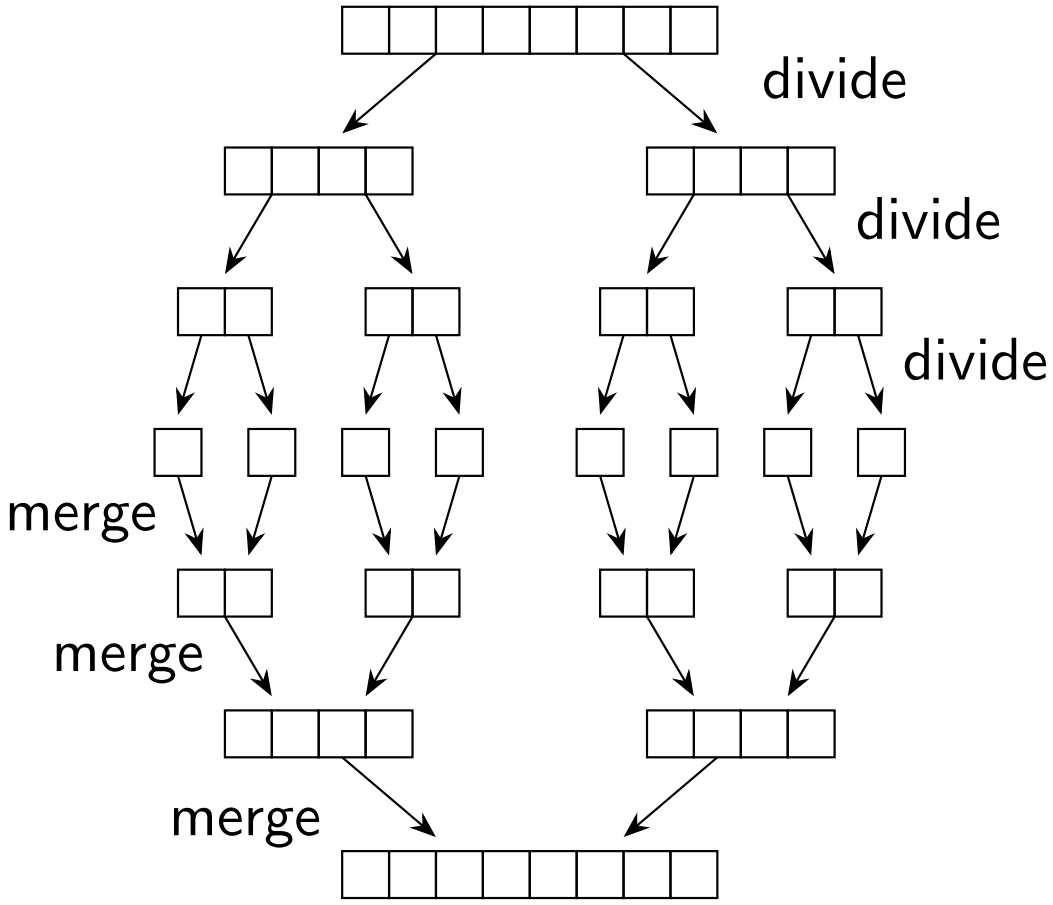
Merge in MIPS

Class 42

Mergesort

- mergesort is a very popular sorting algorithm
- it uses the divide-and-conquer algorithm design pattern, which is a very popular design technique:
  1. partition current problem instance into non-overlapping smaller problem instances
  2. solve smaller instances separately (often recursively)
  3. combine small instance solutions into larger instance solution (not always necessary, depending on problem)

