Name: Abhijeet Kamalekar
PRN: 2020BTECS00010

High Performance Computing Lab

Practical No. 7

Title of practical:

Installation of MPI & Implementation of basic functions of MPI

sudo apt install openmpi-bin libopenmpi-dev

Problem Statement 1:

Implement a simple hello world program by setting number of processes equal to 10

```c
#include <stdio.h>

#include <mpi.h>

int main(int argc, char **argv) {
// Initialize the MPI environment
MPI_Init(&argc, &argv);
// Get the number of processes
int world_size;
MPI_Comm_size(MPI_COMM_WORLD, &world_size);
// Get the rank of the process
int world_rank;
MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);
// Print Hello World message from each process
printf("Hello, World! I am process %d of %d.\n", world_rank, world_size);
// Finalize the MPI environment
MPI_Finalize();
return 0;
}
```

Screenshots:

```
abhijeet@abhijeet-VirtualBox:~/MPI$ mpirun -np 4 ./7_1
Hello, World! I am process 1 of 4.
Hello, World! I am process 2 of 4.
Hello, World! I am process 0 of 4.
Hello, World! I am process 3 of 4.
```

Problem Statement 2:

Implement a program to display rank and communicator group of five processes

```
#include <stdio.h>

#include <mpi.h>

int main(int argc, char **argv) {
// Initialize the MPI environment
MPI_Init(&argc, &argv);
// Get the rank of the process
int world_rank;
MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);
// Get the number of processes in the communicator
int world_size;
MPI_Comm_size(MPI_COMM_WORLD, &world_size);
// Display rank and size of the communicator
printf("Process rank: %d, Communicator size: %d\n", world_rank, world_size);
// Finalize the MPI environment
MPI_Finalize();
return 0;
}
```

Screenshots: