Lab4 – MPI(p2p)

Complete MPI_Send() and MPI_Recv() in the following MPI program(merge.c) to merge sub-arrays to process P_0 from all other processes(P_1, P_2, …, P_{n-1}). Assume that the size of A is divisible by the number of processes.

- **#processes = 2 → 0 1 2 3 4 5 10 11 12 13 14 15**
- **#processes = 3 → 0 1 2 3 10 11 12 13 20 21 22 23**
- **#processes = 4 → 0 1 2 10 11 12 20 21 22 30 31 32**

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

#define N 12

main(int argc, char* argv[])
{
    int np, pid, i, dest, source, tag = 0;
    int A[N], *local_A, local_N;
    MPI_Status status;

    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &np);
    MPI_Comm_rank(MPI_COMM_WORLD, &pid);

    local_N = N/np;
    local_A = (int*)malloc(sizeof(int)*local_N);

    // initialize sub-arrays on all processes
    for (i = 0; i < local_N; i++)
        local_A[i] = pid*10+i;

    // merge
    if (pid != 0)
        MPI_Send(...);
    else {
        for (...) // copy local_A to A on P0
            A[i] = local_A[i];
        for (i= 1; i < np; i++)
            MPI_Recv(...);
    }

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

    MPI_Finalize();
}
```

Submit merge.c when you complete programming.