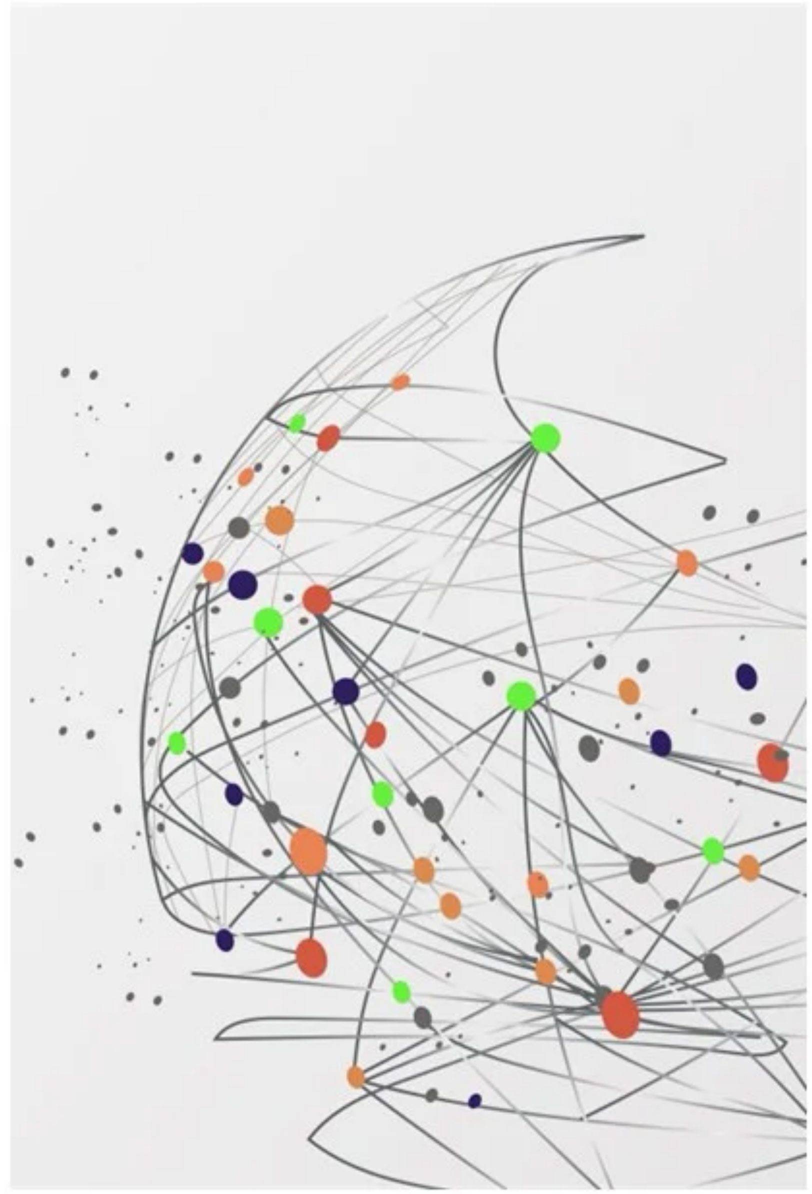
Comp Sci Lunch Social

Come connect!

- ***What***: *delicious food from Tindia (Taste of India)*
- ***Who***: *you, your CS friends, your CS profs*
- ***When***: *Friday May 1st, 12:20-1:20 (drop-in!)*
- ***Where***: *75 Shannon 102*



QUICKSORT

Learning Goals

Use indicator random variables to analyze the average runtime of randomized algorithms [DC4]

Announcements

- Ethics Presentations on Thursday!
- Evening help sessions
- PSet Solutions

Exit Tickets

Review $2/j-i+1$

Why do we care about the base of the log now?

