Nagar Yuwak Shikshan Sanstha’s

Yeshwantrao Chavan College of Engineering

(An Autonomous Institution affiliated to Rashtrasant Tukadoji Maharaj Nagpur University)
Hingna Road, Wanadongri, Nagpur - 441 110
NAAC A++
Ph.: 07104-237919, 234623, 329249, 329250 Fax: 07104-232376, Website: www.ycce.edu

Department of Computer Technology

Vision of the Department

To be a well-known centre for pursuing computer education through innovative pedagogy, value-based education and industry collaboration.

Mission of the Department

To establish learning ambience for ushering in computer engineering professionals in core and multidisciplinary area by developing Problem-solving skills through emerging technologies.

Session 2025-2026

Vision: To harness the power of artificial intelligence and data science to solve real-world problems and enhance human potential.	**Mission:** To acquire skills through coursework, projects, and internships, while actively engaging in research and collaboration with peers to innovate and apply AI solutions.

Program Educational Objectives of the program (PEO): (broad statements that describe the professional and career accomplishments)

PEO1	**Preparation**	**P: Preparation**	**Pep-CL abbreviation pronounce as Pep-si-lL easy to recall**
PEO2	**Core Competence**	**E: Environment (Learning Environment)**	
PEO3	**Breadth**	**P: Professionalism**	
PEO4	**Professionalism**	**C: Core Competence**	
PEO5	**Learning Environment**	**L: Breadth (Learning in diverse areas)**	

Program Outcomes (PO): (statements that describe what a student should be able to do and know by the end of a program)

Keywords of POs:
Engineering knowledge, Problem analysis, Design/development of solutions, Conduct Investigations of Complex Problems, Engineering Tool Usage, The Engineer and The World, Ethics, Individual and Collaborative Team work, Communication, Project Management and Finance, Life-Long Learning

PSO Keywords: Cutting edge technologies, Research

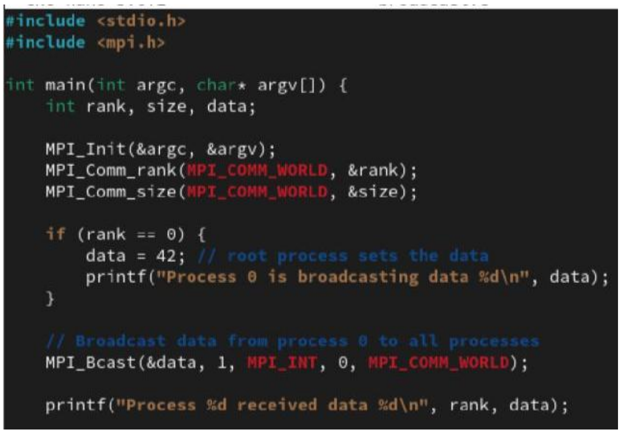
“I am an engineer, and I know how to apply engineering knowledge to investigate, analyse and design solutions to complex problems using tools for entire world following all ethics in a collaborative way with proper management skills throughout my life.” *to contribute to the development of cutting-edge technologies and Research*.

Integrity: I will adhere to the Laboratory Code of Conduct and ethics in its entirety.

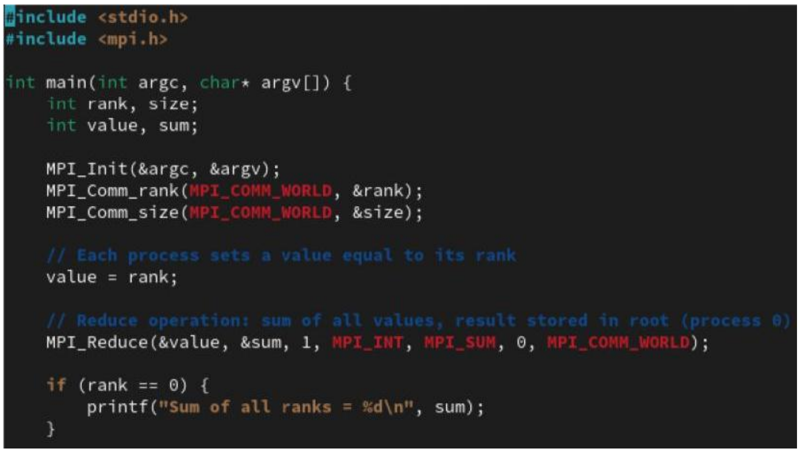
Prerana Bijekar 28 October 2025
Name and Signature of Student and Date
(Signature and Date in Handwritten)

