Homework Assignment 7 – due on Saturday, November 16(Midnight)

Description of Assignment:
Complete an MPI program(psort.c) that sorts an array S[160]. The array S is initialized by random numbers in parallel.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "mpi.h"

#define N 160

#define a_const 1103515245
#define c_const 12345
#define mod 999

void srand2();
int rand2();
int prand_init();
void sort();
void merge();
int chk_square();

int main(int argc, char* argv[])
{
    int S[N], np, pid, local_N, n, eor_bits, partner, half,
i, tag = 0;
    int a = a_const, c = c_const, A, C, seed = 1;
    MPI_Status status;

    MPI_Init(&argc, &argv);

    MPI_Comm_size(MPI_COMM_WORLD, &np);
    MPI_Comm_rank(MPI_COMM_WORLD, &pid);

    // check np is a square number
    if (chk_square(np) != 1) {
        if (pid == 0) fprintf(stderr, "#proc is not a square
number\n");
        MPI_Finalize();
        exit(0);
    }

    local_N = N/np;

    srand2(seed);
    S[0] = prand_init(…) % mod; // first random
numbers
    for (i=1; i<local_N; i++)
        S[i] = rand2(A, C) % mod; // parallel random numbers

    sort(…); // local sort

    // logN gather algorithm
    half = np/2;
    n = local_N;
    for (eor_bits = half; eor_bits > 0; eor_bits = eor_bits>>1) {
        partner = pid^eor_bits;
        if (pid >= eor_bits)
            MPI_Send(…);
        else {
            MPI_Recv(…);
            merge(…);
        }
        n = …;
    }

    if (pid == 0) {
        for (i=0; i<N; i++)
            printf("%4d ", S[i]);
        printf("\n");
    }

    // the sequential result
    if (pid == 0) {
        a = a_const; c = c_const;
        srand2(seed);
        for (i=0; i<N; i++)
            S[i] = rand2(a, c) % mod;

        sort(N, S);

        printf("sequentially sorted array ----------\n");
        for (i=0; i<N; i++)
            printf("%4d ", S[i]);
        printf("\n");
    }

    MPI_Finalize();
    exit(0);
}
```

How to proceed:
(i) Copy /home/course/lib_sort.c into your working directory.
(ii) Compile “mpicc -o psort psort.c lib_sort.c”.
(iii) Run only a square number of processors(2,4, 8, 16, …) for tests.

Turnin the assignment:
After done your assignment, type **turnin** in your current working directory. You can retype the command at any time before the due date.