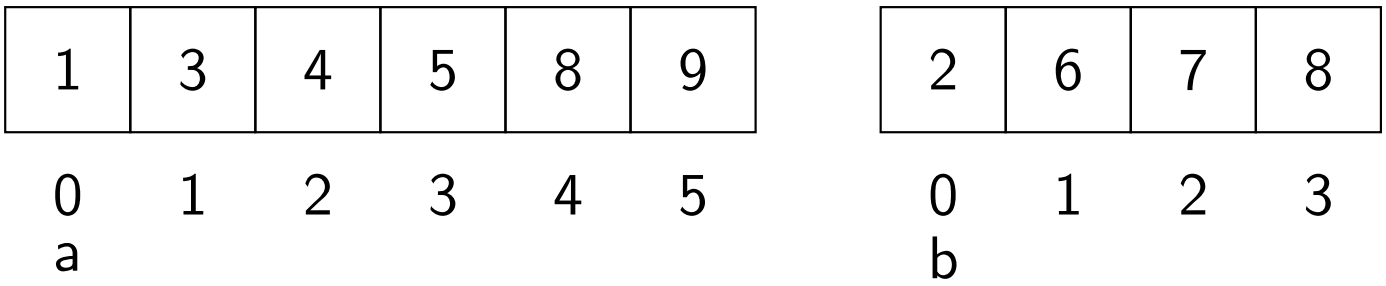
Mergesort

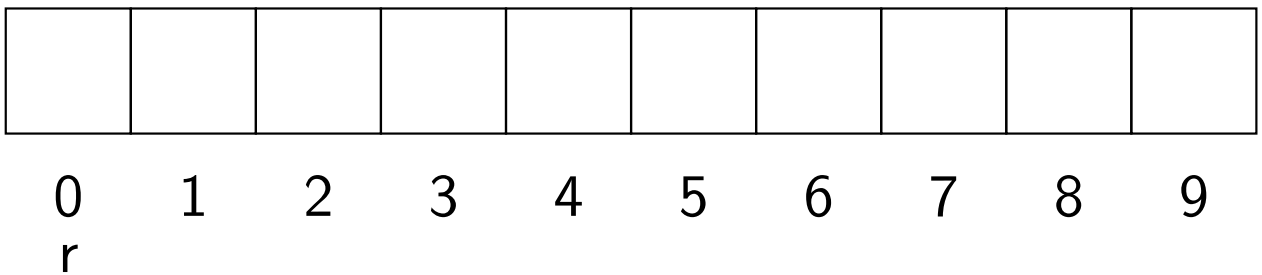


Merge

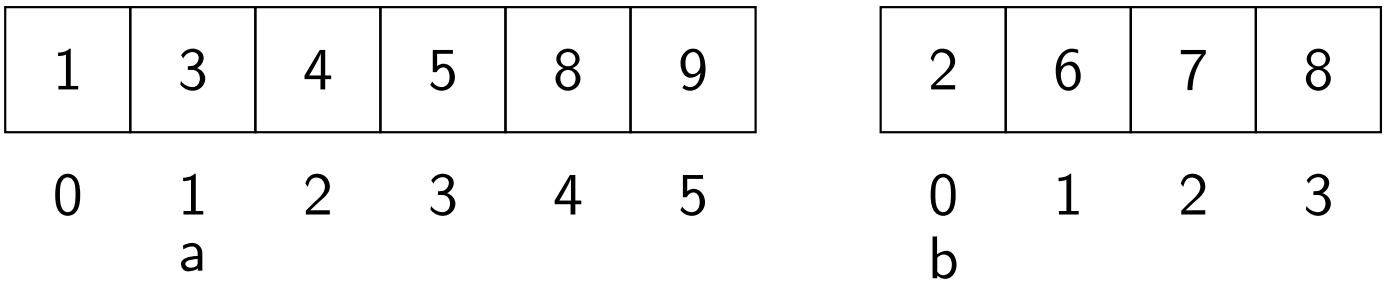
- mergesort requires the merge operation
- two input arrays `a` and `b`, each already sorted, with m and n elements respectively
- one output array `result`, size $m + n$
- three indices `index_a`, `index_b`, and `index_r`, one for each array, all starting at 0

