NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA SURATHKAL
DEPARTMENT OF INFORMATION TECHNOLOGY

IT 301 Parallel Computing LAB 8

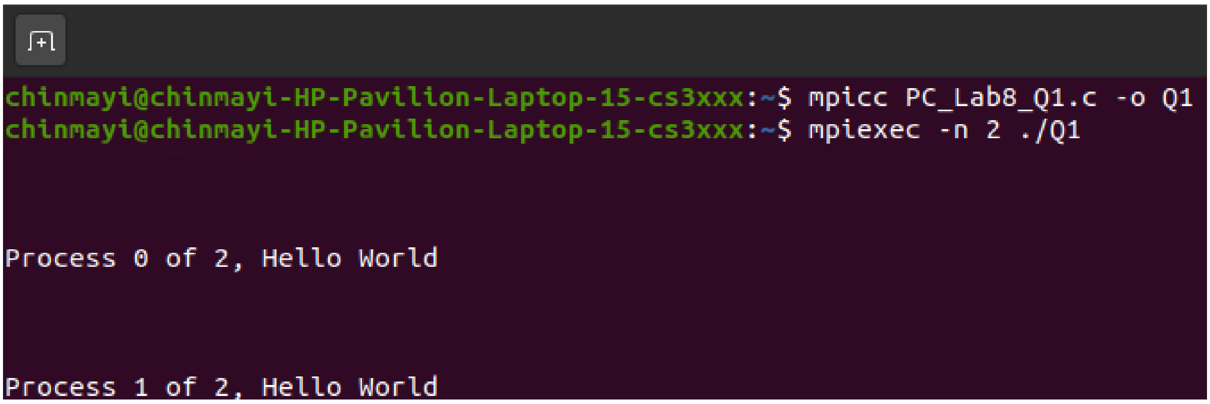
14th October 2020

Faculty: Dr. Geetha V and Mrs. Thanmayee

Name: Chinmayi C. Ramakrishna

Roll No.: 181IT113

1. MPI “Hello World” program:

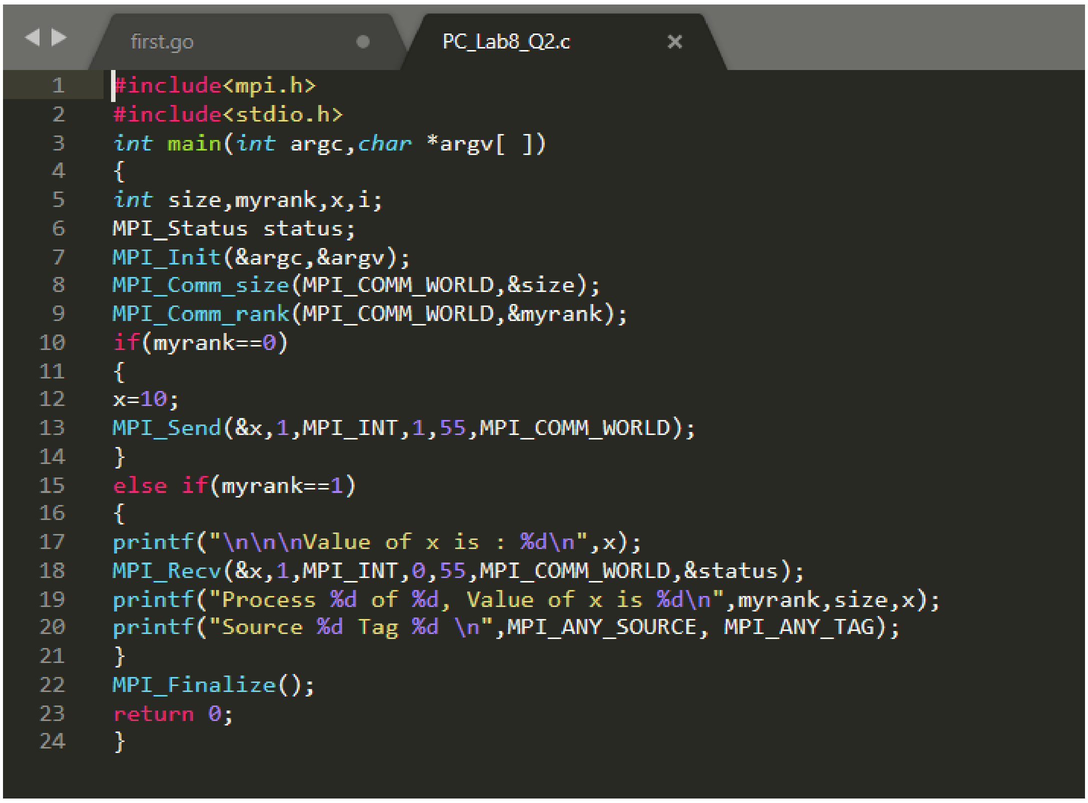


```
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ mpicc PC_Lab8_Q1.c -o Q1
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ mpiexec -n 2 ./Q1



Process 0 of 2, Hello World



Process 1 of 2, Hello World
```

