

Walchand College of Engineering, Sangli
Department of Computer Science and Engineering
Class: Final Year (Computer Science and Engineering)
Year: 2021-22 **Semester:** 1
Course: High Performance Computing Lab

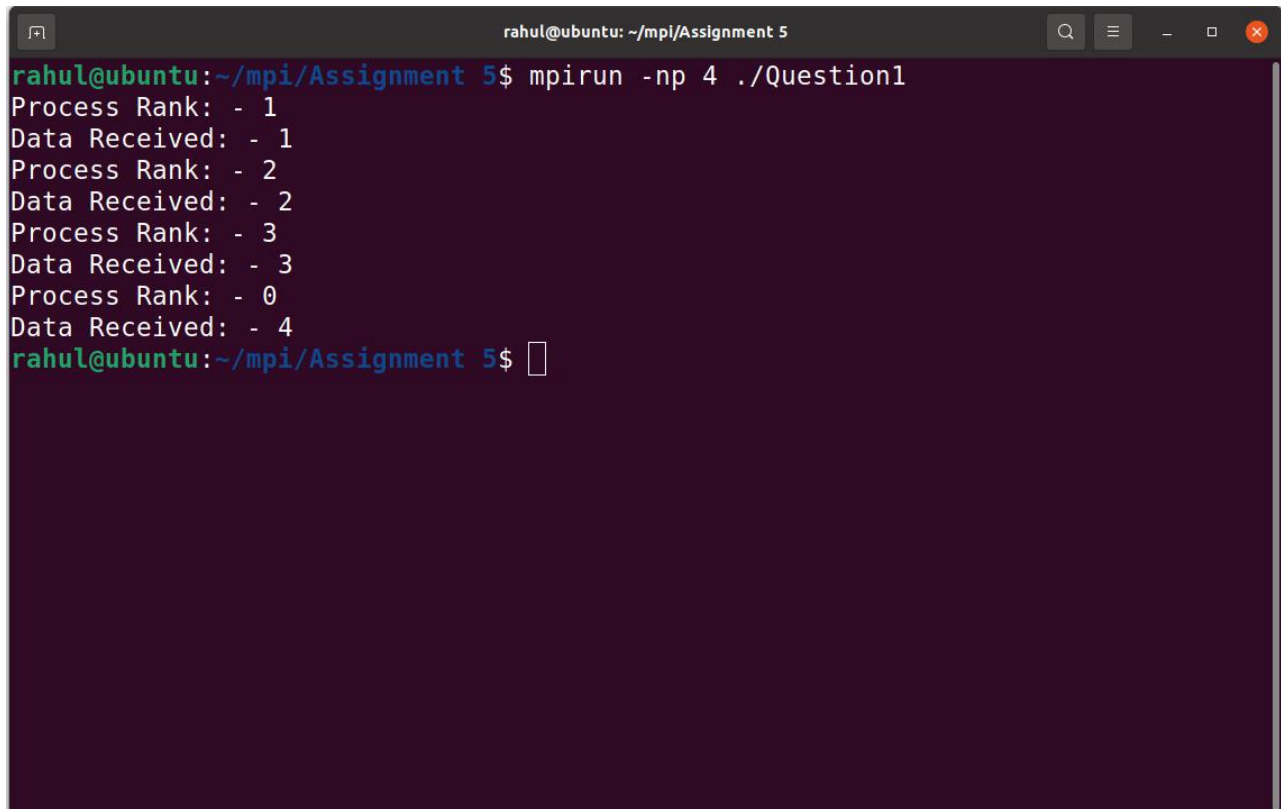
Practical No. 5

Exam Seat No: 2018BTECS00005

Name: Rahul Sanjay Naravadkar

Problem Statement 1: Implement a MPI program to give an example of blocking send and receive between four processes.

Screenshot 1:

A screenshot of a terminal window titled 'rahul@ubuntu: ~/mpi/Assignment 5'. The terminal shows the command 'mpirun -np 4 ./Question1' being executed. The output consists of four lines, each representing a process: 'Process Rank: - 1', 'Data Received: - 1', 'Process Rank: - 2', 'Data Received: - 2', 'Process Rank: - 3', 'Data Received: - 3', 'Process Rank: - 0', and 'Data Received: - 4'. The prompt 'rahul@ubuntu:~/mpi/Assignment 5\$' is visible at the bottom of the terminal window.

```
rahul@ubuntu:~/mpi/Assignment 5$ mpirun -np 4 ./Question1
Process Rank: - 1
Data Received: - 1
Process Rank: - 2
Data Received: - 2
Process Rank: - 3
Data Received: - 3
Process Rank: - 0
Data Received: - 4
rahul@ubuntu:~/mpi/Assignment 5$
```

