Project 1: Variant of Leader Election on a Ring Topology

Group Members:
Andrew Lopez (alopez8969@csu.fullerton.edu)
Allen Sarmiento (allensarmiento@csu.fullerton.edu)
Ryan Shim (rcshim@csu.fullerton.edu)

Description

This ring communication example will send two values through several processes until they get back to the first process of rank 0. Each process generates a random value that can either be even or odd. The program keeps track of "president" and "vice" values which are the largest even value and the largest odd value respectively. The program compares both values at each rank and swaps one if it is greater than the current president or vice. The last two values received back to process 0 will be the greatest odd and even values from all of the processes.

Design Implementation

Process of rank 0 generates a random number from 10-99. Two numbers are added before it to represent the rank.

Example: Process of rank 5 generates a random number of 25.
Value assigned to rank 5 is 1525.

As the even and odd values get passed on to each process, the value being compared are the last two digits. So, we consider 1151 > 1531 because 51 > 31.

Example: Process of rank 3 receives 1181 and 1214.
Process of rank 3 generates a random number of 1321.
Process of rank 3 sends 1181 and 1214 to next process.

How to Execute:

Compile: mpicc leader_election.c
Execute: mpirun -np <N> ./a.out

- The value of N will be between 6 and 20.

Pseudocode

```
/* This program sends the maximum odd/even value in a ring-like fashion.
 * Each rank receives two values and depending on whether or not the RNG
 * token is odd or even, the process sends it's values to the next process.
```

```
 * This repeats for every non-zero rank until we get back to rank 0.
 * Constraints: The value must be from 10-100.
 *              The number of ranks will be from 6-20.
 */

// Initialization
Create process with N ranks
number = new Random Number
number = '1' + rank + number

// Passing values between ranks
if rank != 0
  receive values
  print values

  create random token value for each rank

  print token value

  if is max even:
      set value as president

  if is max odd:
      set value as vice

else
  create random token value for each rank
  if is even:
      set value as president

  if is odd:
      set value as vice
  print  process 0 token value

send president and vice to next process
// REPEAT FOR OTHER RANKS
```

```
//final step
if rank == 0
  receive president and vice values
  print president and vice values
  if is max even:
      set value as president

  if is max odd:
      set value as vice
  print president and vice values
  print president/vice processes
```