Hello World is printed once for each process.

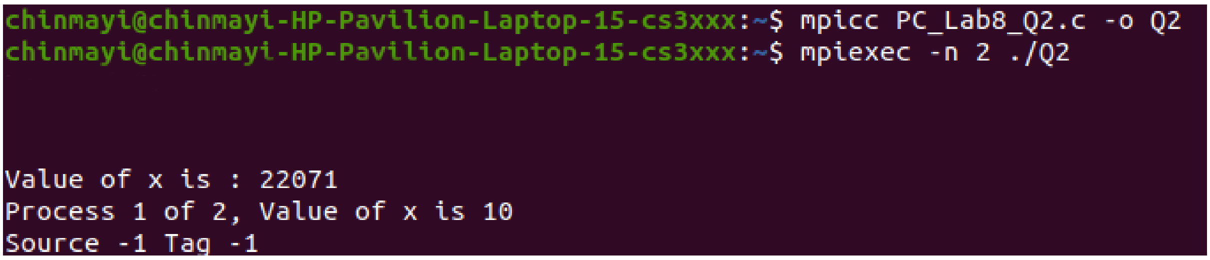
2. Demonstration of MPI_Send() and MPI_Recv(). Sending an Integer.

Code:



```c
#include<mpi.h>
#include<stdio.h>
int main(int argc,char *argv[ ])
{
int size,myrank,x,i;
MPI_Status status;
MPI_Init(&argc,&argv);
MPI_Comm_size(MPI_COMM_WORLD,&size);
MPI_Comm_rank(MPI_COMM_WORLD,&myrank);
if(myrank==0)
{
x=10;
MPI_Send(&x,1,MPI_INT,1,55,MPI_COMM_WORLD);
}
else if(myrank==1)
{
printf("\n\n\nValue of x is : %d\n",x);
MPI_Recv(&x,1,MPI_INT,0,55,MPI_COMM_WORLD,&status);
printf("Process %d of %d, Value of x is %d\n",myrank,size,x);
printf("Source %d Tag %d \n",MPI_ANY_SOURCE, MPI_ANY_TAG);
}
MPI_Finalize();
return 0;
}
```

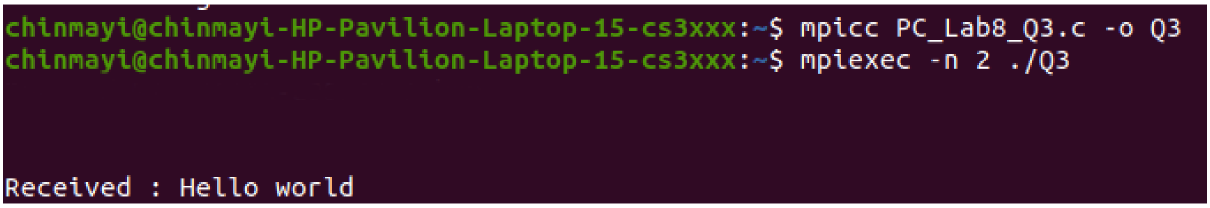
Output:



```
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ mpicc PC_Lab8_Q2.c -o Q2
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ mpiexec -n 2 ./Q2



Value of x is : 22071
Process 1 of 2, Value of x is 10
Source -1 Tag -1
```

Mpi_any_source tells MPI to receive the message without restricting the rank of the sender.Mpi_any_tag tells MPI to give any tag to the process.