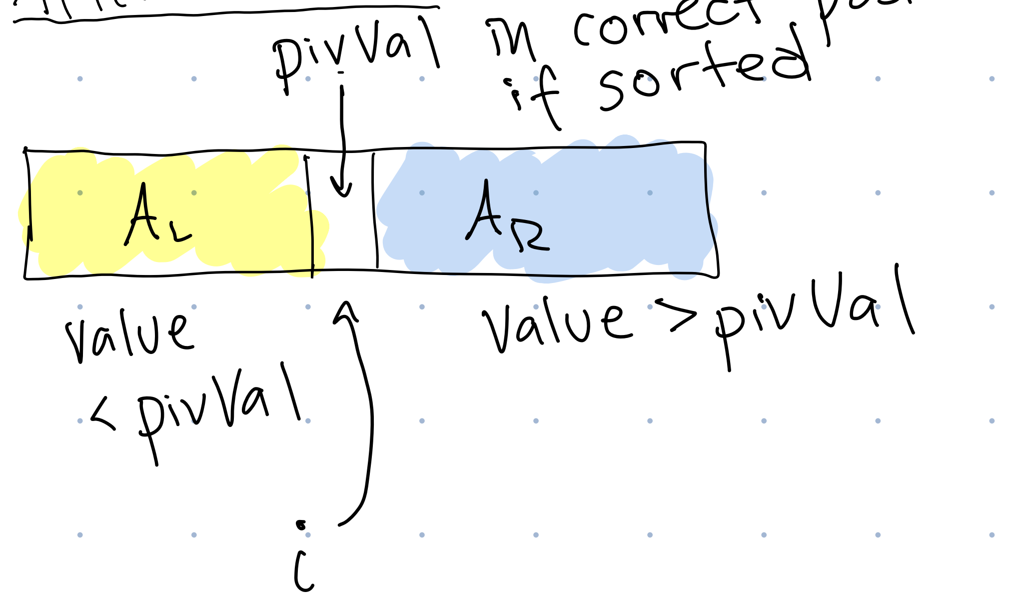
QuickSort

Input: Array A of unique integers
Output: Sorted A

pivInd ⇒ pivot index
pivVal ⇒ pivot value

- If $|A| = 1$: Return A
- pivInd ← randomly chosen index, with value pivVal
- Partition(A, pivInd)
- QuickSort(A_L)
- QuickSort(A_R)

After Partition:

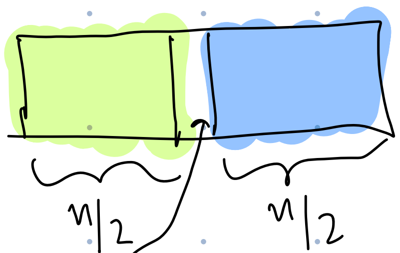


Key Pts

- Partition takes $O(|A|)$ time
- If pivVal is z_i (i^{th} smallest element of A), after partition, pivVal is at position i

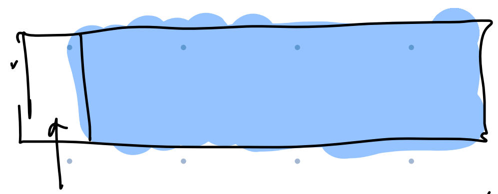
Lucky vs. Unlucky Pivot Choices

Lucky:



pivVal is median array value, $z_{n/2}$

Unlucky:



pivVal is smallest/largest array value z_1 or z_n

1. Suppose you get <u>lucky</u> at every recursive call of QuickSort.

2. Suppose you get <u>unlucky</u> at every recursive call of QuickSort.

   - Create recurrence relation for runtime of QuickSort in each case
   - Solve recurrence to determine runtime in each case

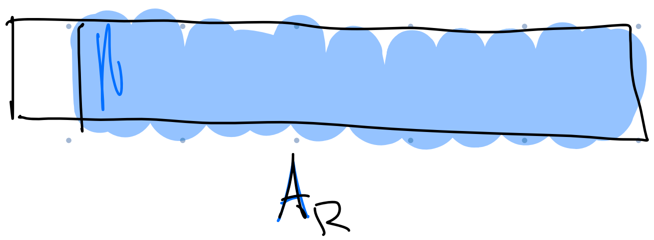
3. What is Sample Space? Random Variable? Expectation value? Linearity of Expectation?

