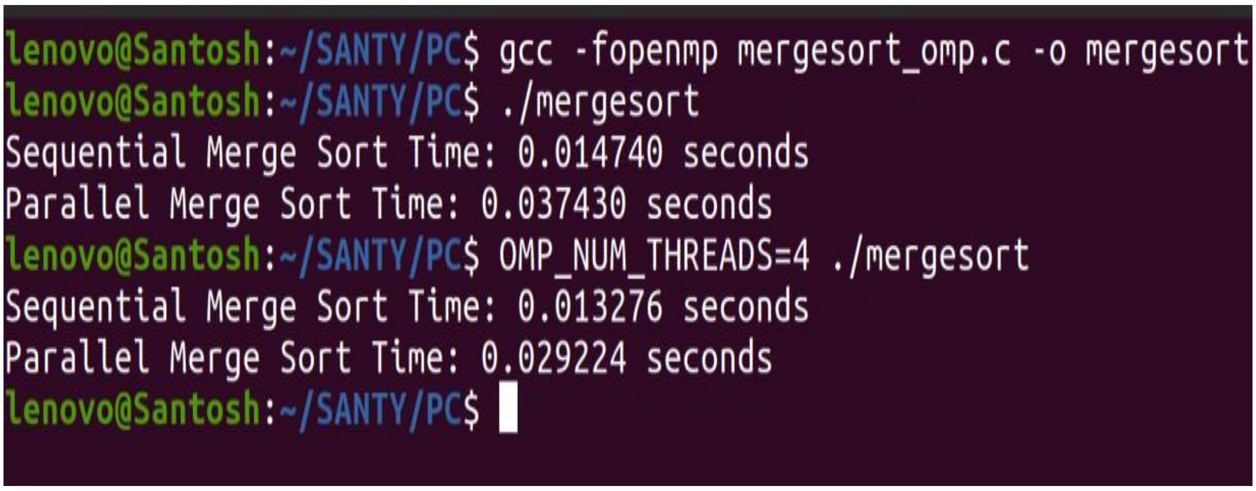
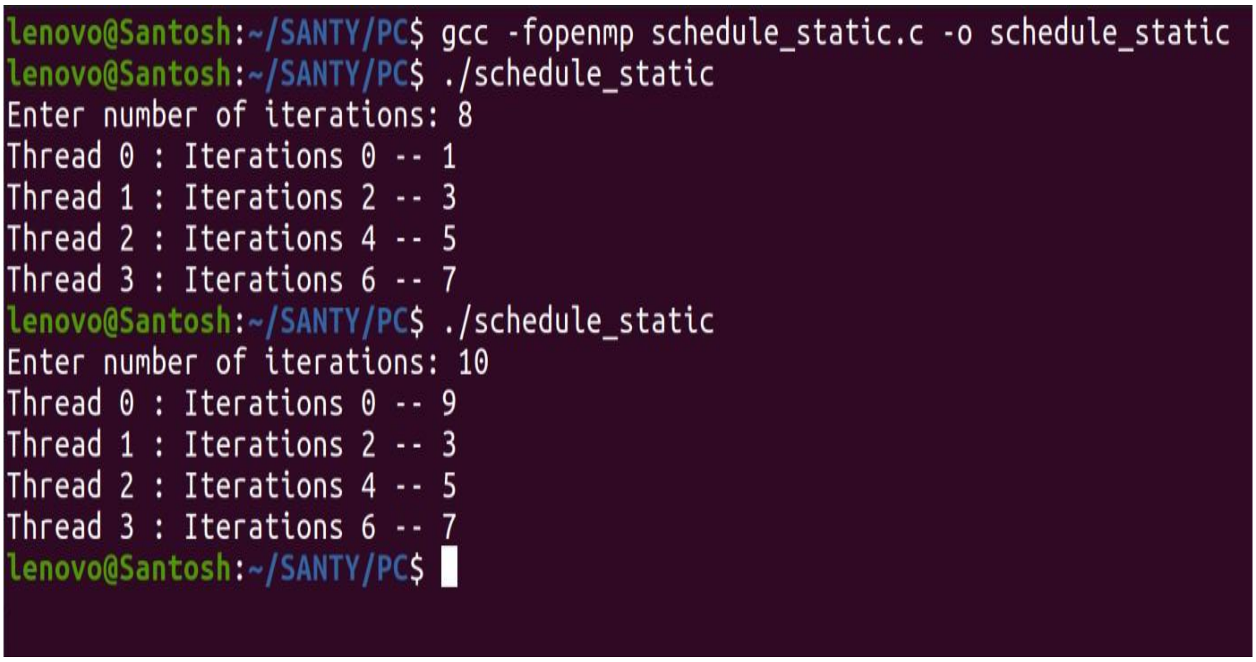
Output :



