Homework 2

My algorithm works like this:

- First I generate $\frac{n}{p}$ integers on each process.
- Then I jump directly into the recursive step:
  - I choose the pivot using the algorithm where each process picks a random element, and the median of those is picked.
  - The way I moved S and L arrays around is:
    1. First `MPI_Alltoallv` the plan for *which* processors are going to be sent to, including exact calculations of which local index is being copied from and to.
    2. Then, each processor loops through all the processors and if they have something to send, they send it.
    3. This way, I can coordinate all of the senders/receivers and the ones with nothing to send don't do anything.
  - For the recursion, I opted to make the recursive step have different lengths. (**NOTE:** The reason I have a different "capacity" than "length" is because for the `displs` array I opted to have them all be the same length, so there's extra padding on the shorter ones)
    - If the boundary between S and L falls between a $\frac{n}{p}$ segment, I'd extend the one before and shorten the one after.
    - Then, I recursively process all the S's and all the L's separately using `MPI_Comm_split`.
    - Once it's done processing, I reverse the exact operation that extends / shortens the arrays. This ensures everything is always back to $\frac{n}{p}$ at the end.
- Everything is collected back at the end via a `Send`/`Recv` to save on allocations.

Allocations are all on the order of $O\left(p + \frac{n}{p}\right)$.

Unfortunately I didn't finish debugging segfaults in time, and have this report prepared for the parts of the assignment that I *did* do. It works on small integers (capped at 100) but for some reason segfaults at address `(nil)` at the end... I spent several hours debugging but have not discovered how this occurs.