Session	2025-26 (ODD)	**Course Name**	HPC Lab
Semester	7	**Course Code**	22ADS706
Roll No	11	**Name of Student**	Prerana Bijekar

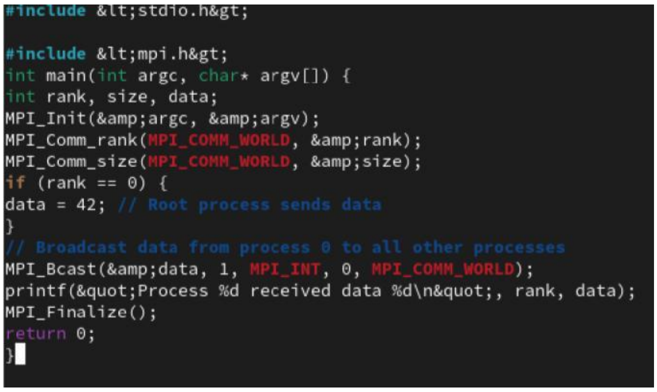
Practical Number	5
Course Outcome	**CO1:** Understand and Apply Parallel Programming Concepts **CO2:** Analyze and Improve Program Performance. **CO3:** Demonstrate Practical Skills in HPC Tools and Environments.
Aim	Basics of MPI Programming
Theory (100 words)	Measuring program performance involves quantifying a program's efficiency and effectiveness. The main goal is to understand how well a program utilizes resources and if it's achieving its intended goals. Key metrics include execution time (wall-clock time), CPU utilization, and memory usage. The theory is that by systematically collecting and analyzing this data, you can identify bottlenecks, optimize code, and make informed decisions to improve a program's overall performance. This isn't just about making things faster; it's about ensuring your program is a good neighbor in a shared computing environment by not wasting resources.
Procedure and Execution (100 Words)	Steps of Implementation:- • Install MPI library (e.g., OpenMPI or MPICH) on Linux. • Write MPI program in C/C++ using functions like MPI_Init, MPI_Comm_rank, and MPI_Send/Recv. • Compile using mpicc program.c -o program. • Run with multiple processes: mpirun -np 4 ./program. • Observe outputs from each process (rank IDs)
	Algorithm: • sudo apt-get install openmpi-bin openmpi-common libopenmpi-dev brew • install open-mpi • mpicc hello.c -o hello mpirun -np 4 ./hello • mpirun -np 4 ./program_name
	Code:



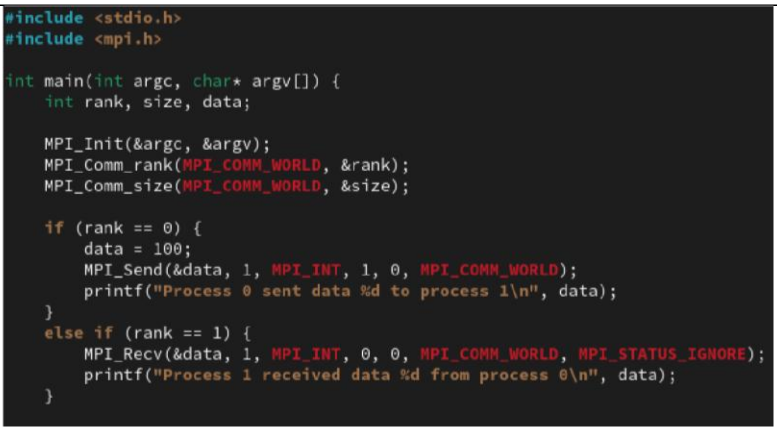
```
#include <stdio.h>
#include <mpi.h>

int main(int argc, char* argv[]) {
    int rank, size, data;

    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    if (rank == 0) {
        data = 42; // root process sets the data
        printf("Process 0 is broadcasting data %d\n", data);
    }

    // Broadcast data from process 0 to all processes
    MPI_Bcast(&data, 1, MPI_INT, 0, MPI_COMM_WORLD);

    printf("Process %d received data %d\n", rank, data);
```



