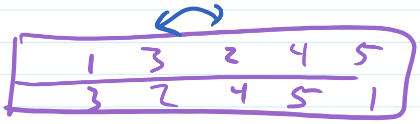
1st person's favorite

1st person's next favorite



Kat	2	3	1	4	5
Clarence	4	2	3	5	1

if just have each person's pref for each of n items sort columns using 1st row as key



1	3	2	4	5
3	2	4	5	1

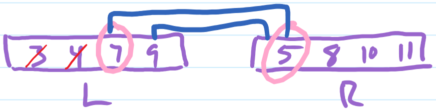
1	2	3	4	5
3	4	2	5	1

```
MERGE_AND_COUNT(L, R)
    count ← 0
    A ← empty list
    i ← 0
    j ← 0
    while i < len(L) and j < len(R)
        if L[i] ≤ R[j]
            append L[i] to A
            i ← i+1
        else
            count ← count + len(L) − i
            append R[j] to A
            j ← j+1

    append L[i ... len(L)−1] to A
    append R[j ... len(R)−1] to A

    return A, count
```

$T(n) \le \underline{2} \cdot T(\textcircled{\tfrac{n}{2}}) + \Theta(n)$

same as merge sort

$T(n)$ is $\Theta(n \log n)$

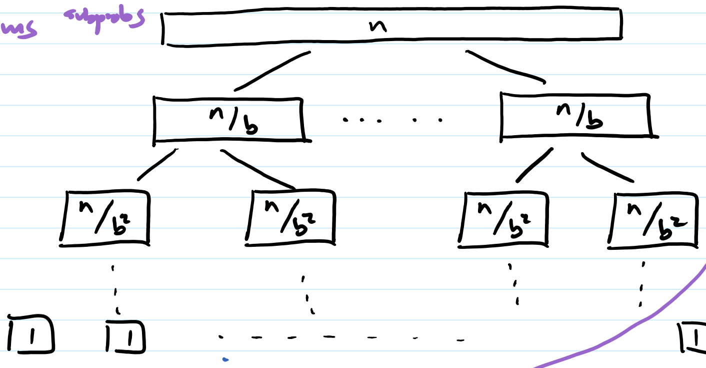


3	4	5	7	8	9	10	11

Generic Tree

$T(1) = k$

$T(n) = \underbrace{a}_{\#\text{ subproblems}} \cdot T(\overset{\text{size of subprobs}}{\tfrac{n}{b}}) + \underbrace{f(n)}_{\text{work for divide + combine}}$



$f(n)$

$a \cdot f(\tfrac{n}{b})$

$a^2 \cdot f(\tfrac{n}{b^2})$

$\vdots$

work in divide/combine

$k \cdot \underbrace{a^{\log_b n}}_{\substack{\#\text{problems} \\ \text{that hit the base case}}} = k \cdot n^{\log_b a}$ work in base cases

$$\sum_{i=0}^{\log_b n - 1} a^i \cdot f\left(\frac{n}{b^i}\right)$$

suppose $f(n) = n^c$

$$= \sum_{i=0}^{\log_b n - 1} a^i \frac{n^c}{b^{i \cdot c}}$$

$$= n^c \cdot \sum_{i=0}^{\log_b n - 1} \left(\frac{a}{b^c}\right)^i$$

$\sum_{i=0}^{\infty} r^i = \frac{1}{1-r} \quad (0<r<1)$

$\frac{a}{b^c} < 1$

$a < b^c$

$\log_b a < c$

$\leq n^c \cdot \underbrace{\sum_{i=0}^{\infty} \left(\frac{a}{b^c}\right)^i}_{} \ \frac{1}{1-r}$ (constant)

$^*\ \Theta(n^c)$

$\frac{a}{b^c} = 1$

$c = \log_b a$

$= n^c \cdot \log_b n$

$\Theta(n^c \cdot \log n)$

$c < \log_b a$

$\sum_{i=0}^{k} r^i = \frac{r^{k+1} - 1}{r - 1} \quad (r \neq 1)$

$$= n^c \cdot \frac{\left(\frac{a}{b^c}\right)^{\log_b n} - 1}{\frac{a}{b^c} - 1}$$

$$\leq n^c \cdot \frac{\frac{a^{\log_b n}}{n^c}}{\frac{a}{b^c} - 1}$$

$$= n^{\log_b a} \cdot \underbrace{\left(\frac{a}{b^c} - 1\right)}_{\text{constant}}$$

$^*\ \Theta(n^{\log_b a})$

note that adding in the $\Theta(n^{\log_b a})$ for work in the base cases does not change these

* technically, we showed O, not Θ ($\leq$, not $=$)
but the Σ is $\geq n^c$ for large enough n too