**High Performance Computing Lab**

**Practical No. 7**

**Name : Manish Namdev Barage  
PRN – 22520007(B6)**

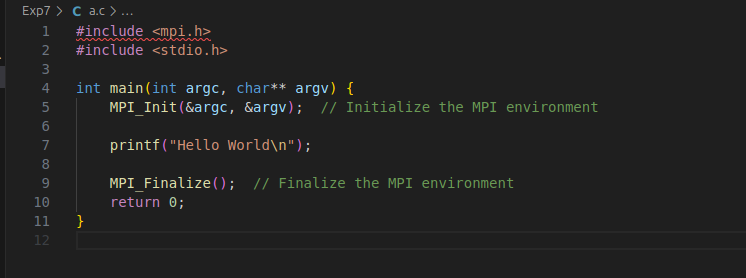
**Title of practical:**

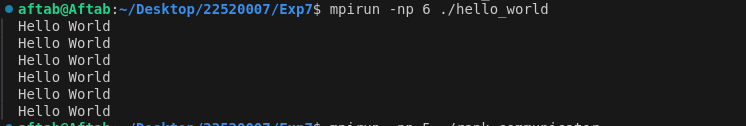
Installation of MPI & Implementation of basic functions of MPI

**Problem Statement 1:**

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

**Screenshots:**

 **Output :**

****

**Information 1:**

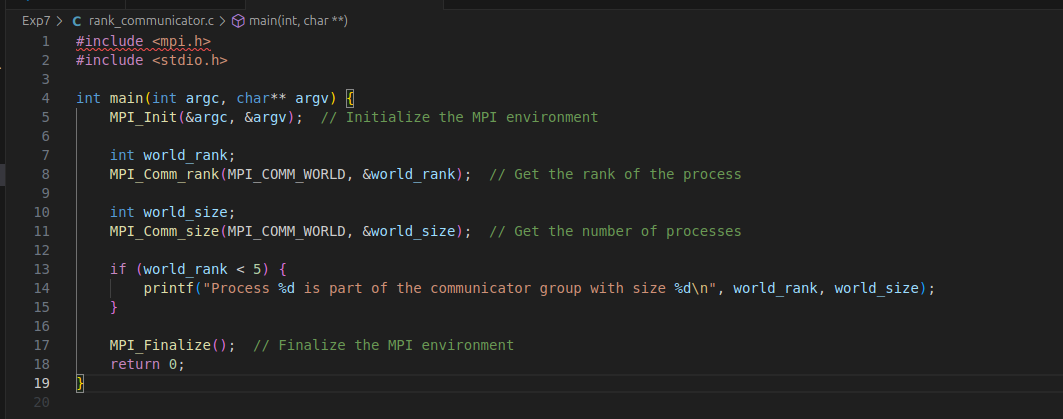
**MPI Overview:**

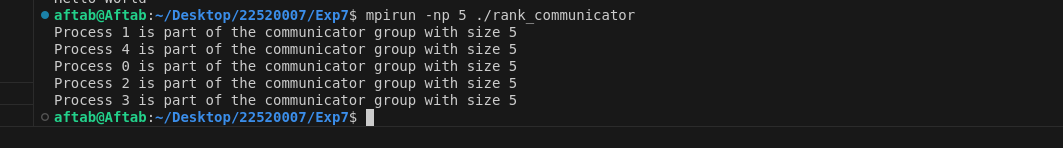
* + **MPI (Message Passing Interface)** is a library used to implement parallelism in programs by enabling communication between processes. It is widely used in distributed memory systems to perform computations in parallel.

1. mpirun -np 6 ./hello\_world:
   * The mpirun command is used to launch the program in parallel using multiple processes.
   * -np 6: This option tells mpirun to start **10 processes**. Each process will execute the same program independently.
   * ./hello\_world is the executable that will run on all 10 processes.
2. **MPI Functions Used**:
   * MPI\_Init(&argc, &argv): Initializes the MPI environment. This function must be called at the start of the program.
   * MPI\_Comm\_size(MPI\_COMM\_WORLD, &world\_size): Retrieves the total number of processes in the communicator MPI\_COMM\_WORLD. In this case, world\_size will be 10.
   * MPI\_Comm\_rank(MPI\_COMM\_WORLD, &world\_rank): Retrieves the rank (unique identifier) of each process in the communicator. The rank will be an integer between 0 and 6.
   * MPI\_Finalize(): Cleans up the MPI environment. This function is called at the end of the program to free resources.
3. **Program Output**:
   * Each process will print a "Hello World" message along with the total number of processes. Since there are 6 processes, you will see this message 6 times, one from each process.

**Problem Statement 2:**

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



****

**Information:  
Communicator Group:**

* A **communicator group** in MPI is a collection of processes that can communicate with one another. The default communicator in MPI is MPI\_COMM\_WORLD, which includes all processes in the program.
* Each process in a communicator has a unique **rank**, which is used to identify it during communication.
* mpirun -np 5 ./rank\_comm\_group:
* mpirun starts the program with **5 processes**.
* -np 5 specifies that the program should be executed with 5 processes.
* ./rank\_comm\_group is the executable, which runs on each of the 5 processes.
* **MPI Functions Used**:
* MPI\_Comm\_rank(MPI\_COMM\_WORLD, &world\_rank): This function determines the **rank** of each process within the communicator MPI\_COMM\_WORLD. Each process gets a unique rank between 0 and 4.
* MPI\_Comm\_size(MPI\_COMM\_WORLD, &world\_size): This function determines the total number of processes in MPI\_COMM\_WORLD. In this case, world\_size will be 5, as there are 5 processes in the communicator.
* **Communicator Information**:
* MPI\_COMM\_WORLD is the default communicator that includes all processes running in the program. Each process can send and receive messages to/from any other process within this communicator.
* **Program Output**:
* Each of the 5 processes will print its **rank** and information about the **communicator group** (MPI\_COMM\_WORLD). For instance, process 0 will print:  
  "Process 0 out of 5 in MPI\_COMM\_WORLD"
* This will confirm that all 5 processes are part of the same communicator.