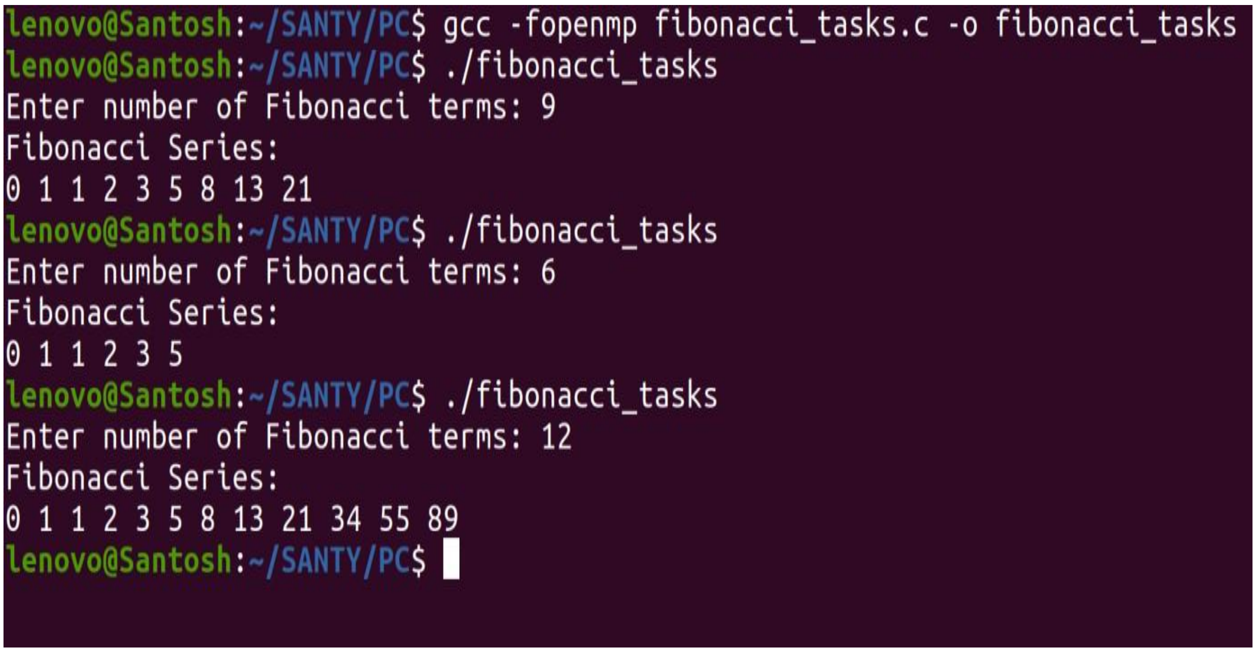
```
lenovo@Santosh:~/SANTY/PC$ gcc -fopenmp mergesort_omp.c -o mergesort
lenovo@Santosh:~/SANTY/PC$ ./mergesort
Sequential Merge Sort Time: 0.014740 seconds
Parallel Merge Sort Time: 0.037430 seconds
lenovo@Santosh:~/SANTY/PC$ OMP_NUM_THREADS=4 ./mergesort
Sequential Merge Sort Time: 0.013276 seconds
Parallel Merge Sort Time: 0.029224 seconds
lenovo@Santosh:~/SANTY/PC$
```

