Goals

- Describe impact of pivot choice on runtime of quicksort
- Evaluate expectation value of random variables

**Input:** Array $A$ of length $n$, no repeated elements

**Output:** Array with sorted elements

**QuickSort** (array $A$)

1. If $|A| = 1$: return $A$
2. $\text{pivot} = \text{ChoosePivot}(A)$
3. $\text{Partition}(A, \text{pivot})$
4. $A = A_L \mid \mid A_R$
5. QuickSort($A_L$) \{conquer\}
6. QuickSort($A_R$)

**Partition**

1. Move pivot to front of array
2. Maintain invariant

Diagram:

- $p$: pivot
- $\text{val} \leq p$
- $\text{val} > p$
- $\text{unchecked}$
Runtime of Quicksort is $O(\# \text{ of comparisons})$

Pf: Partition does most of the work, and runtime of partition is $O(\# \text{ of comparisons})$.

Q: How many comparisons are done by Partition on input array of size $n$?

A: $O(n^2)$  B: $O(n)$  C: $O(n \log n)$  D: $O(n^2)$

Q: What is the runtime of Quicksort when the pivot is always chosen to be $(\frac{n^2}{2})^{th}$ largest element of array?

A: $O(n\sqrt{n})$  B: $O(n)$  C: $O(n \log n)$  D: $O(n^2)$
Runtime of Quicksort is $O(\# \text{ of comparisons})$

Pf: Partition does most of the work, and runtime of partition is $O(\# \text{ of comparisons})$.

Q: How many comparisons are done by Partition on input array of size $n$?

A: $O(n^2)$  B: $O(n)$  C: $O(n \log n)$  D: $O(n^2)$

Current increases by 1, goes from $1 \to n$.

Q: What is the runtime of Quicksort when the pivot is always chosen to be $(\frac{n}{2})^{th}$ largest element of array?

A: $O(n^2)$  B: $O(n)$  C: $O(n \log n)$  D: $O(n^2)$

Use master method

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

\[\text{QuickSort on } A_L, |A_L| = \frac{n}{2}\]

\[\text{QuickSort on } A_R, |A_R| = \frac{n}{2}\]
Q: What is the run time of QuickSort if the pivot is always chosen to be the smallest item in array? A: $O(n)$ B: $O(n \log n)$ C: $O(n^{3/2})$ D: $O(n^2)$

Partitioning takes time $n$

AL size is 0

AR size is $n-1$

AL size is 0

AR size is $n-2$

Run time: $n + n - 1, \ldots, 1 = O(n^2)$  
Formula: $\frac{n(n-1)}{2}$

Choice of pivot important.

<table>
<thead>
<tr>
<th></th>
<th>Bad Choice</th>
<th>Good Choice</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run Time</td>
<td>$O(n^2)$</td>
<td>$O(n \log n)$</td>
</tr>
</tbody>
</table>

We will show: random choice of pivot is good!
Steps to Calculate Average Runtime of Randomized Alg. (Naive)

1. Describe sample space: $S$ of possible random outcomes
2. Calculate probability $p(s)$ for each $s \in S$
3. Create random variable $R: S \rightarrow \mathbb{R}$ where $R(s)$ = runtime with outcomes $s \in S$, and evaluate for each $s \in S$
4. Evaluate: $E[R] = \sum_{s \in S} R(s)p(s)$

1. **Sample space** $S$ = set of possible outcomes of random choices

Example:

\[
\begin{array}{c}
8 \\
5 \\
7 \\
\end{array}
\]

$S$ = set of possible pivot choices of the algorithm.

What is $S$?

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

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

C) $S = \{(7, 5, 7), (5, 8), (8, 5), (8, 7)\}$
Sample space $S = \text{set of possible outcomes}$

Example:

$8 \ 5 \ 7$

$S = \text{set of possible pivot choices of the algorithm}$

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

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

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

If 7 chosen: Arrange too small, no partition

If 5 chosen: 8 or 7 can be chosen as pivot

If 8 chosen: 5 or 7 can be chosen as pivot

For large arrays... it will be a mess
Step 2: Describe probability

**Probability** \( P : S \rightarrow \mathbb{R} \) \( P(\sigma) \) = probability of outcome \( \sigma \) occurring

If pivot is chosen randomly at each recursive call of partition, what is \( P(5, 7) \)?

A) \( \frac{1}{6} \)

B) \( \frac{1}{5} \)

C) \( \frac{1}{3} \)

D) Impossible to determine
If pivot is chosen randomly at each recursive call of partition, what is $P(5, 7)$?

- A) $\frac{1}{6}$
- B) $\frac{1}{3}$
- C) $\frac{1}{3}$
- D) Impossible to determine

... For large arrays it is going to be a mess

Calculate $R(s)$ for each $s \in R$... makes my head hurt...
**Better Approach**

1. Describe (generally) the sample space
   
   ex: \( S = \{ s : s \text{ is a sequence of possible pivot choices for our array} \} \)

2. Describe (generally) the run time random variable
   
   ex: \( R(s) = \# \text{ of comparisons over algorithm with pivot choices } s \)

3. Write \( R = \sum_{i} X_i \)
   
   ex: \( X_{ij}(s) = \# \text{ of times } i^{th} \text{ largest element + } j^{th} \text{ largest element are compared with pivot choices } s \)
   
   \( R(s) = \sum_{i,j} X_{ij}(s) \)  
   
   To create \( X_i \), think about what are the basic events that are adding an amount of 1 to \( R \)

4. \( \mathbb{E}[R] = \sum_{i} \mathbb{E}[X_i] \)
   
   linearity of expectation

   \[ \mathbb{E}[X_i] = \sum_{s \in S} X_i(s) p(s) \]

   If choose \( X_i \) carefully, this is easier to calculate.

   Best \( X_i = \text{ indicator random variable: takes value 0 or 1} \).