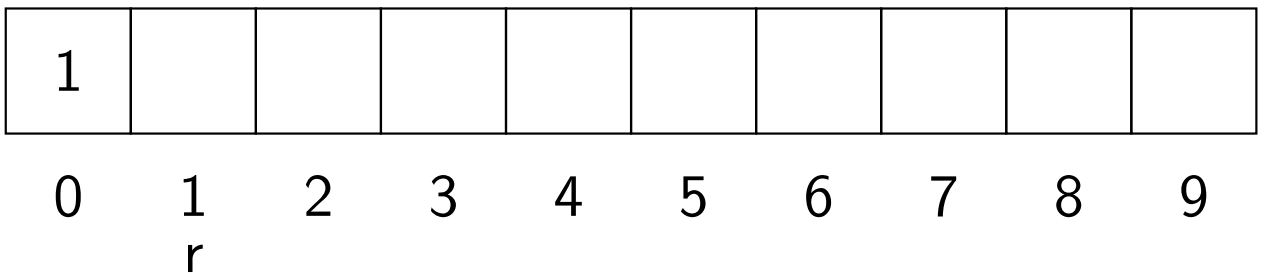
Merge 0



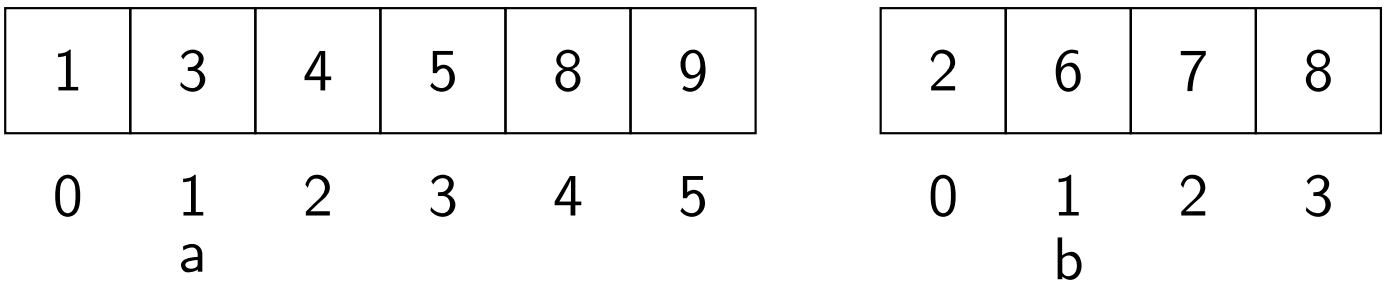


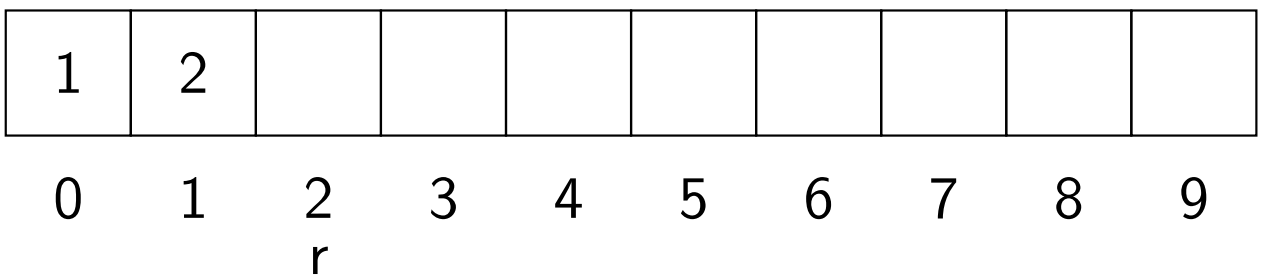
Merge 1



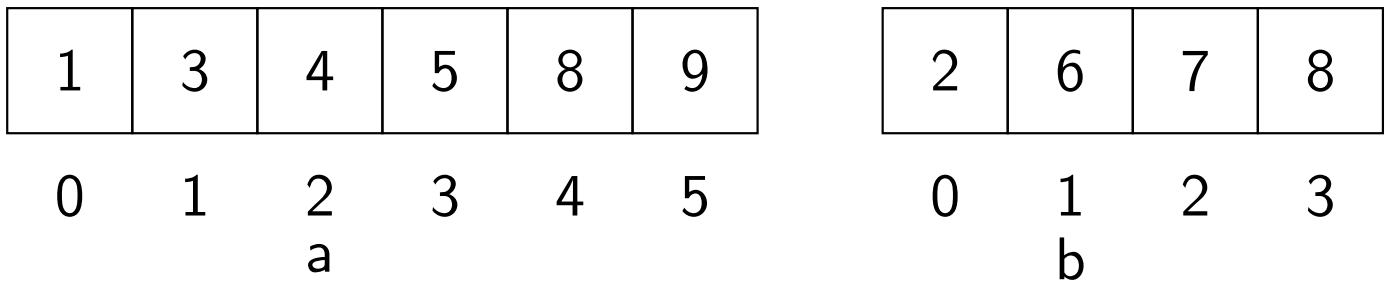