Output :



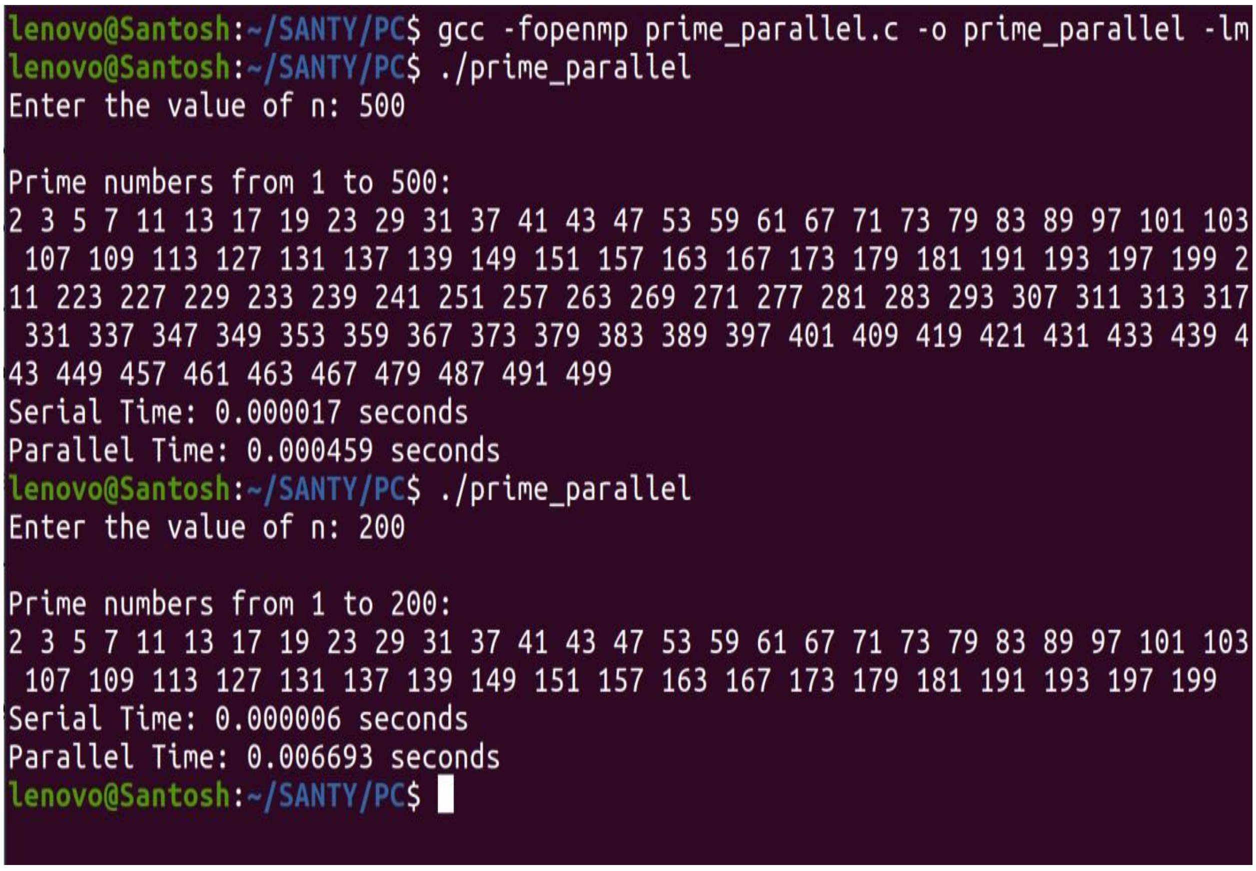
```
lenovo@Santosh:~/SANTY/PC$ gcc -fopenmp schedule_static.c -o schedule_static
lenovo@Santosh:~/SANTY/PC$ ./schedule_static
Enter number of iterations: 8
Thread 0 : Iterations 0 -- 1
Thread 1 : Iterations 2 -- 3
Thread 2 : Iterations 4 -- 5
Thread 3 : Iterations 6 -- 7
lenovo@Santosh:~/SANTY/PC$ ./schedule_static
Enter number of iterations: 10
Thread 0 : Iterations 0 -- 9
Thread 1 : Iterations 2 -- 3
Thread 2 : Iterations 4 -- 5
Thread 3 : Iterations 6 -- 7
lenovo@Santosh:~/SANTY/PC$
```