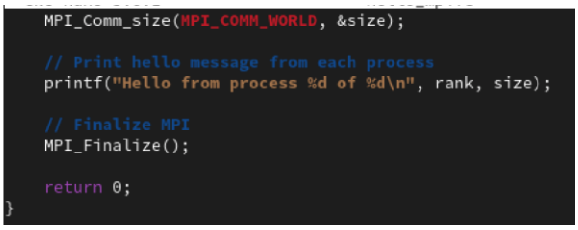
```
#include <stdio.h>
#include <mpi.h>

int main(int argc, char* argv[]) {
    int rank, size;
    int value, sum;

    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    // Each process sets a value equal to its rank
    value = rank;

    // Reduce operation: sum of all values, result stored in root (process 0)
    MPI_Reduce(&value, &sum, 1, MPI_INT, MPI_SUM, 0, MPI_COMM_WORLD);

    if (rank == 0) {
        printf("Sum of all ranks = %d\n", sum);
    }
```



```
#include &lt;stdio.h&gt;

#include &lt;mpi.h&gt;
int main(int argc, char* argv[]) {
int rank, size, data;
MPI_Init(&amp;argc, &amp;argv);
MPI_Comm_rank(MPI_COMM_WORLD, &amp;rank);
MPI_Comm_size(MPI_COMM_WORLD, &amp;size);
if (rank == 0) {
data = 42; // Root process sends data
}
// Broadcast data from process 0 to all other processes
MPI_Bcast(&amp;data, 1, MPI_INT, 0, MPI_COMM_WORLD);
printf(&quot;Process %d received data %d\n&quot;, rank, data);
MPI_Finalize();
return 0;
}
```

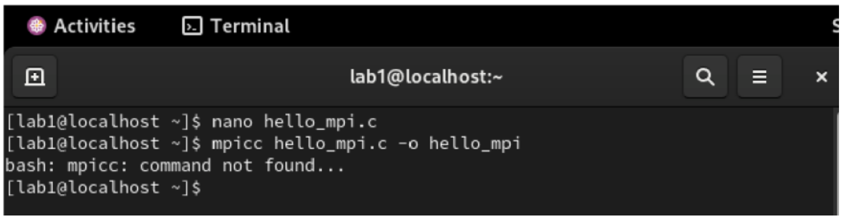


```c
#include <stdio.h>
#include <mpi.h>

int main(int argc, char* argv[]) {
    int rank, size, data;

    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    if (rank == 0) {
        data = 100;
        MPI_Send(&data, 1, MPI_INT, 1, 0, MPI_COMM_WORLD);
        printf("Process 0 sent data %d to process 1\n", data);
    }
    else if (rank == 1) {
        MPI_Recv(&data, 1, MPI_INT, 0, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
        printf("Process 1 received data %d from process 0\n", data);
    }
```

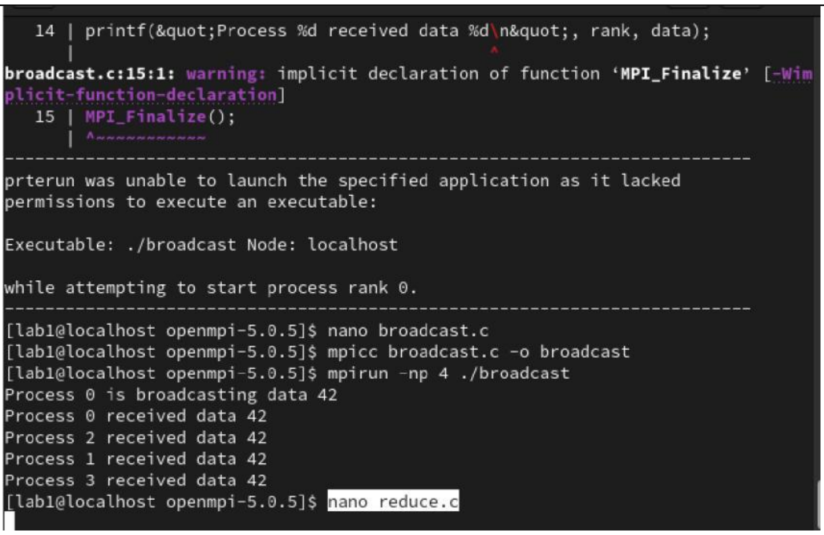


```c
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    // Print hello message from each process
    printf("Hello from process %d of %d\n", rank, size);

    // Finalize MPI
    MPI_Finalize();

    return 0;
}
```

