

| | | |
|-------|-------|---------|
| Name: | Date: | Period: |
|-------|-------|---------|

Lab13: Rank-Merge

- Implement the surplus-log binary search rank-merge algorithm, page 25.
- Attach a code printout.
- Then...
 - Partitioned rank-merge.
 - Only binary search every “log n”th element in each array.
 - * There are $O(n/\log n)$ such elements selected.
 - * Use these select elements as the partition endpoints.
 - Serial rank-merge the $O(\log n)$ remaining elements of each partition.
 - * The $O(n/\log n)$ partitions are now independent!
 - * These serial rank-merges can happen in parallel!!
 - * And $O(\log n) \cdot O(n/\log n) = O(n)$ Work, wow!!!
 - Given:

| | | | | | | | | | |
|-------|---|---|---|---|----|----|----|----|----|
| Index | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| Array | 4 | 6 | 8 | 9 | 16 | 17 | 18 | 19 | 20 |
| Array | 1 | 2 | 3 | 5 | 7 | 10 | 11 | 12 | 13 |

- Which elements are selected to be the endpoints of the partitions?
- Which groups of remaining elements are serial rank-merged together?

Official Use Only

| | | | |
|------------------|-----------|----------------|---------------------|
| Header: | Name | Correct Date | Program Description |
| Style: | Comments | Variable Names | Modular |
| Data Structures: | Obvious | General | Lean |
| Algorithm: | Clear | Correct | Efficient |
| Scoring: | Raw _____ | Late _____ | Total _____ |