Output :



```
lenovo@Santosh:~/SANTY/PC$ gcc -fopenmp fibonacci_tasks.c -o fibonacci_tasks
lenovo@Santosh:~/SANTY/PC$ ./fibonacci_tasks
Enter number of Fibonacci terms: 9
Fibonacci Series:
0 1 1 2 3 5 8 13 21
lenovo@Santosh:~/SANTY/PC$ ./fibonacci_tasks
Enter number of Fibonacci terms: 6
Fibonacci Series:
0 1 1 2 3 5
lenovo@Santosh:~/SANTY/PC$ ./fibonacci_tasks
Enter number of Fibonacci terms: 12
Fibonacci Series:
0 1 1 2 3 5 8 13 21 34 55 89
lenovo@Santosh:~/SANTY/PC$ 
```

Output :

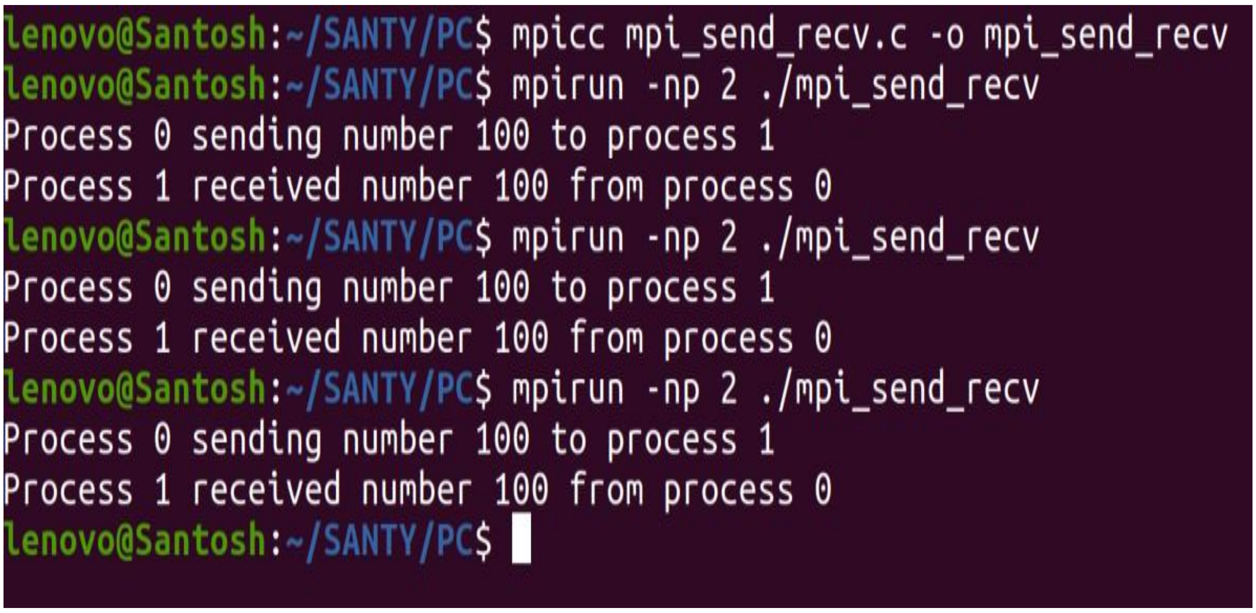


```
lenovo@Santosh:~/SANTY/PC$ gcc -fopenmp prime_parallel.c -o prime_parallel -lm
lenovo@Santosh:~/SANTY/PC$ ./prime_parallel
Enter the value of n: 500

Prime numbers from 1 to 500:
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103
 107 109 113 127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199 2
11 223 227 229 233 239 241 251 257 263 269 271 277 281 283 293 307 311 313 317
 331 337 347 349 353 359 367 373 379 383 389 397 401 409 419 421 431 433 439 4
43 449 457 461 463 467 479 487 491 499
Serial Time: 0.000017 seconds
Parallel Time: 0.000459 seconds
lenovo@Santosh:~/SANTY/PC$ ./prime_parallel
Enter the value of n: 200

Prime numbers from 1 to 200:
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103
 107 109 113 127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199
Serial Time: 0.000006 seconds
Parallel Time: 0.006693 seconds
lenovo@Santosh:~/SANTY/PC$ 
```