Lucky vs. Unlucky Pivot Choices

1. Suppose you get lucky at every recursive call of QuickSort.

$$T(n) = \begin{cases} 2T\left(\frac{n}{2}\right) + O(n) \\ O(1) \quad \text{if } n=1 \end{cases} \Rightarrow \text{Tree Method} \quad O(n \log n)$$

2. Suppose you get unlucky at every recursive call of QuickSort.



$$T(n) = \begin{cases} T(n-1) + O(n) \\ O(1) \quad \text{if } n=1 \end{cases} \Rightarrow \text{Expand + Hope} \quad O(n^2)$$

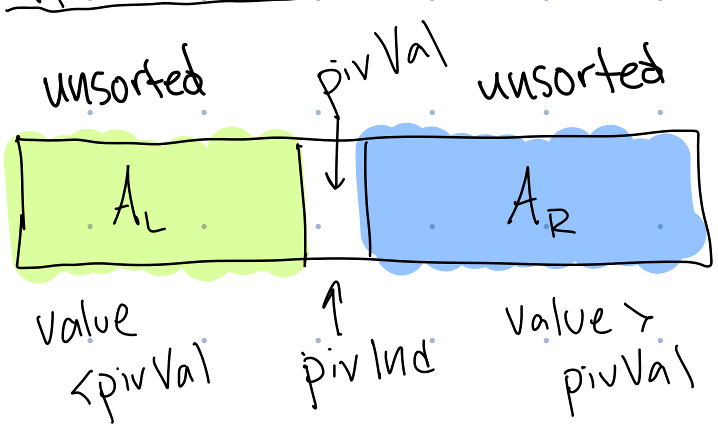
Partition (A, pivInd, pivVal)

- pivVal ← A[pivInd]
- Swap pivot with A[1]
- Current ← 2
- While current ≤ |A|:

```
*  | If A[current] < pivVal:
   |     | Swap A[current], pivVal
   |     | Swap A[pivInd+1], pivVal
   | current ++
```

← every element of array is compared once to pivot

After Partition:



Strategy: count # of times line * is run over whole alg

Analyzing Average Runtime

1. Determine "Sample Space", S = Set of all possible sequences of random events that might occur over the course of the al orithm.

