CSCI653 Assignment 3—Hypercube Quicksort
Due: Monday, September 30, 2019

Write an MPI program to perform hypercube quicksort following the pseudocode:

```
{Hypercube Quicksort}
bitvalue := $2^{dimension-1}$;
mask := $2^{dimension} - 1$;
for L := dimension downto 1
begin
  if myid AND mask = 0 then
    choose a pivot value for the L-dimensional subcube;
  broadcast the pivot from the master to the other members of the subcube;

  partition list[0:$n_{element}$-1] into two sublists such that
  list[0:j] ≤ pivot < list[j+1:$n_{element}$-1];

  partner := myid XOR bitvalue;
  if myid AND bitvalue = 0 then
    begin
      send the right sublist list[j+1:$n_{element}$-1] to partner;
      receive the left sublist of partner;
      append the received list to my left list
    end
  else
    begin
      send the left sublist list[0:j] to partner;
      receive the right sublist of partner;
      append the received list to my right list
    end
  $n_{element}$ := $n_{element}$ - $n_{send}$ + $n_{receive}$;
  mask = mask XOR bitvalue;
  bitvalue = bitvalue/2
end

sequential quicksort to list[0:$n_{element}$-1]
```

Notes

1. Write a program that works for any hypercube dimension.
2. Initially, each process randomly generates `n` (let it be 4) local elements in the range [0, `MAX` = 99).
   ```
   srand((unsigned) myid+1);
   for (i=0; i<n; i++) list[i] = rand()%MAX;
   ```
3. Print out the initial and final elements that each process has.
4. Before sending a sublist, check how many elements must be sent to the partner (it could be zero); send that information to the partner, and the partner issues a receive call for that number; issue a send or receive call for sublists only if the corresponding size is nonzero.
5. Implement a broadcast operation within an *L*-dimensional subcube. A hierarchy of subcubes can be implemented by nested calls to `MPI_Comm_create()` (see the lecture note on Message Passing Interface to learn about MPI communicators).

Submission

Submit the source code as well as a printout of the output from an 8-processor run.