Output :

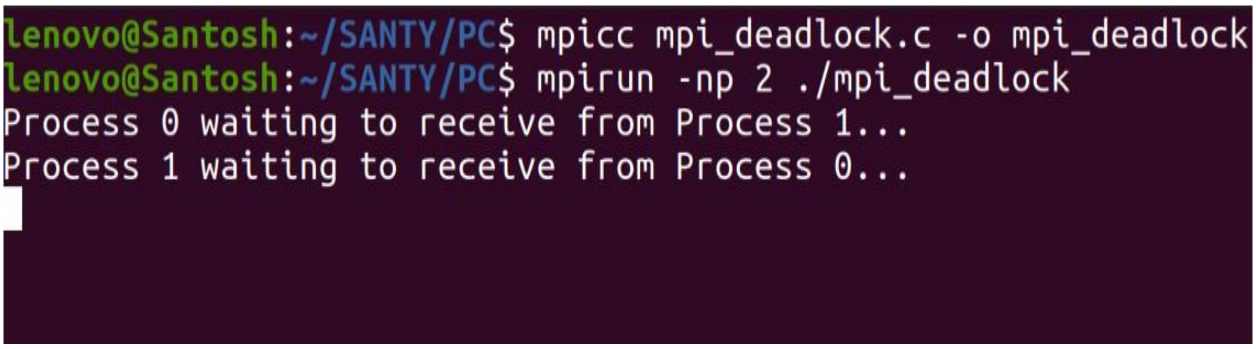


```
lenovo@Santosh:~/SANTY/PC$ mpicc mpi_send_recv.c -o mpi_send_recv
lenovo@Santosh:~/SANTY/PC$ mpirun -np 2 ./mpi_send_recv
Process 0 sending number 100 to process 1
Process 1 received number 100 from process 0
lenovo@Santosh:~/SANTY/PC$ mpirun -np 2 ./mpi_send_recv
Process 0 sending number 100 to process 1
Process 1 received number 100 from process 0
lenovo@Santosh:~/SANTY/PC$ mpirun -np 2 ./mpi_send_recv
Process 0 sending number 100 to process 1
Process 1 received number 100 from process 0
lenovo@Santosh:~/SANTY/PC$ 
```

Output :

Output for MODE = 1 (Deadlock)