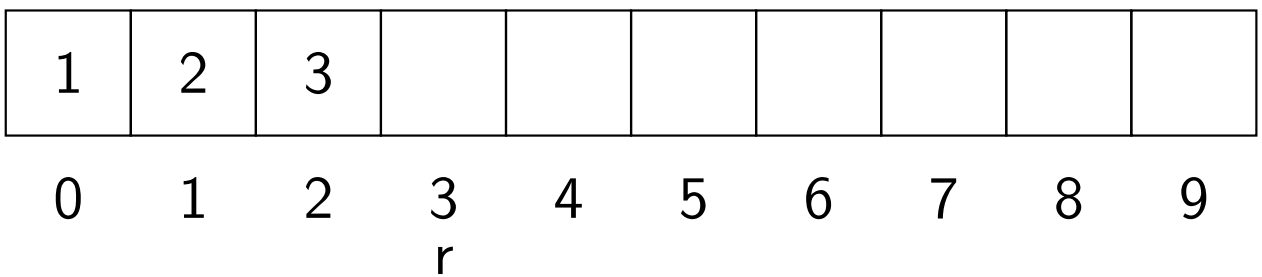
Merge 2



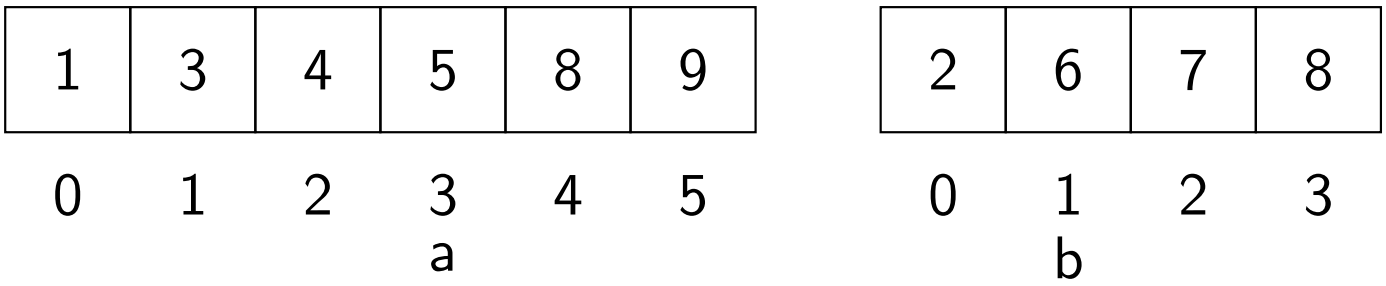


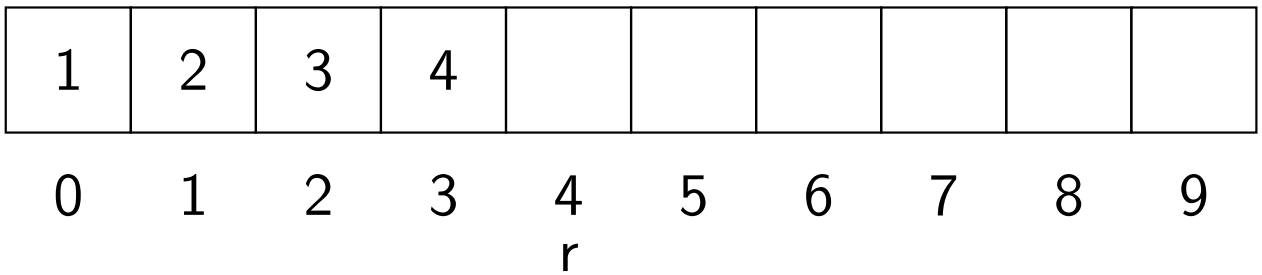
Merge 3



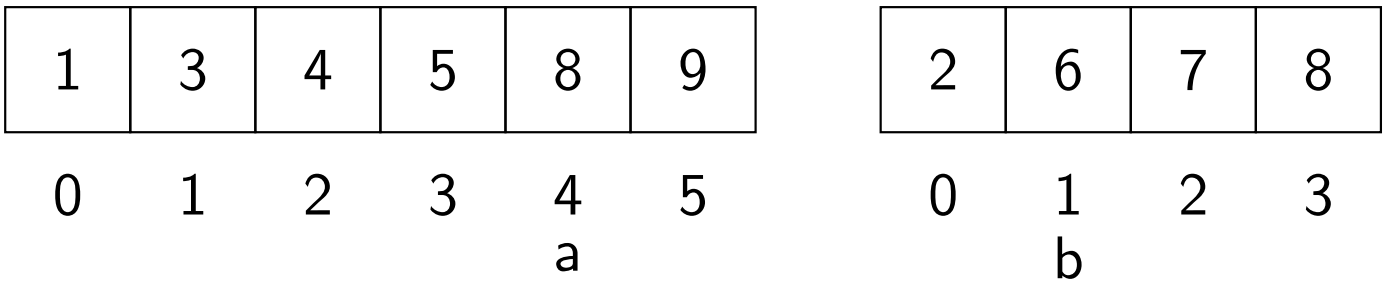


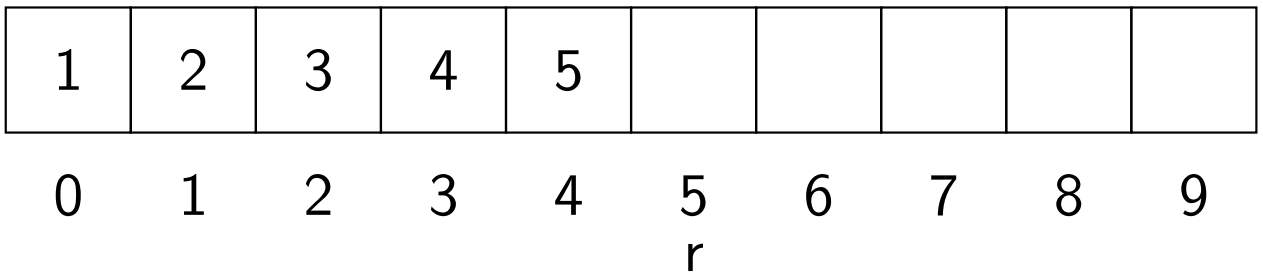