QuickSort → S = set of all possible pivot choice sequences of the algorithm

What is the sample space if QuickSort is run on

8	5	7

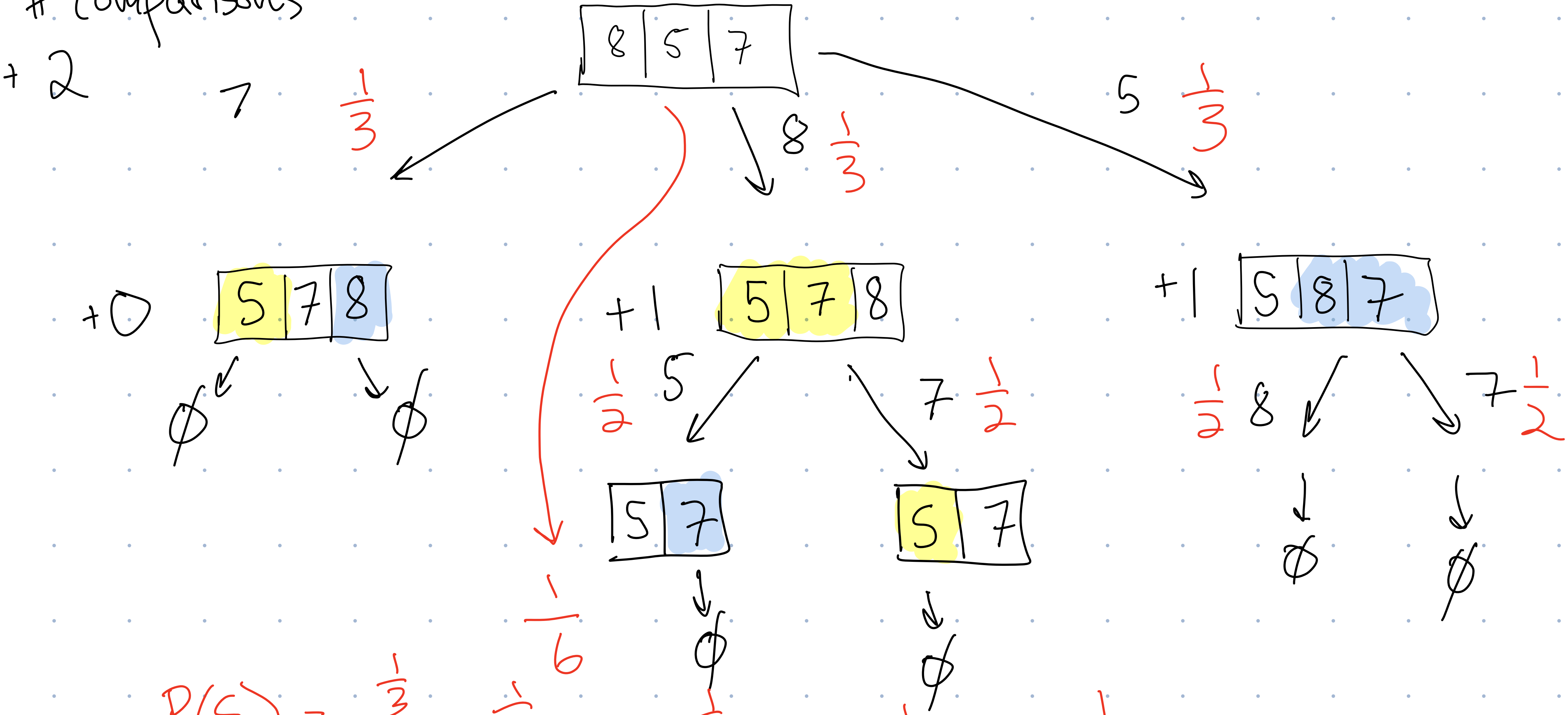
A) S = {8, 5, 7}

B) S = All possible permutations of {8, 5, 7}

C) S = Power set of {8, 5, 7} (set of all subsets of {8, 5, 7})

D) S = {(7), (8, 5), (8, 7), (5, 8), (5, 7)}

comparisons



$P(S) = \frac{1}{3} \quad \frac{1}{6} \quad \frac{1}{6} \quad \frac{1}{6} \quad \frac{1}{6}$

$S = \{ 7, (8,5), (8,7), (5,8), (5,7) \}$

$R(7) = 2 \qquad R(8,5) = 3 \; \ldots$

Analyzing Average Runtime

2. Create "Random Variable" that scales with runtime

$R : S \to \mathbb{R}$

random variable is a function that maps each element of sample space to a number

Particular sequence of pivot choices

QuickSort: $R(\sigma) = \#$ of comparisons $\left(\begin{array}{l} \# \text{ of times} \\ \cancel{\#} \text{ is run} \end{array} \right)$

3. Take Expectation Value of R to get average Runtime:

$$E[R] = \sum_{\sigma \in S} R(\sigma) \cdot p(\sigma)$$

$$\underbrace{R(7)p(7)}_{2 \cdot \frac{1}{3}} + \underbrace{R(8,5)p(8,5) + R(8,7)p(8,7) + \cdots}_{4 \times 3 \times \frac{1}{6}} = \frac{8}{3}$$

2. (Alternate) Create $R: S \to \mathbb{R}$, but break into sum of other simpler random variables

e.g. $f(x) = x + x^2$ $\quad f_1(x) = x$
$f_2(x) = x^2$

$f = f_1 + f_2$

QuickSort: simpler random variable:

$X_{ij}(\sigma)$ = # of comparisons between z_i and z_j over whole algorithm

8	5	7

$z_1 = 5$ $\quad z_2 = 7$ $\quad z_3 = 8$

$X_{23}(8,5) = 1$

$$R(\sigma) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} X_{ij}(\sigma)$$

$$= \sum_{i<j} X_{ij}(\sigma)$$

3. (alternate) Use linearity of expectation

$$\mathbb{E}[R] = \mathbb{E}\left[\sum_{i<j} X_{ij}\right]$$

$$= \sum_{i<j} \mathbb{E}[X_{ij}]$$

To Analyze $E[X_{ij}]$, Consider:

- pivot never in recursive call
- only pivot compared to each other elt

- Suppose z_i, z_j $(i<j)$ are both in a subarray that is input to some recursive call of QuickSort. For each of the following cases (☆)
  - are z_i, z_j compared in this call?
  - are they kept together or separated in future recursive calls

  ☆ z_i or z_j chosen as pivot

  ☆ z_k chosen as pivot

  ☆ $k > i, j$ ☆ $k < i, j$ ☆ $i < k < j$

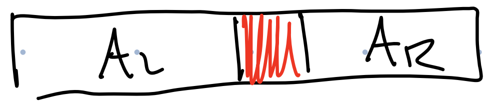
- What values can X_{ij} take (only 2 possible), and under which conditions does it take those values

- What is probability of z_i, z_j being compared?