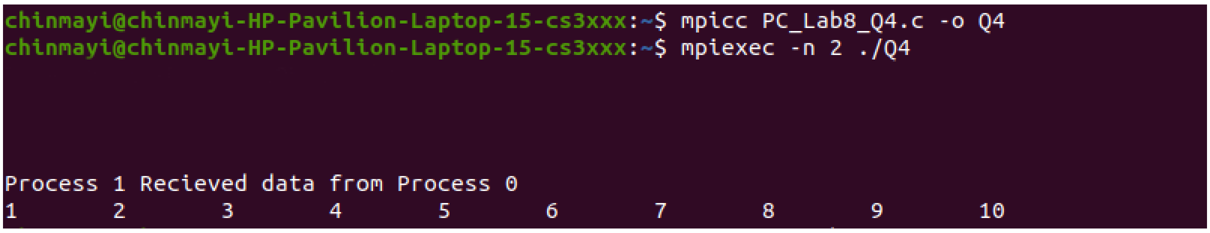
3. Demonstration of MPI_Send() and MPI_Recv(). Sending a string.



```
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ mpicc PC_Lab8_Q3.c -o Q3
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ mpiexec -n 2 ./Q3



Received : Hello world
```

The message sent by process 0 is successfully received by process 1.

4. Demonstration of MPI_Send() and MPI_Recv(). Sending elements of an array.

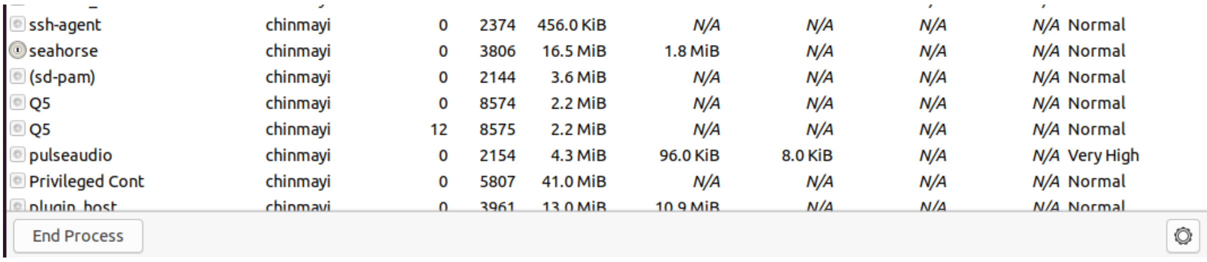


```
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ mpicc PC_Lab8_Q4.c -o Q4
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ mpiexec -n 2 ./Q4




Process 1 Recieved data from Process 0
1         2         3         4         5         6         7         8         9         10
```

A count of first 10 messages is successfully received by process 1 from process 0.

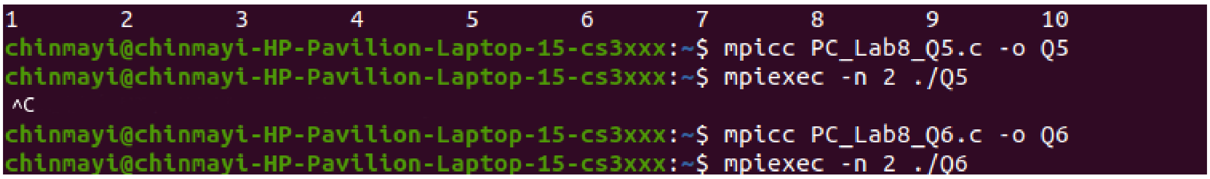
5. Demonstration of Blocking Send and Receive with mismatched tags.

Output:



ssh-agent	chinmayi	0	2374	456.0 KiB	N/A	N/A	N/A	N/A	Normal
seahorse	chinmayi	0	3806	16.5 MiB	1.8 MiB	N/A	N/A	N/A	Normal
(sd-pam)	chinmayi	0	2144	3.6 MiB	N/A	N/A	N/A	N/A	Normal
Q5	chinmayi	0	8574	2.2 MiB	N/A	N/A	N/A	N/A	Normal
Q5	chinmayi	12	8575	2.2 MiB	N/A	N/A	N/A	N/A	Normal
pulseaudio	chinmayi	0	2154	4.3 MiB	96.0 KiB	8.0 KiB	N/A	N/A	Very High
Privileged Cont	chinmayi	0	5807	41.0 MiB	N/A	N/A	N/A	N/A	Normal
plugin_host	chinmayi	0	3961	13.0 MiB	10.9 MiB	N/A	N/A	N/A	Normal

