CSE4001 - Parallel & Distributed computing.

Slot : L29 + L30.

DA - 4.

Name : Ashit Bose Tagore.
Reg. No : 20BCE2150

Qs. Write a MPI program to print the communication world parameters - data.

Soln:

```
// MPI program - prints comm world params.
# include <stdio.h>
# include <mpi.h>
int main(int argc, char *argv[]){
        int rank;
        int size;
        char processor_name[MPI_MAX_PROCESSOR_NAME];
        int name_len;
       MPI_INIT(&argc, &argv);
       MPI_Comm_rank(MPI_COMM_WORLD, &rank);
       MPI_Comm_size(MPI_COMM_WORLD, &size);
       MPI_Get_processor_name(processor_name, &name_len);
     MPI_Finalize();
     return 0;
}
```

Q2. Write a MPI Program to perform a point-to-point communication by sending a date

Sol^n:

```
#include <stdio.h>
#include <mpi.h>
int main(int argc, char *argv[]){
    int rank, size;
    int send_data, received_data;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    if(size < 2){
        fprintf(stderr, "This program requires atleast
            2 processes.\n");
        MPI_Abort(MPI_COMM_WORLD, 1);
    }
    if(rank == 0){
        send_data = 42;
        MPI_Send(&send_data, 1, MPI_INT, 1, 0,
                 MPI_COMM_WORLD);
        printf("Process 0 sent data %d\n", send_data);
    } else if(rank == 1){
        MPI_Recv(&received_data, 1, MPI_INT, 0, 0,
                 MPI_COMM_WORLD, MPI_STATUS_IGNORE);
        printf("Process 1 received data: %d\n", received
                                                 data);
    }
    MPI_Finalize();
    return 0;
}
```