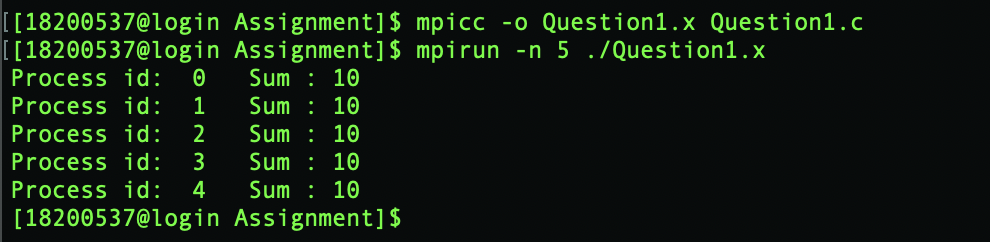
**Question 1: Communication in a Ring**

Here the program sends a message to its right neighbour and message terminates once it reached its originator rank. Message contains the sum of all the ranks.

Here are the commands to execute the code:

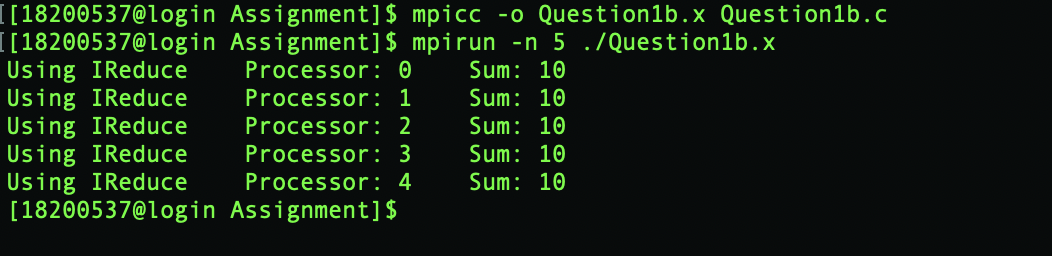
Using non-blocking communications:

Here I’m using MPI\_Issend to send and MPI\_Recv to receive message.



Using MPI Ireduce calls:

Here I’m using MPI\_Ireduce to send message instead of MPI\_Issend.



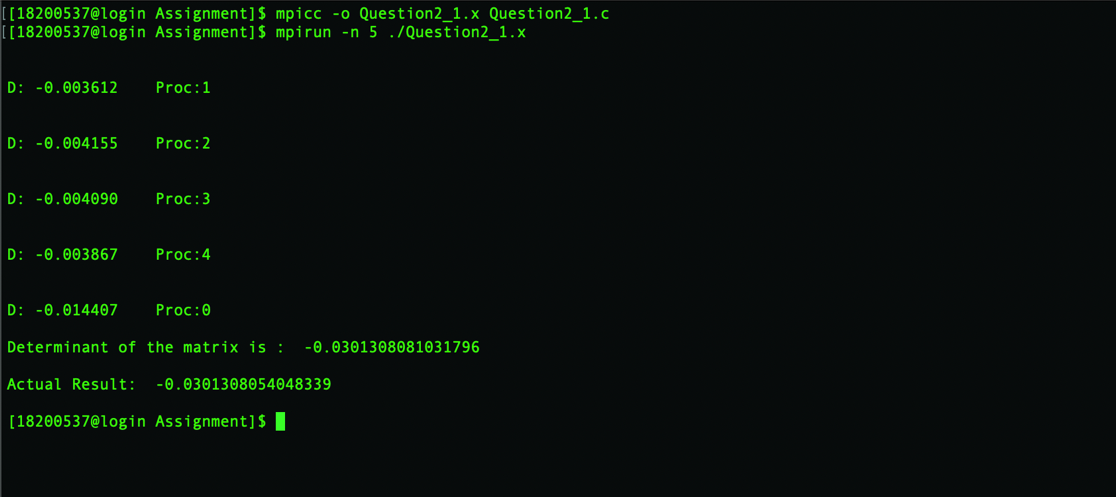
**Question 2: Determinant of 5x5 matrix**

Here we are calculating the determinant of 5x5 matrix using Cramer’s rules and printing the result on rank 0.

Steps:

1. Find the five (based on rank value) 4x4 submatrices of original 5x5 matrix i.e. from the original 5x5 matrix remove the respective column based on rank value and row number 0.
2. For each 4x4 submatrix determine four 3x3 matrices.
3. Find the determinant of each of the 3x3 matrices and multiple this return value with respective value and sign in 4x4 and similarly multiple this return value to with respective value in 5x5 and sign to obtain determinant based on rank.
4. Sum all these 5 determinants to obtain the final determinant of 5x5 matrix.

Command to execute:



**Question 3:**

Deadlock and removal of bugs:

Deadlock since *MPI\_Bcast* involve all the processes grouped in a communicator and must be called by all the processes in the communicator here we need to remove the *MPI\_Bcast* command outside the if condition (which tell to execute *MPI\_Bcast* only if rank==0).

Bugs:

Since processor 0 reads from a file 25 random integers and processor 0 broadcasts these integers to all the other processors we need to change the value 1 to 0 and change MPI\_Double to MPI\_INT because we are broadcasting only integer values.

After modification code is taking less than 30 seconds to execute.

Given:

ierr = MPI\_Bcast( indata, 25, MPI\_Double, 1, MPI\_COMM\_WORLD);

Modified:

ierr = MPI\_Bcast( indata, 25, MPI\_INT, 0, MPI\_COMM\_WORLD);