To Analyze $E[X_{ij}]$, consider:

☆ z_i or z_j chosen as pivot (z_i)

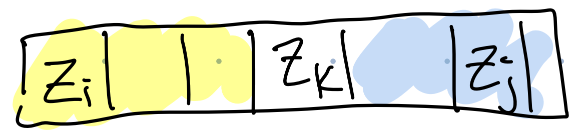
- z_i, z_j are compared
- separated



$X_{ij} = 1$

☆ z_k chosen as pivot, $i < k < j$

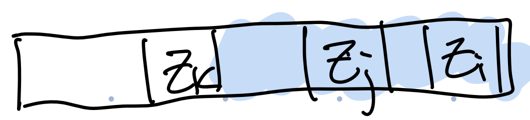
- z_i, z_j not compared
- separated



$X_{ij} = 0$

☆ z_k chosen as pivot, $k < i, j$

- z_i, z_j not compared
- kept together
  - ↳ might be compared
  - ↳ might not be compared



$X_{ij} = ?$

Back to Average Runtime:

Linearity of Expectations

$$\mathbb{E}[R(\sigma)] = \mathbb{E}\left[\sum_{i<j} X_{ij}\right] = \sum_{i<j} \mathbb{E}[X_{ij}]$$

$$= \sum_{i<j}\left(\sum_{\sigma \in S} X_{ij}(\sigma)\Pr(\sigma)\right)$$

$$\underbrace{\sum_{\substack{\sigma \in S \\ \text{s.t. } X_{ij}(\sigma)=0}} X_{ij}(\sigma)\Pr(\sigma)}_{0} + \sum_{\substack{\sigma \in S \\ \text{s.t. } X_{ij}(\sigma)=1}} X_{ij}(\sigma)\Pr(\sigma)$$

$$+ \boxed{\sum_{\substack{\sigma \in S \\ \text{s.t. } X_{ij}(\sigma)=1}} \Pr(\sigma)}$$

$$= \text{Probability that } X_{ij} = 1$$

Probability that $X_{ij} = 1$

z_1 z_n

z_1 z_n $\frac{2}{n}$

$$\underbrace{z_1 \; z_2 \; z_3 \; \cdots \; z_{i-1}}_{\text{Pivot here, delay decision}} \; \underbrace{z_i \; z_{i+1} \; \cdots \; z_{j-1} \; z_j}_{\text{Pivot here, decision}} \; \underbrace{z_{j+1} \; \cdots \; z_n}_{\text{Pivot here, delay decision}}$$

What is the probability that z_i, z_j are compared?

A) $\frac{1}{j-i}$ B) $\frac{2}{j-i+1}$ C) $\frac{2}{n}$ D) $\frac{1}{n^2}$

Continuing $\mathbb{E}[R]$ analysis:

Indicator random variable = random variable that only takes value 0 or 1.

$$\mathbb{E}[R] = \mathbb{E}\left[\sum_{i<j} X_{ij}\right]$$

$$= \sum_{i<j}\left[\mathbb{E}[X_{ij}]\right]$$

thing that makes i.r.v take value 1

$$= \sum_{i<j}\left(\underbrace{\text{Probability that } z_i \text{ and } z_j \text{ are compared}}\right)$$

$$= \sum_{i<j} \frac{2}{j-i+1}$$

when $j=i+1$ $\quad \dfrac{2}{(i+1)-i+1} = \dfrac{2}{2}$

$$= \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{2}{j-i+1}$$

Harmonic Series

$$= \sum_{i=1}^{n-1}\left(\frac{2}{2} + \frac{2}{3} + \frac{2}{4} + \cdots \frac{2}{n-i+1}\right)$$

$$< \sum_{i=1}^{n-1}\left(\frac{2}{2} + \frac{2}{3} \cdots \frac{2}{n-i+1} + \cdots \frac{2}{n}\right)$$

