Problem 1. Consider the merging problem in our PRAM discussion. Let A_1 be the array $(1, 17, 28, 29, 55, 61, 69, 80)$ and A_2 be the array $(10, 13, 25, 33, 38, 56, 72, 75)$. Give the content of array B_1.

Solution. $B_1 = (1, 4, 6, 7, 10, 12, 13, 16)$.

Problem 2. Consider the sorting problem in EM. Let A be the input file of n integers (which is stored in $O(n/B)$ blocks). Give an algorithm to produce $O(n/M)$ files satisfying all the following requirements:

- Each file stores at most M integers of A in ascending order using $O(M/B)$ blocks.
- All the files are mutually disjoint.
- The union of all the files is the set of integers in A.

Your algorithm must terminate in $O(n/B)$ I/Os.

Solution. Load the first M numbers of A into memory, sort them, and then write the sorted list into a file of $O(M/B)$ blocks. Then, do the same to the next M numbers of A. Repeat until having exhausted A.

Problem 3. This question concerns the PRAM model. Suppose that we have already obtained a sorting algorithm $\mathcal{A}$ finishing in $f(n)$ steps when the number p of CPUs equals n (recall that n is the number of integers to sort). Consider now the scenario where $p < n$. Describe how to use $\mathcal{A}$ to design an algorithm that finishes in $O(\frac{n}{p} \cdot f(n))$ steps.

Solution. The key is to simulate a step of $\mathcal{A}$, which runs on n CPUs, by performing $O(n/p)$ steps with p CPUs. I suggest giving a full mark as long as the student manages to make the above observation.

For a complete answer, however, we must clarify the details of the reduction. Let $\Upsilon_1, \Upsilon_2, ..., \Upsilon_n$ be the n CPUs of $\mathcal{A}$; call them the *$\mathcal{A}$-CPUs*. Define the *state* of each Υ_i ($i \in [1, n]$) as the current set of register values in Υ_i. To simulate $\mathcal{A}$, we preserve the state of every Υ_i in $O(1)$ special memory words.

We simulate every step of $\mathcal{A}$ — call this an *$\mathcal{A}$-step* — in $O(n/p)$ steps through $\lceil n/p \rceil$ phases. The i-th ($i \in [1, n]$) phase executes the atomic operations performed by $\Upsilon_{i \cdot p+1}, \Upsilon_{i \cdot p+2}, ..., \Upsilon_{(i+1) \cdot p}$ during the $\mathcal{A}$-step. For this purpose, we first perform $O(1)$ steps to load the state of Υ_j, for each $j \in [i \cdot p + 1, (i + 1) \cdot p]$, into the ($j \mod p$)-th CPU. The atomic operations of the p CPUs can then be carried out in another step. Because $\mathcal{A}$ is a CREW algorithm, no two CPUs can have a conflict in memory accesses during an $\mathcal{A}$-step. Therefore, we guarantee that the states of all $\mathcal{A}$-CPUs after the $\lceil n/p \rceil$ phases are identical to those after the $\mathcal{A}$-step.

Problem 4. Give a PRAM algorithm that settles the sorting problem in $O(\frac{n}{p} \log^2 n)$ steps.

Solution. Apply the reduction in Problem 3 to our discussion in the seminar.