Merge 4



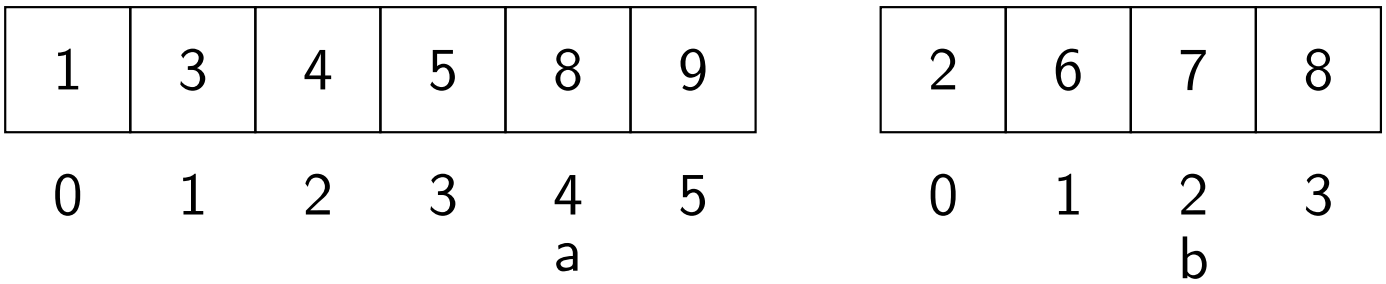


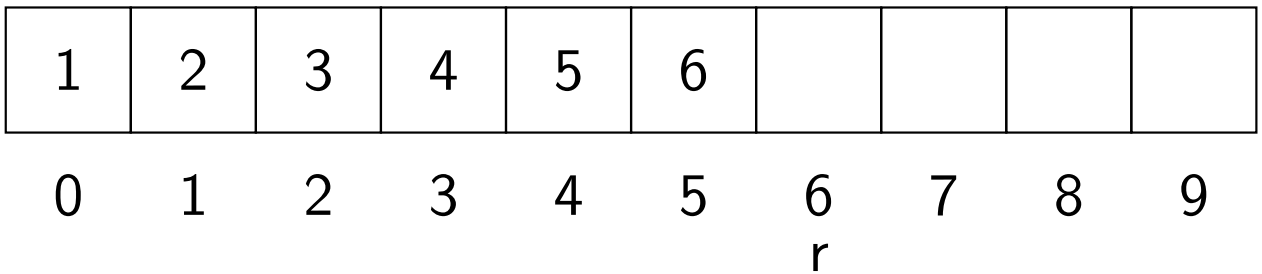
Merge 5



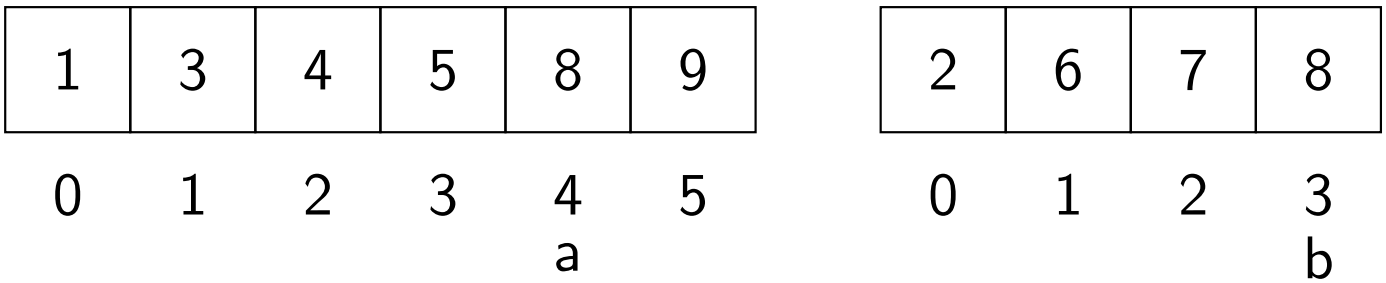


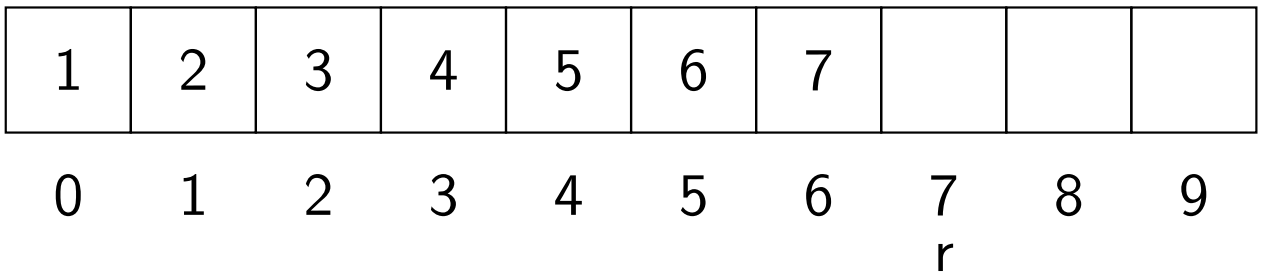
Merge 6