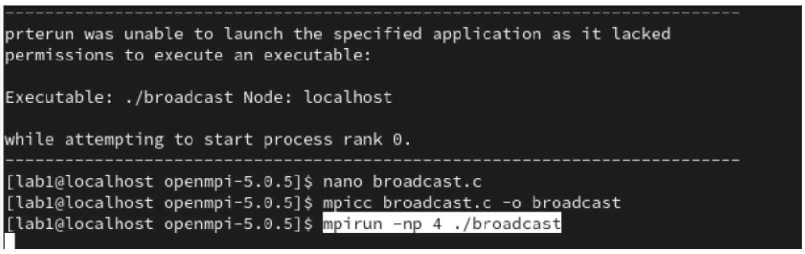
Output:



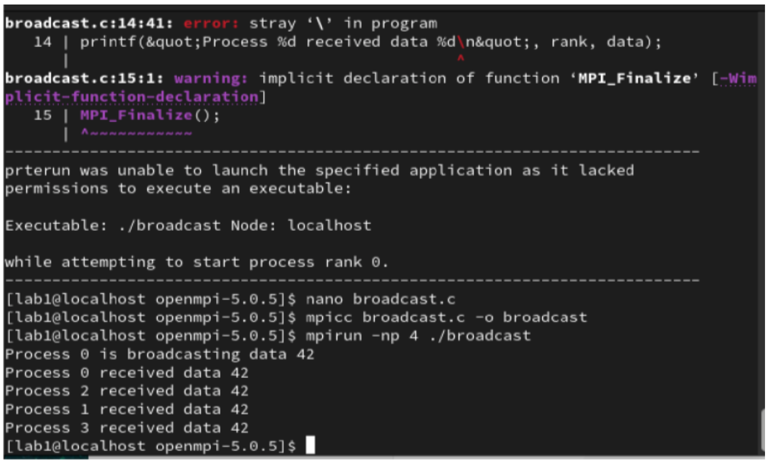
```
[lab1@localhost ~]$ nano hello_mpi.c
[lab1@localhost ~]$ mpicc hello_mpi.c -o hello_mpi
bash: mpicc: command not found...
[lab1@localhost ~]$
```



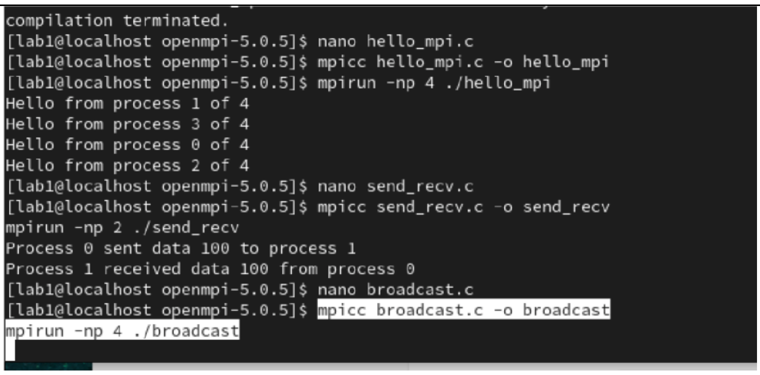
```
   14 | printf(&quot;Process %d received data %d\n&quot;, rank, data);
      |                                         ^
broadcast.c:15:1: warning: implicit declaration of function ‘MPI_Finalize’ [-Wim
plicit-function-declaration]
   15 | MPI_Finalize();
      | ^~~~~~~~~~~~
--------------------------------------------------------------------------
prterun was unable to launch the specified application as it lacked
permissions to execute an executable:

Executable: ./broadcast Node: localhost

while attempting to start process rank 0.
--------------------------------------------------------------------------
[lab1@localhost openmpi-5.0.5]$ nano broadcast.c
[lab1@localhost openmpi-5.0.5]$ mpicc broadcast.c -o broadcast
[lab1@localhost openmpi-5.0.5]$ mpirun -np 4 ./broadcast
Process 0 is broadcasting data 42
Process 0 received data 42
Process 2 received data 42
Process 1 received data 42
Process 3 received data 42
[lab1@localhost openmpi-5.0.5]$ nano reduce.c
```