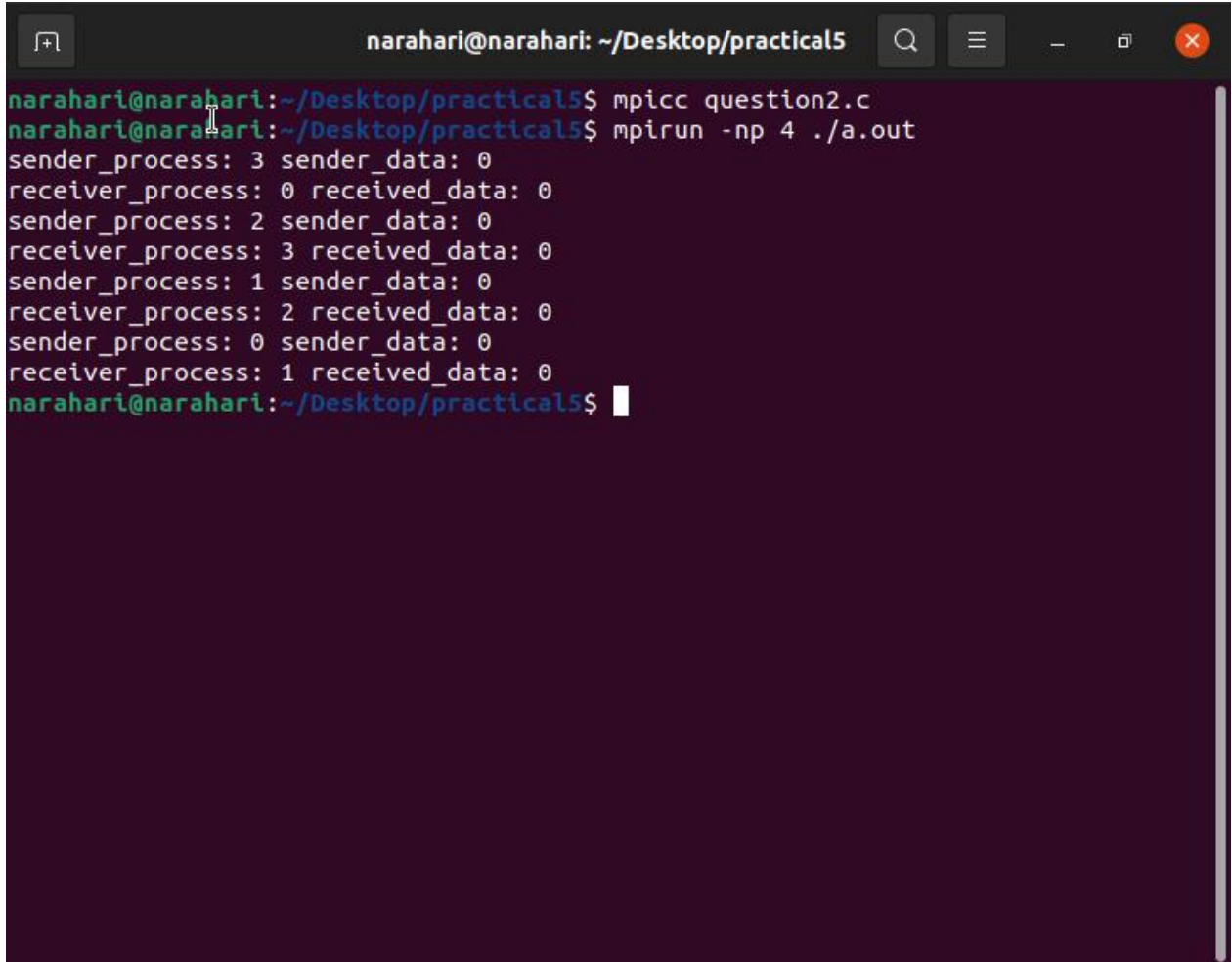
Information #:

Compile: - mpicxx -o Question1 Question1.cpp

Run - mpirun -np 4 ./Question1

Problem statement 2 : Implement MPI program using non-blocking send & receive functions to demonstrate. Nearest neighbour exchange of data in a ring topology

