**MPI Programming Tasks**

This repository contains implementations of fundamental MPI (Message Passing Interface) programs, covering essential communication techniques used in parallel computing.

**📌 Tasks Overview**

**1️⃣ Hello World Program**

This program demonstrates the basic setup of an MPI environment. Each process prints a message indicating its rank and the total number of processes.

**Key Functions Used:**

* MPI\_Init - Initializes the MPI environment.
* MPI\_Comm\_rank - Retrieves the rank of the process.
* MPI\_Comm\_size - Gets the total number of processes.
* MPI\_Finalize - Terminates the MPI execution environment.

**2️⃣ Sending and Receiving with MPI\_Send and MPI\_Recv**

This program demonstrates point-to-point communication, where one process sends a message and another receives it.

**Key Functions Used:**

* MPI\_Send - Sends a message from one process to another.
* MPI\_Recv - Receives a message sent by another process.

**3️⃣ Dynamic Receiving with MPI\_Probe and MPI\_Status**

This program introduces dynamic message receiving, allowing a process to check for incoming messages before receiving them. This helps in handling messages of varying sizes.

**Key Functions Used:**

* MPI\_Probe - Checks for incoming messages without receiving them.
* MPI\_Get\_count - Determines the size of the received message.
* MPI\_Status - Holds information about the received message.

**4️⃣ Point-to-Point Communication Application - Random Walking**

This program simulates a random walk using MPI processes. Each process performs a step in the random walk and communicates with others to track movement.

**Key Concepts Used:**

* MPI\_Send and MPI\_Recv for message passing.
* Random number generation for walk direction.
* Synchronization among processes to ensure accurate simulation.

**🔧 Compilation and Execution**

**🔹 Compilation**

To compile any of the programs, use:

mpicc program\_name.c -o program\_name

**🔹 Running the Programs**

To execute with multiple processes:

mpirun -np <num\_processes> ./program\_name

Example:

mpirun -np 4 ./hello\_world

**📜 Requirements**

* MPI Library (OpenMPI or MPICH)
* GCC Compiler
* Linux/macOS environment (or WSL for Windows users)

**💡 Future Enhancements**

* Extend point-to-point communication for more complex applications.
* Implement collective communication techniques such as MPI\_Bcast and MPI\_Reduce.
* Optimize random walk simulation with load balancing strategies.

**🤝 Contributing**

Feel free to submit pull requests for improvements or additional examples!

**📄 License**

This project is open-source and available under the MIT License.