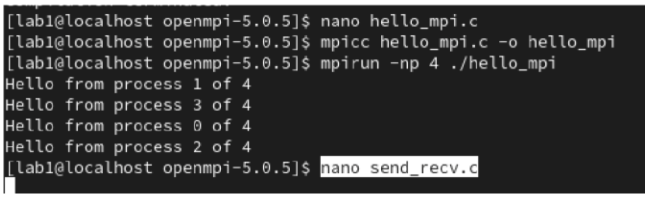
```
--------------------------------------------------------------------------
prterun was unable to launch the specified application as it lacked
permissions to execute an executable:

Executable: ./broadcast Node: localhost

while attempting to start process rank 0.
--------------------------------------------------------------------------
[lab1@localhost openmpi-5.0.5]$ nano broadcast.c
[lab1@localhost openmpi-5.0.5]$ mpicc broadcast.c -o broadcast
[lab1@localhost openmpi-5.0.5]$ mpirun -np 4 ./broadcast
```



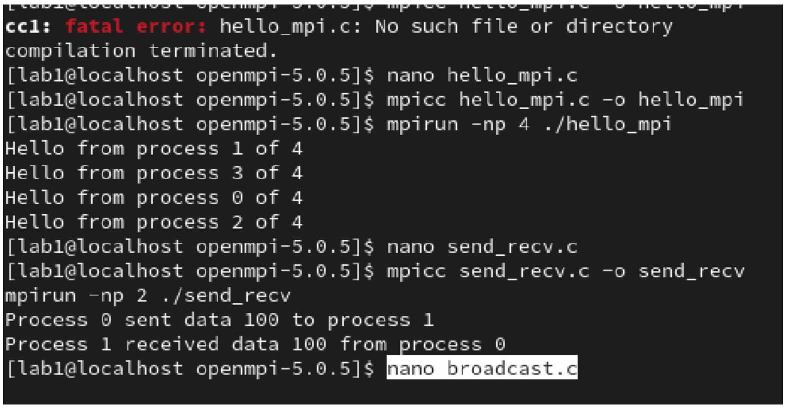
```
broadcast.c:14:41: error: stray ‘\’ in program
   14 | printf(&quot;Process %d received data %d\n&quot;, rank, data);
      |                                         ^
broadcast.c:15:1: warning: implicit declaration of function ‘MPI_Finalize’ [-Wim
plicit-function-declaration]
   15 | MPI_Finalize();
      | ^~~~~~~~~~~~
--------------------------------------------------------------------------
prterun was unable to launch the specified application as it lacked
permissions to execute an executable:

Executable: ./broadcast Node: localhost

while attempting to start process rank 0.
--------------------------------------------------------------------------
[lab1@localhost openmpi-5.0.5]$ nano broadcast.c
[lab1@localhost openmpi-5.0.5]$ mpicc broadcast.c -o broadcast
[lab1@localhost openmpi-5.0.5]$ mpirun -np 4 ./broadcast
Process 0 is broadcasting data 42
Process 0 received data 42
Process 2 received data 42
Process 1 received data 42
Process 3 received data 42
[lab1@localhost openmpi-5.0.5]$
```



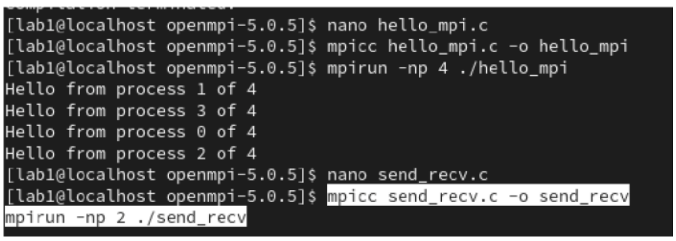
```
compilation terminated.
[lab1@localhost openmpi-5.0.5]$ nano hello_mpi.c
[lab1@localhost openmpi-5.0.5]$ mpicc hello_mpi.c -o hello_mpi
[lab1@localhost openmpi-5.0.5]$ mpirun -np 4 ./hello_mpi
Hello from process 1 of 4
Hello from process 3 of 4
Hello from process 0 of 4
Hello from process 2 of 4
[lab1@localhost openmpi-5.0.5]$ nano send_recv.c
[lab1@localhost openmpi-5.0.5]$ mpicc send_recv.c -o send_recv
mpirun -np 2 ./send_recv
Process 0 sent data 100 to process 1
Process 1 received data 100 from process 0
[lab1@localhost openmpi-5.0.5]$ nano broadcast.c
[lab1@localhost openmpi-5.0.5]$ mpicc broadcast.c -o broadcast
mpirun -np 4 ./broadcast
```



