Ian Schweer

22514022

Part B:

Assuming the palindrome function is already there, this will be my guideline to implement part B.

Int main() {

…… // code to initialize palindromes from file (TA provided)

MPI\_Init(argc, argv) // Initalize openMPI

MPI\_Comm\_size(MPI\_COMM\_WORLD, &size) // get the size of the world

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &id) // get my position in the world

PrepareBuffers() // prepare the buffer data with the palindromes to search through

MPI\_Scatter(data, datasize, MPI\_CHAR, recvbuf, recvsize, MPI\_CHAR, id, comm);

FindPalindrome() // Find my palindrom

MPI\_Gather(data, datasize, MPI\_CHAR, recvbuf, recvsize, MPI\_CHAR, id, comm);

MPI\_Barrier // ensure everything gets here before stopping

Find longest of long palindroms

MPI\_Exit() // close MPI

}

Part C:

Assuming I’m called with palindrome data

MyGather() {

getLongestPalindrome()

compare mydata with previous node (me – 1)

send result to neighbor

}

Then at the root

MyGather\_Root() {

getLongestPalindrome()

sendToNextNode()

buffer r

for each node n:

r = MPI\_recv on node n > |r| // take mpi\_recv if greater then r

findLongest() // longest should be first element

}