Fact: $\sum_{i=1}^{n} \frac{1}{n} \leq \ln(n) + 1$

$\ln(n) = \log_e(n)$

$$\leq \sum_{i=1}^{n-1} 2(\ln(n)+1)$$

$$= 2(n-1)(\ln(n)+1)$$

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

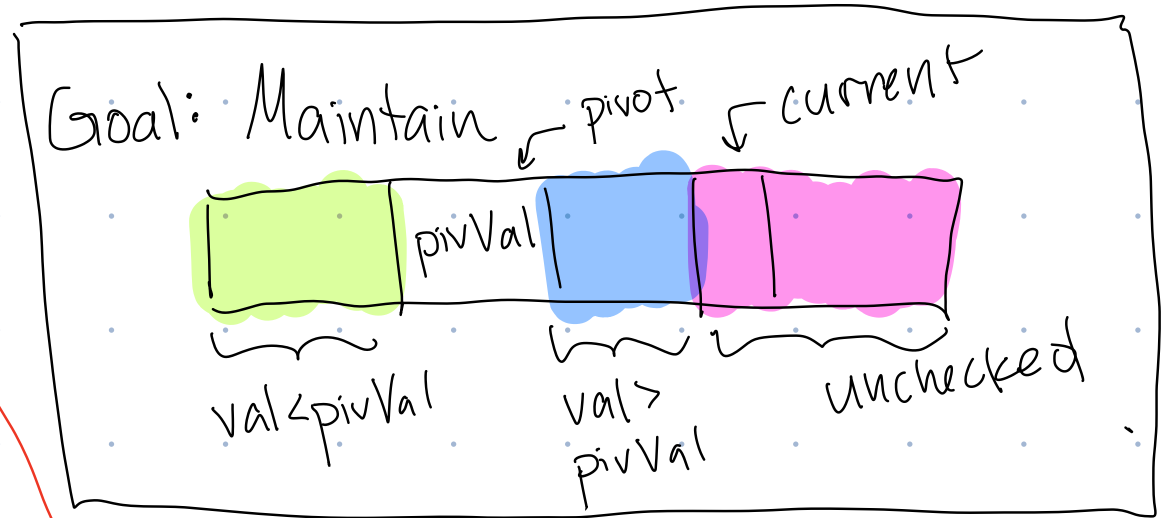
Prob.
Runtime

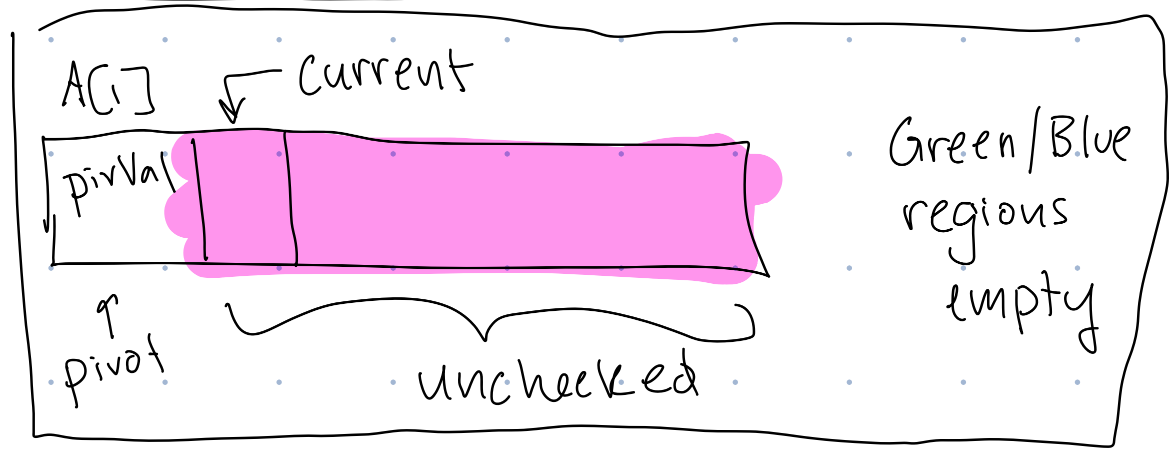
<u>MergeSort</u>? or <u>QuickSort</u>?