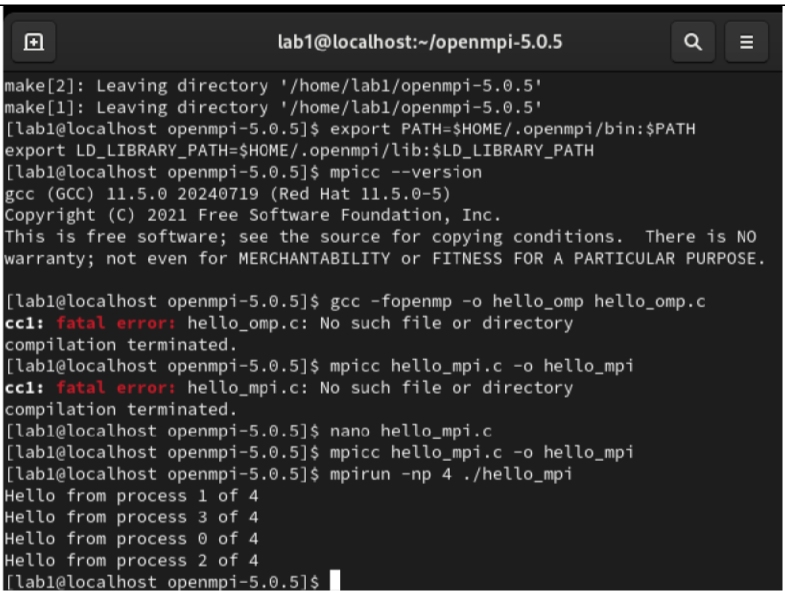
```
[lab1@localhost openmpi-5.0.5]$ nano hello_mpi.c
[lab1@localhost openmpi-5.0.5]$ mpicc hello_mpi.c -o hello_mpi
[lab1@localhost openmpi-5.0.5]$ mpirun -np 4 ./hello_mpi
Hello from process 1 of 4
Hello from process 3 of 4
Hello from process 0 of 4
Hello from process 2 of 4
[lab1@localhost openmpi-5.0.5]$ nano send_recv.c
```



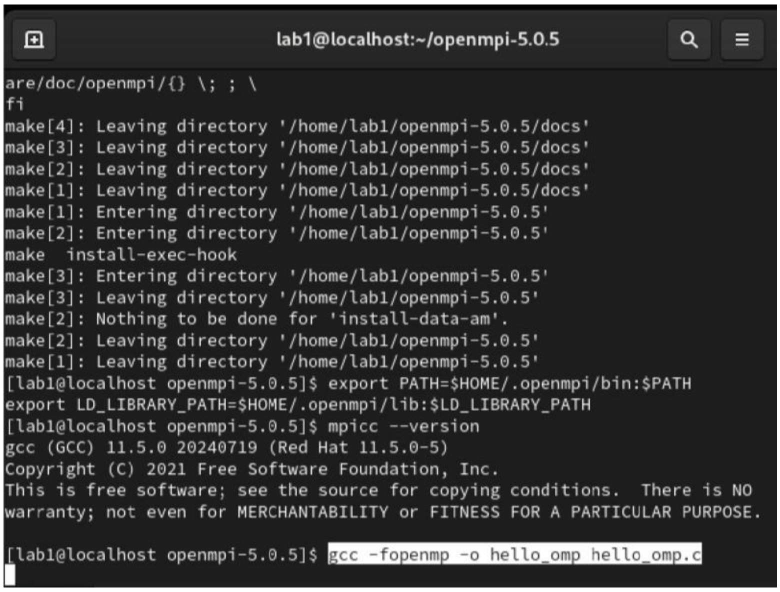
```
cc1: fatal error: hello_mpi.c: No such file or directory
compilation terminated.
[lab1@localhost openmpi-5.0.5]$ nano hello_mpi.c
[lab1@localhost openmpi-5.0.5]$ mpicc hello_mpi.c -o hello_mpi
[lab1@localhost openmpi-5.0.5]$ mpirun -np 4 ./hello_mpi
Hello from process 1 of 4
Hello from process 3 of 4
Hello from process 0 of 4
Hello from process 2 of 4
[lab1@localhost openmpi-5.0.5]$ nano send_recv.c
[lab1@localhost openmpi-5.0.5]$ mpicc send_recv.c -o send_recv
mpirun -np 2 ./send_recv
Process 0 sent data 100 to process 1
Process 1 received data 100 from process 0
[lab1@localhost openmpi-5.0.5]$ nano broadcast.c
```