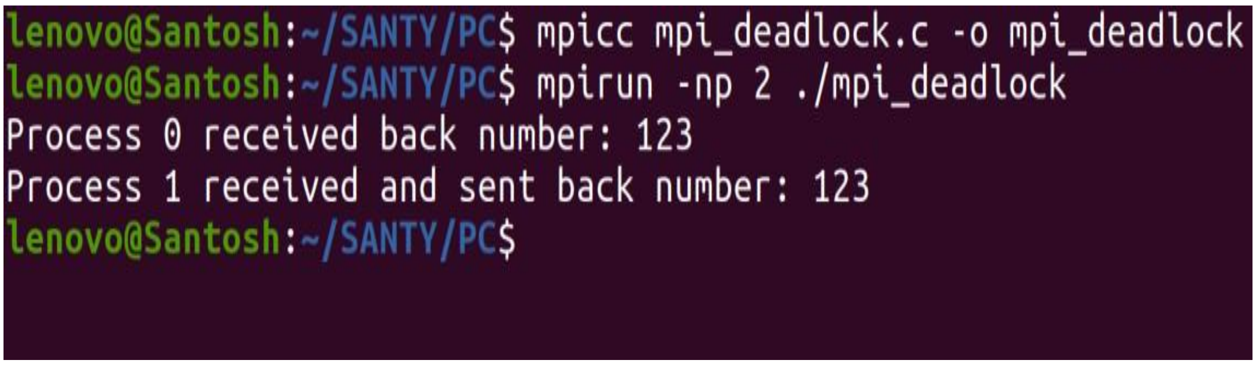
Both processes start with MPI_Recv() and wait for each other's message —
no one sends first → both are blocked forever → deadlock.



```
lenovo@Santosh:~/SANTY/PC$ mpicc mpi_deadlock.c -o mpi_deadlock
lenovo@Santosh:~/SANTY/PC$ mpirun -np 2 ./mpi_deadlock
Process 0 waiting to receive from Process 1...
Process 1 waiting to receive from Process 0...
```

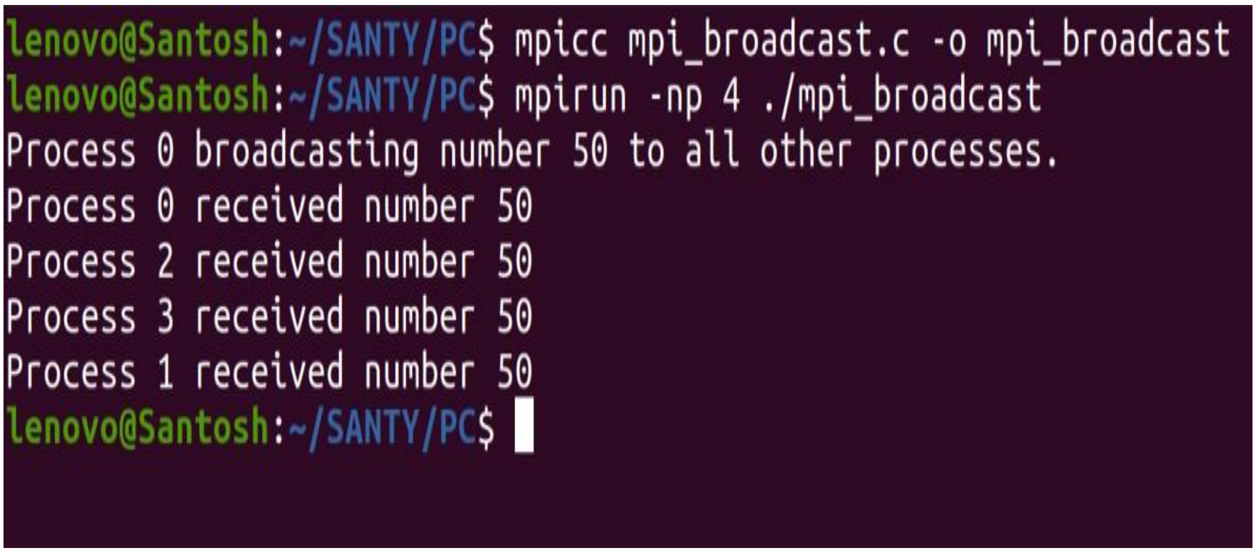
Output for MODE = 2 (Deadlock Avoidance)

Process 0 sends first, process 1 receives first.
Their communication order matches correctly → data exchange succeeds → no deadlock.