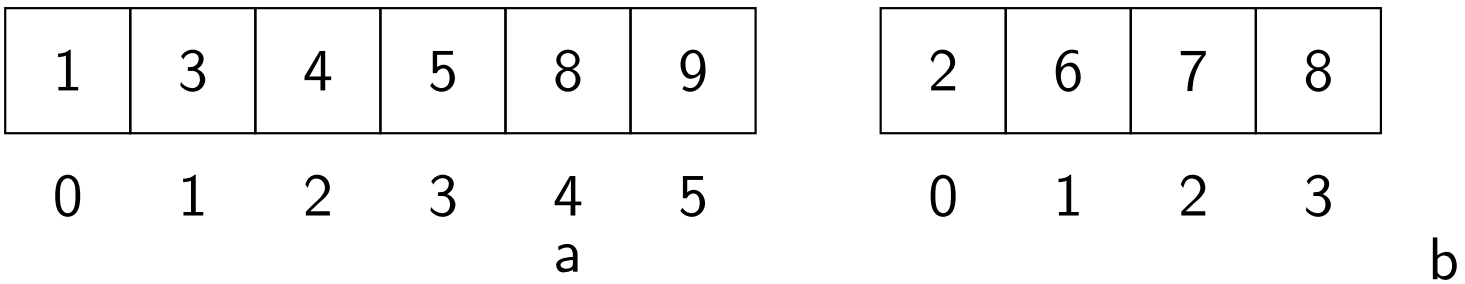


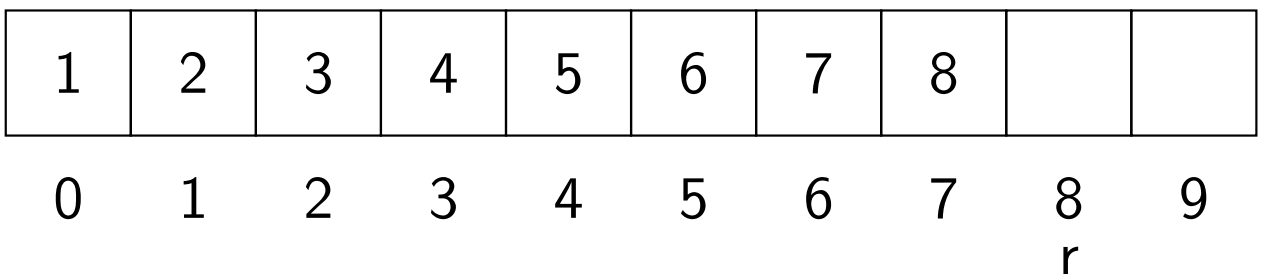
Merge 7



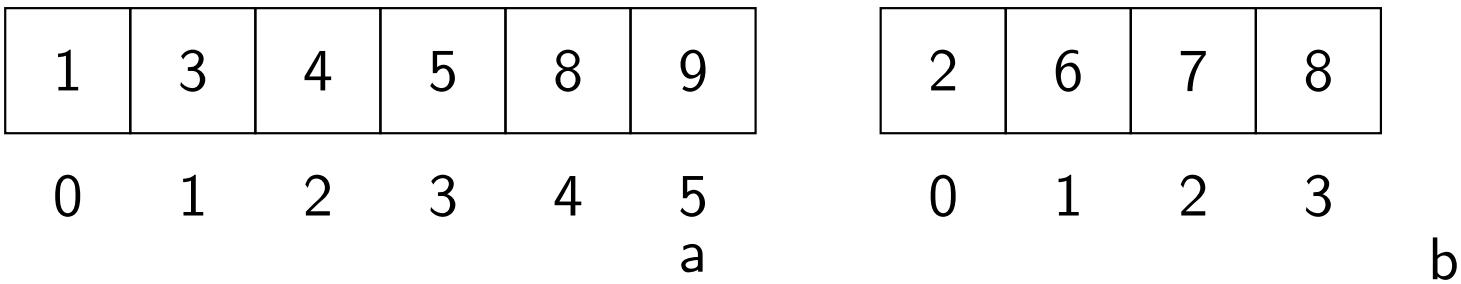


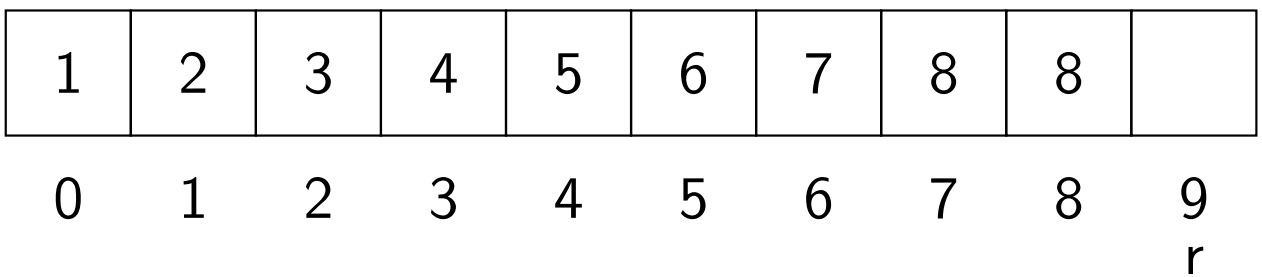
Merge 8



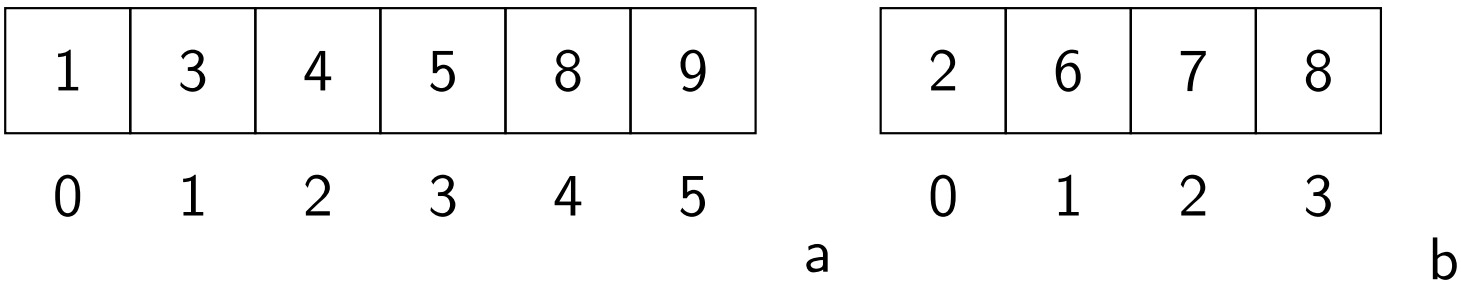