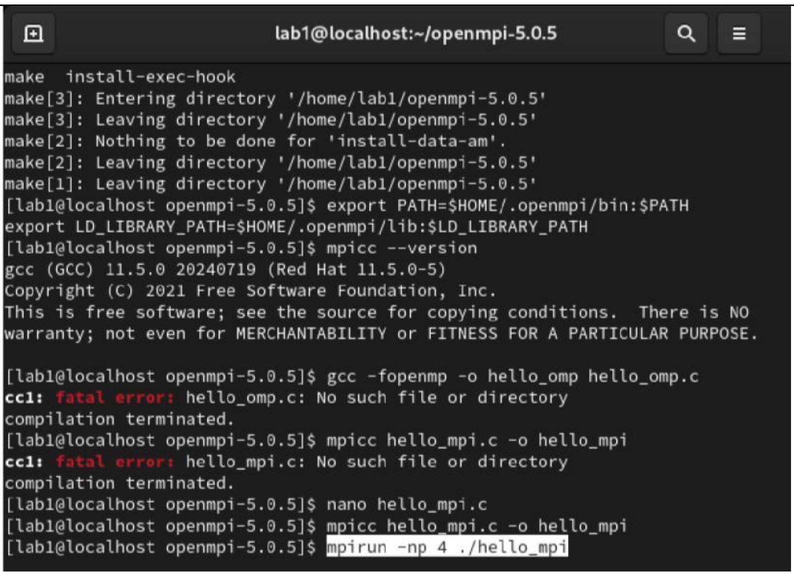
```
[lab1@localhost openmpi-5.0.5]$ nano hello_mpi.c
[lab1@localhost openmpi-5.0.5]$ mpicc hello_mpi.c -o hello_mpi
[lab1@localhost openmpi-5.0.5]$ mpirun -np 4 ./hello_mpi
Hello from process 1 of 4
Hello from process 3 of 4
Hello from process 0 of 4
Hello from process 2 of 4
[lab1@localhost openmpi-5.0.5]$ nano send_recv.c
[lab1@localhost openmpi-5.0.5]$ mpicc send_recv.c -o send_recv
mpirun -np 2 ./send_recv
```

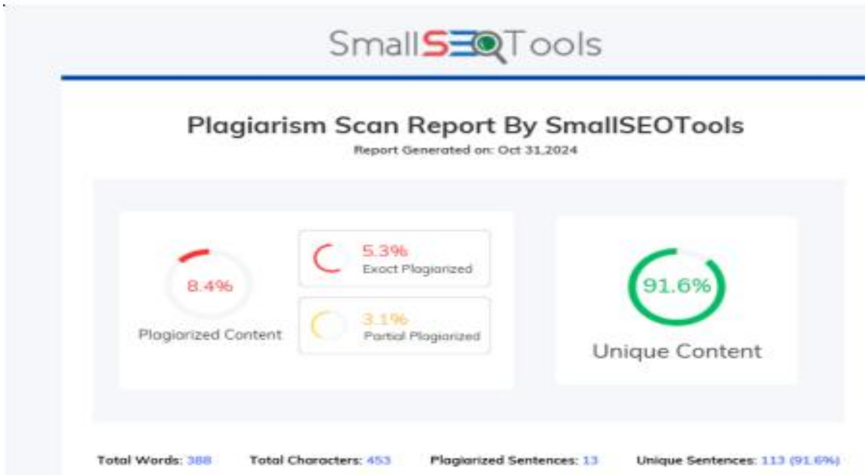


```
make[2]: Leaving directory '/home/lab1/openmpi-5.0.5'
make[1]: Leaving directory '/home/lab1/openmpi-5.0.5'
[lab1@localhost openmpi-5.0.5]$ export PATH=$HOME/.openmpi/bin:$PATH
export LD_LIBRARY_PATH=$HOME/.openmpi/lib:$LD_LIBRARY_PATH
[lab1@localhost openmpi-5.0.5]$ mpicc --version
gcc (GCC) 11.5.0 20240719 (Red Hat 11.5.0-5)
Copyright (C) 2021 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

[lab1@localhost openmpi-5.0.5]$ gcc -fopenmp -o hello_omp hello_omp.c
cc1: fatal error: hello_omp.c: No such file or directory
compilation terminated.
[lab1@localhost openmpi-5.0.5]$ mpicc hello_mpi.c -o hello_mpi
cc1: fatal error: hello_mpi.c: No such file or directory
compilation terminated.
[lab1@localhost openmpi-5.0.5]$ nano hello_mpi.c
[lab1@localhost openmpi-5.0.5]$ mpicc hello_mpi.c -o hello_mpi
[lab1@localhost openmpi-5.0.5]$ mpirun -np 4 ./hello_mpi
Hello from process 1 of 4
Hello from process 3 of 4
Hello from process 0 of 4
Hello from process 2 of 4
[lab1@localhost openmpi-5.0.5]$
```



```
are/doc/openmpi/{} \; ; \
fi
make[4]: Leaving directory '/home/lab1/openmpi-5.0.5/docs'
make[3]: Leaving directory '/home/lab1/openmpi-5.0.5/docs'
make[2]: Leaving directory '/home/lab1/openmpi-5.0.5/docs'
make[1]: Leaving directory '/home/lab1/openmpi-5.0.5/docs'
make[1]: Entering directory '/home/lab1/openmpi-5.0.5'
make[2]: Entering directory '/home/lab1/openmpi-5.0.5'
make  install-exec-hook
make[3]: Entering directory '/home/lab1/openmpi-5.0.5'
make[3]: Leaving directory '/home/lab1/openmpi-5.0.5'
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/home/lab1/openmpi-5.0.5'
make[1]: Leaving directory '/home/lab1/openmpi-5.0.5'
[lab1@localhost openmpi-5.0.5]$ export PATH=$HOME/.openmpi/bin:$PATH
export LD_LIBRARY_PATH=$HOME/.openmpi/lib:$LD_LIBRARY_PATH
[lab1@localhost openmpi-5.0.5]$ mpicc --version
gcc (GCC) 11.5.0 20240719 (Red Hat 11.5.0-5)
Copyright (C) 2021 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

[lab1@localhost openmpi-5.0.5]$ gcc -fopenmp -o hello_omp hello_omp.c
```