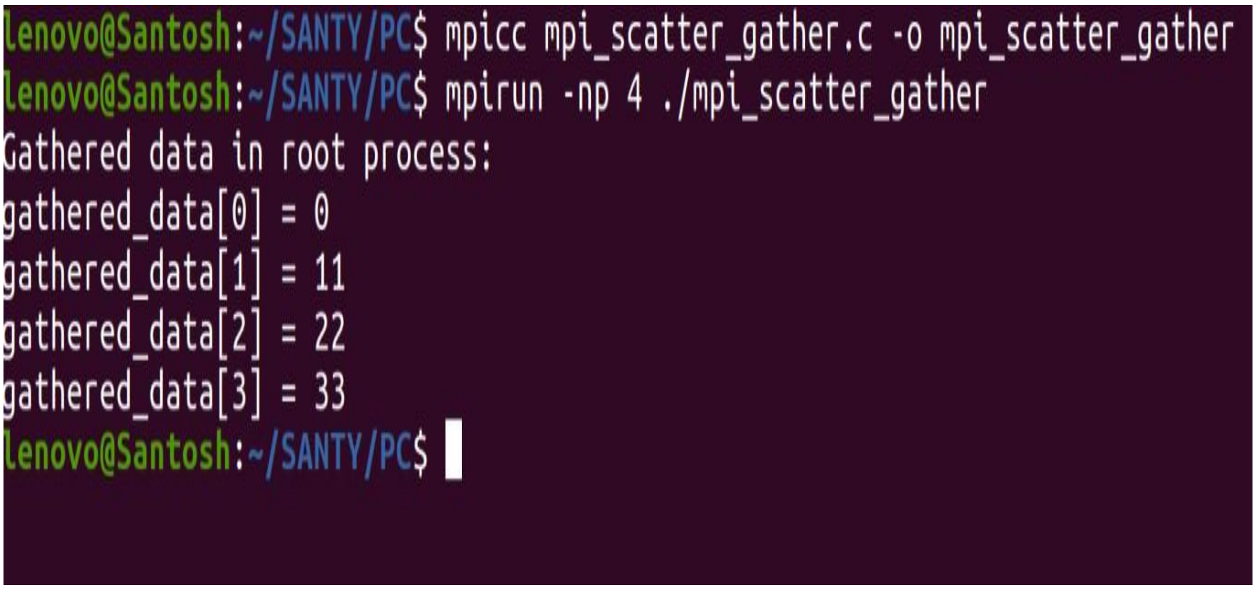
```
lenovo@Santosh:~/SANTY/PC$ mpicc mpi_deadlock.c -o mpi_deadlock
lenovo@Santosh:~/SANTY/PC$ mpirun -np 2 ./mpi_deadlock
Process 0 received back number: 123
Process 1 received and sent back number: 123
lenovo@Santosh:~/SANTY/PC$
```

Output :



