Project 1 Representative Selection Pseudocode

Alex Vidal

```
//Find representative by means of passing generated numbers amongst
//neighbors in ring topology. If a process detects it has the smallest value of its
//group it will be considered the representative of the group.
int main(void){
      int rank, size, lowerRank, upperRank, randN, genN, lowerGen, upperGen
      string buffer

      //Start communications and find ranks
      MPI_Init()
      rank = MPI_Comm_Rank()
      size = MPI_Comm_Size()
      upperRank = (rank+1)%size
      if(rank == 0) {
            lowerRank = size-1 }
      else {
            lowerRank = rank – 1 }

      //Generate unique number
      seed(time())
      randN = rand() % 100
      if(randN < 10){ randN += 10 }

      //Generated number is a 1 followed by the random number
      //as 2 digits, and the rank as 2 digits. Rank ensure uniqueness
      sprintf(buffer, “1%02d%02d”, randN, rank)
      sscanf(buffer, “%d”, &genN)

      //Communicate generated numbers with neighbors. All communications are
      //non-blocking to simplify code. Synchronize with a wait
      MPI_Irecv(lowerGen, lowerRank)
      MPI_Irecv(upperGen, upperRank)
      MPI_Isend(genN, lowerRank)
      MPI_Isend(genN, upperRank)
      MPI_Wait()

      //print details and if representative
      printf(“Rank %d, rand %d, gen %d, lower %d, upper %d. ”,
            rank, randN, genN, lowerGen, upperGen)
      if(genN < lowerGen && genN < upperGen){
            printf(“I am representative.”)
      }
      printf(“\n”);

      //close comms
      MPI_Finalize()
}
```