	
	lab1@localhost:~/openmpi-5.0.5
	`make  install-exec-hook`
	`make[3]: Entering directory '/home/lab1/openmpi-5.0.5'`
	`make[3]: Leaving directory '/home/lab1/openmpi-5.0.5'`
	`make[2]: Nothing to be done for 'install-data-am'.`
	`make[2]: Leaving directory '/home/lab1/openmpi-5.0.5'`
	`make[1]: Leaving directory '/home/lab1/openmpi-5.0.5'`
	`[lab1@localhost openmpi-5.0.5]$ export PATH=$HOME/.openmpi/bin:$PATH`
	`export LD_LIBRARY_PATH=$HOME/.openmpi/lib:$LD_LIBRARY_PATH`
	`[lab1@localhost openmpi-5.0.5]$ mpicc --version`
	`gcc (GCC) 11.5.0 20240719 (Red Hat 11.5.0-5)`
	`Copyright (C) 2021 Free Software Foundation, Inc.`
	`This is free software; see the source for copying conditions.  There is NO`
	`warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.`
	
	`[lab1@localhost openmpi-5.0.5]$ gcc -fopenmp -o hello_omp hello_omp.c`
	`cc1: fatal error: hello_omp.c: No such file or directory`
	`compilation terminated.`
	`[lab1@localhost openmpi-5.0.5]$ mpicc hello_mpi.c -o hello_mpi`
	`cc1: fatal error: hello_mpi.c: No such file or directory`
	`compilation terminated.`
	`[lab1@localhost openmpi-5.0.5]$ nano hello_mpi.c`
	`[lab1@localhost openmpi-5.0.5]$ mpicc hello_mpi.c -o hello_mpi`
	`[lab1@localhost openmpi-5.0.5]$ mpirun -np 4 ./hello_mpi`

Output Analysis	When executed with multiple processes, each process prints its rank and the total number of processes. Communication via MPI_Send and MPI_Recv demonstrates message passing between processes. The output confirms correct distribution of tasks across processes, proving parallel execution on Linux.
Github Link	https://github.com/Prerana-Bijekar/HPC
Conclusion	MPI allows efficient communication between processes for parallel programming. It provides scalability and portability for distributed computing on Linux systems.
Plag Report (Similarity index < 12%)	 SmallSEOTools Plagiarism Scan Report By SmallSEOTools Report Generated on: Oct 31,2024 8.4% Plagiarized Content 5.3% Exact Plagiarized 3.1% Partial Plagiarized 91.6% Unique Content Total Words: 388 Total Characters: 453 Plagiarized Sentences: 13 Unique Sentences: 113 (91.6%)
Date	28 October 2025