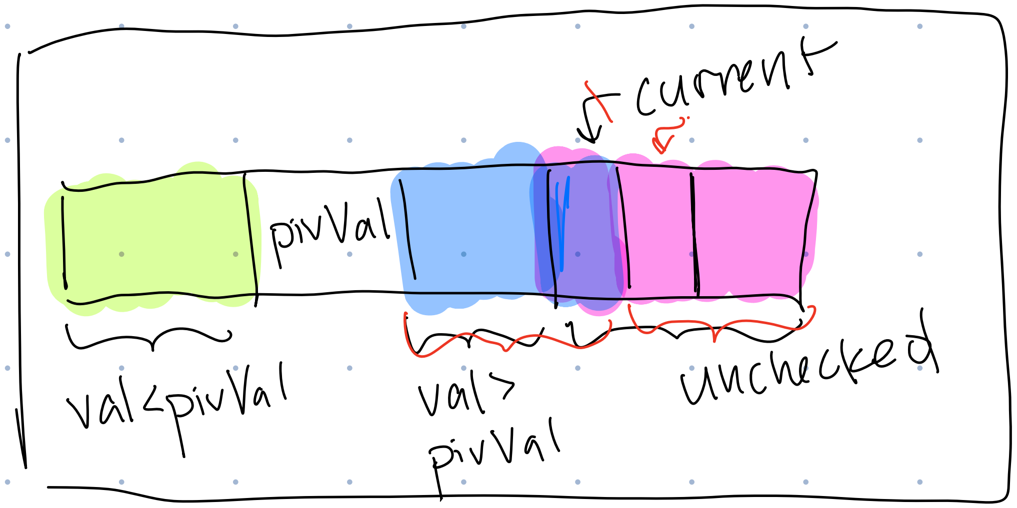
- Limited Space?
- Sorting Multiple Lists in Parallel?
- Array as linked list?
- Small Array
- Want speed, and array calls are quick?

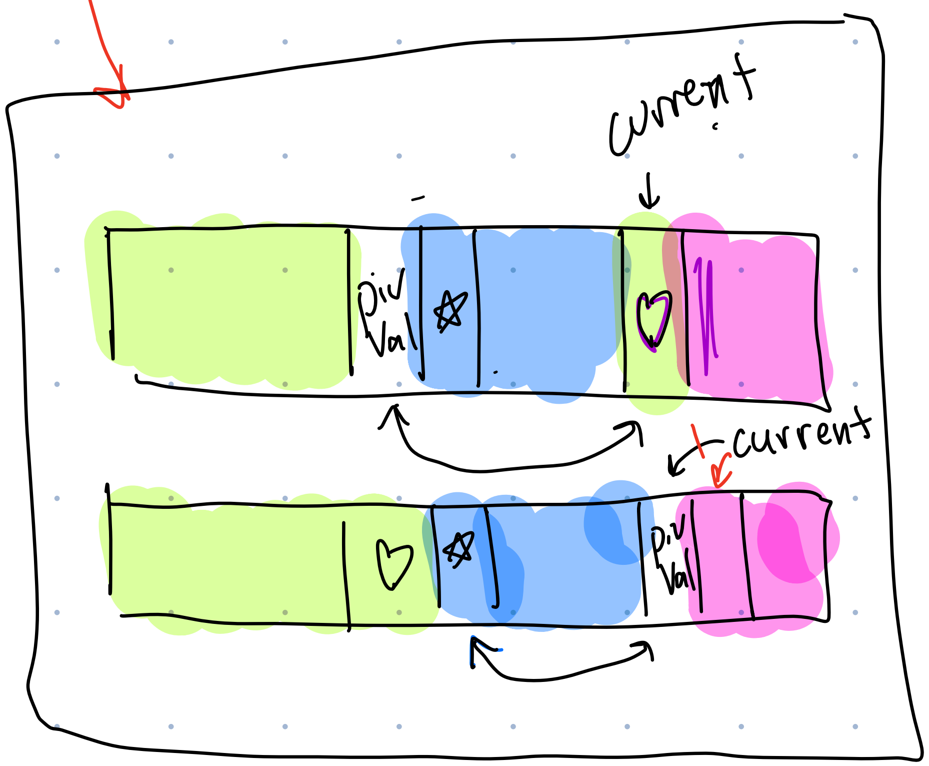
Partition (A, pivInd, pivVal)

- Swap pivot with A[1]
- Current ← 2
- While current ≤ |A| :
  - If A[current] < pivVal:
    - Swap A[current], pivVal
    - Swap A[pivInd+1], pivVal
  - current ++









End: