Information
Solutions in english or german are fine.

10.1. Problem (10) \textit{Better splitters for Sample-Sort}

We consider the following procedure to determine a set of \( p - 1 \) good splitters. After sorting its \( \frac{p}{p'} \) keys a process selects keys in position \( i \cdot \frac{p}{p'} \) for \( i = 0, \ldots, p - 1 \) as its sample and sends this sample to process 1, who then sorts all \( p^2 \) keys. Then process 1 computes the final sample \( S \) by selecting all keys in positions \( i \cdot p \) for \( i = 1, \ldots, p - 1 \) and broadcasts \( S = \{ s_1 < s_2 < \ldots < s_{p-1} \} \).

Show that at most \( \frac{2n}{p} \) keys are smaller than \( s_1 \) (resp. in between \( s_{k-1} \) and \( s_k \), or larger than \( s_{p-1} \)).

10.2. Problem (10+10) \textit{Odd-Even Merge}

Let \( m \) be a power of two. For a sequence \( x = (x_0, \ldots, x_{m-1}) \) let \( \text{even}(x) = (x_0, x_2, x_4, \ldots, x_{m-2}) \) and \( \text{odd}(x) = (x_1, x_3, x_5, \ldots, x_{m-1}) \) be the subsequences of even- and odd-numbered components of \( x \).

Assume that \( x \) and \( y \) are sorted sequences of length \( m \) each. To merge \( x \) and \( y \), odd-even Merge recursively merges

- \( \text{even}(x) \) and \( \text{odd}(y) \) to obtain the sorted sequence \( u = (u_0, u_1, \ldots, u_{m-1}) \) and
- \( \text{odd}(x) \) and \( \text{even}(y) \) to obtain the sorted sequence \( v = (v_0, v_1, \ldots, v_{m-1}) \).

Finally odd-even merge computes the output sequence \( w = (w_0, \ldots, w_{2m-1}) \) from \( u \) and \( v \) as follows: in parallel for \( 0 \leq i \leq m - 1 \), if \( u_i \leq v_i \) then set \( w_{2i} = u_i, w_{2i+1} = v_i \) and otherwise \( w_{2i} = v_i, w_{2i+1} = u_i \).

(a) Show with the 0-1 principle that odd-even merge works correctly.

(b) Assume that consecutive intervals of \( x \) and \( y \) of length \( m/2q \) each---are distributed among \( q \) processes. Show how to implement odd-even merge in computing time \( O\left(\frac{m}{q} \cdot \log_2 2q\right) \) and communication time \( O\left(\frac{m}{q} \cdot \log_2 q\right) \).