Screenshot:



```
narahari@narahari: ~/Desktop/practical5
narahari@narahari:~/Desktop/practical5$ mpicc question2.c
narahari@narahari:~/Desktop/practical5$ mpirun -np 4 ./a.out
sender_process: 3 sender_data: 0
receiver_process: 0 received_data: 0
sender_process: 2 sender_data: 0
receiver_process: 3 received_data: 0
sender_process: 1 sender_data: 0
receiver_process: 2 received_data: 0
sender_process: 0 sender_data: 0
receiver_process: 1 received_data: 0
narahari@narahari:~/Desktop/practical5$
```

Information #:

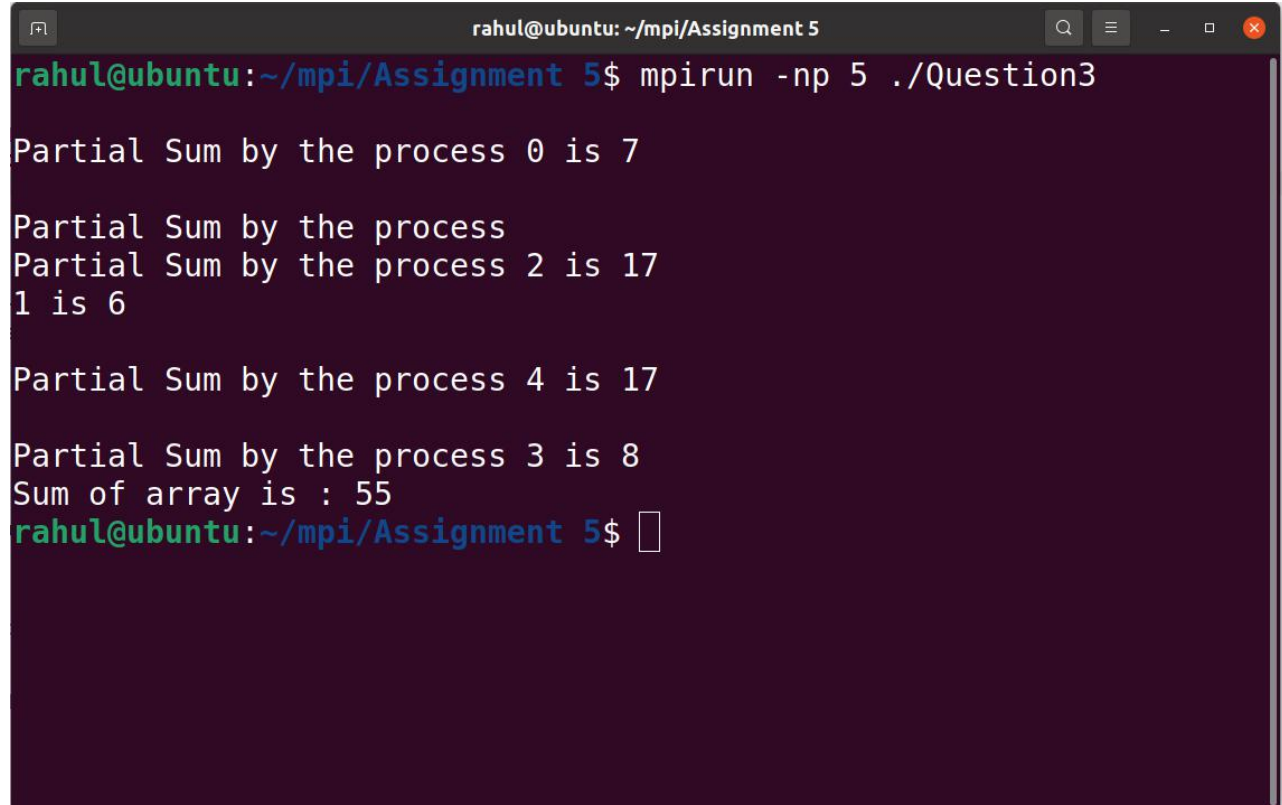
Compile: - mpicxx -o Question2 Question2.cpp

Run - mpirun -np 4 ./Question2

Problem Statement 3:

Write a MPI program to find the sum of all the elements of an array A of size n using m number of processes. The two sums then are added to get the final result.

Screenshot:



```
rahul@ubuntu: ~/mpi/Assignment 5$ mpirun -np 5 ./Question3
Partial Sum by the process 0 is 7
Partial Sum by the process
Partial Sum by the process 2 is 17
1 is 6
Partial Sum by the process 4 is 17
Partial Sum by the process 3 is 8
Sum of array is : 55
rahul@ubuntu:~/mpi/Assignment 5$
```

Information #:

Compile: - mpicxx -o Question3 Question3.cpp

Run - mpirun -np 5 ./Question3

[Github Link](#)