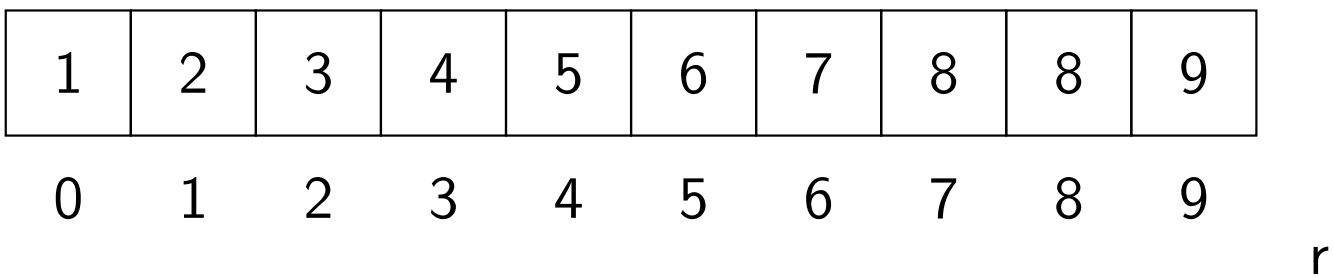
Merge 9





Merge 10





Merge Implementation

- a brief note on implementation
- in real implementations, there are not three arrays (a, b, and result)
- there is just one array, partitioned into two contiguous ranges
- the "result" is simply the combined range

- it's much easier to understand with three separate arrays, so that's how we will implement it