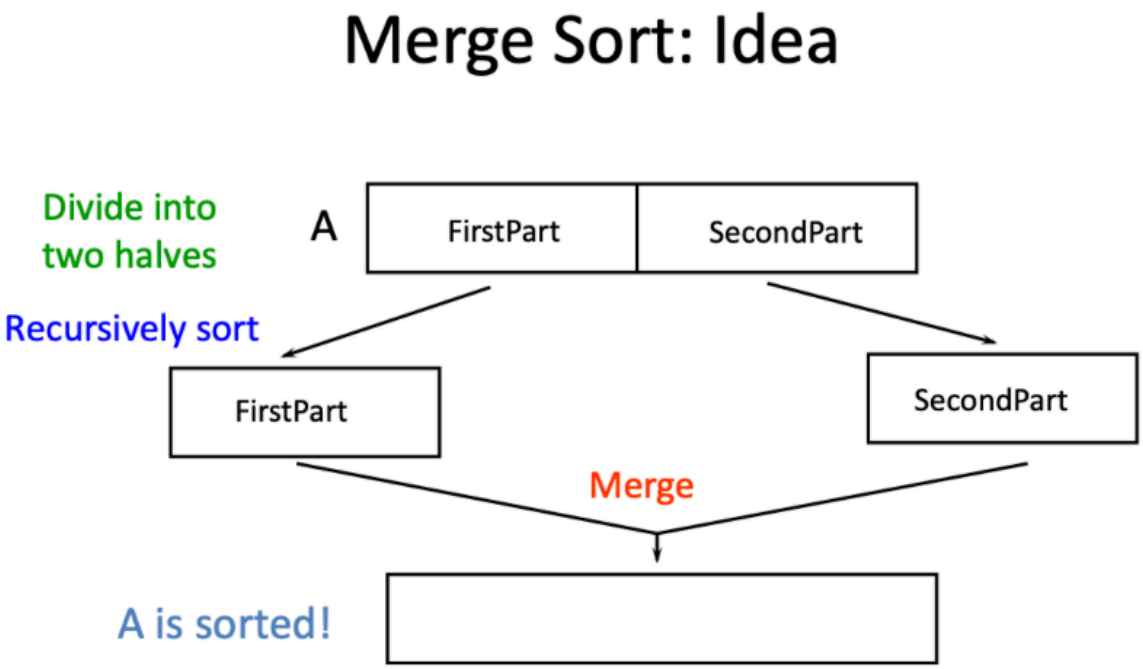
- Fast algorithms
  - **Divide and Conquer**
    1. Base Case, solve the problem directly if it is small enough
    2. Divide the problems into 2+ similar and smaller subproblems
    3. Recursively solve the problem
    4. Combine solutions to the subproblems
  - **Merge Sort**
    1. Base case at most one element (left $\geq$right), return
    2. Divide A into two subarrays: FirstPart, SecondPart
        1. sort FirstPart, sort SecondPart
    3. Recursively sort FirstPart, sort SecondPart
    4. Combine sorted FirstPart and sorted SecondPart



    - Time $\theta(n \log n)$
  - **Quick Sort**
    - Divide: Pick any element p as the piviot point (aka first element)
    - Partition the remaining elements greater than p and less than p
  - Recursively sort the FirstPart and SecondPart
  - Combine: no work is necessary since sorting is done in place
  - Time - Most of the work done in partitioning
    - Average takes O(n log(n)) time

- Worst takes O(n^2)) time