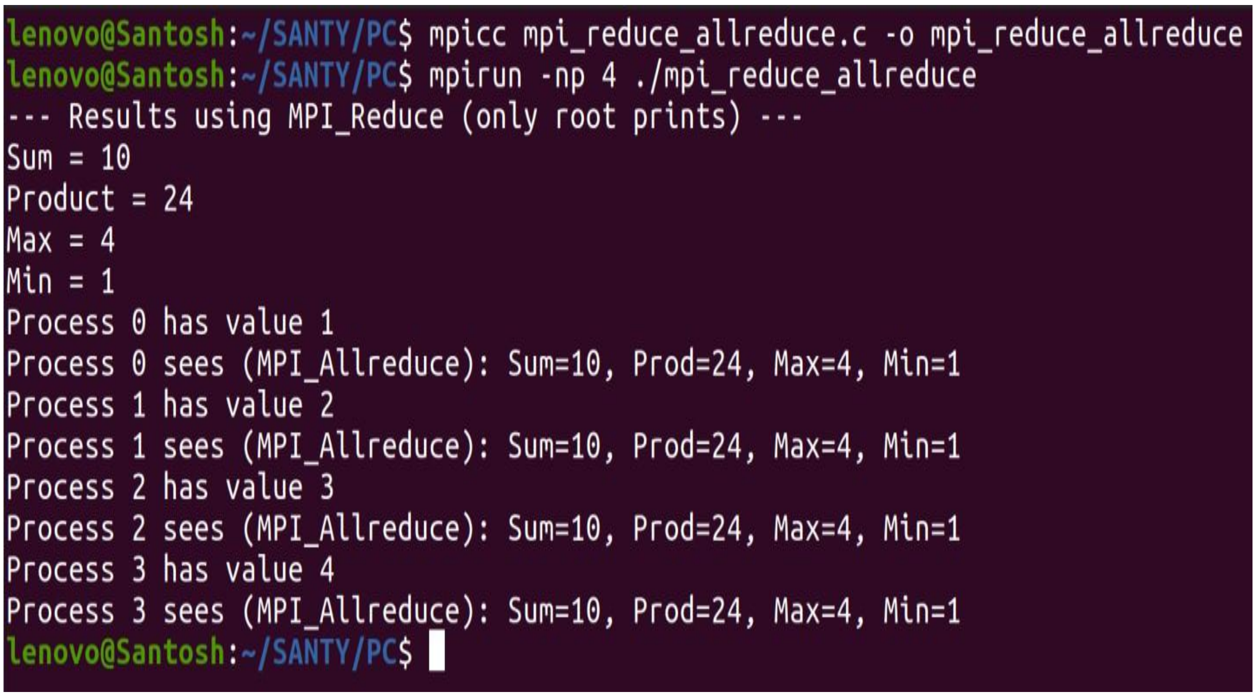
```
lenovo@Santosh:~/SANTY/PC$ mpicc mpi_broadcast.c -o mpi_broadcast
lenovo@Santosh:~/SANTY/PC$ mpirun -np 4 ./mpi_broadcast
Process 0 broadcasting number 50 to all other processes.
Process 0 received number 50
Process 2 received number 50
Process 3 received number 50
Process 1 received number 50
lenovo@Santosh:~/SANTY/PC$ 
```

Output :



```
lenovo@Santosh:~/SANTY/PC$ mpicc mpi_scatter_gather.c -o mpi_scatter_gather
lenovo@Santosh:~/SANTY/PC$ mpirun -np 4 ./mpi_scatter_gather
Gathered data in root process:
gathered_data[0] = 0
gathered_data[1] = 11
gathered_data[2] = 22
gathered_data[3] = 33
lenovo@Santosh:~/SANTY/PC$
```

Output :



```
lenovo@Santosh:~/SANTY/PC$ mpicc mpi_reduce_allreduce.c -o mpi_reduce_allreduce
lenovo@Santosh:~/SANTY/PC$ mpirun -np 4 ./mpi_reduce_allreduce
--- Results using MPI_Reduce (only root prints) ---
Sum = 10
Product = 24
Max = 4
Min = 1
Process 0 has value 1
Process 0 sees (MPI_Allreduce): Sum=10, Prod=24, Max=4, Min=1
Process 1 has value 2
Process 1 sees (MPI_Allreduce): Sum=10, Prod=24, Max=4, Min=1
Process 2 has value 3
Process 2 sees (MPI_Allreduce): Sum=10, Prod=24, Max=4, Min=1
Process 3 has value 4
Process 3 sees (MPI_Allreduce): Sum=10, Prod=24, Max=4, Min=1
lenovo@Santosh:~/SANTY/PC$
```