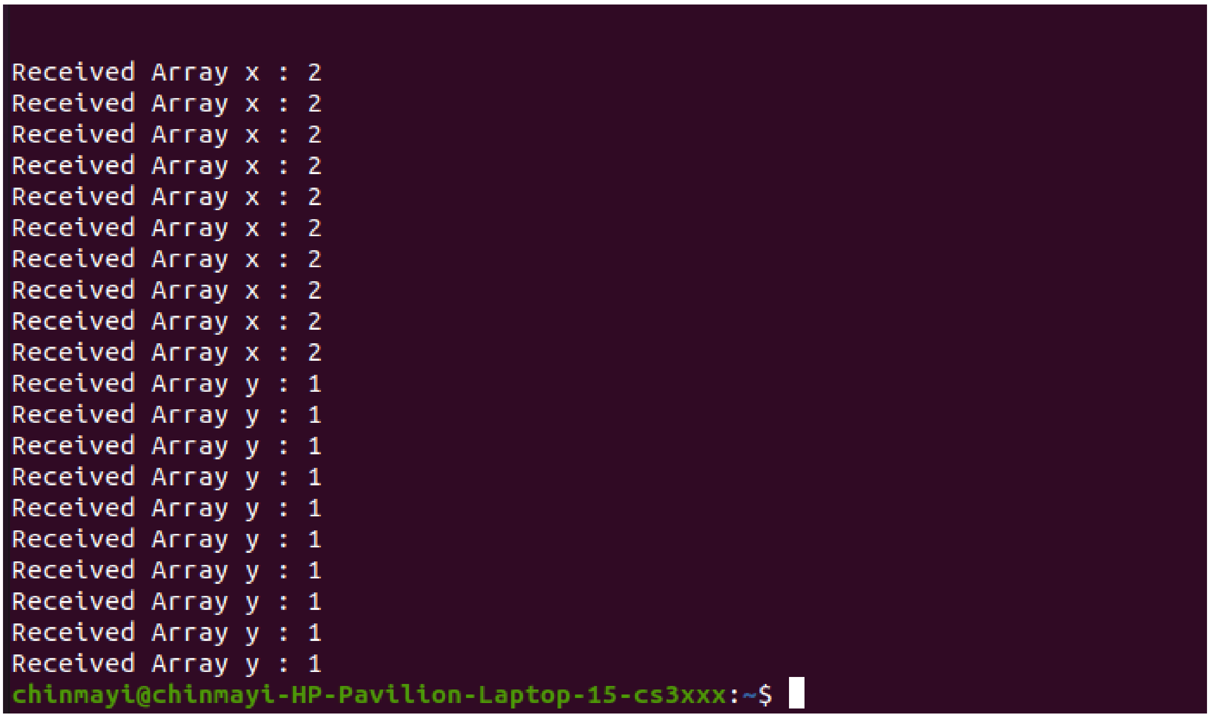
Process when it’s being executed.



```
1         2         3         4         5         6         7         8         9         10
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ mpicc PC_Lab8_Q5.c -o Q5
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ mpiexec -n 2 ./Q5
^C
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ mpicc PC_Lab8_Q6.c -o Q6
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ mpiexec -n 2 ./Q6
```

The process ends when the code is exited.

6. **MPI_Send() and MPI_Recv() standard mode:**



```
Received Array x : 2
Received Array x : 2
Received Array x : 2
Received Array x : 2
Received Array x : 2
Received Array x : 2
Received Array x : 2
Received Array x : 2
Received Array x : 2
Received Array x : 2
Received Array y : 1
Received Array y : 1
Received Array y : 1
Received Array y : 1
Received Array y : 1
Received Array y : 1
Received Array y : 1
Received Array y : 1
Received Array y : 1
Received Array y : 1
chinmayi@chinmayi-HP-Pavilion-Laptop-15-cs3xxx:~$ 
```

a) Note down your observation on the content of x and y at Process 1.

The value of x is not received as the tags of the send of process 0 and receive of process 1 do not match.

The value of y is not received as the tags of the send of process 0 and receive of process 1 do not match.

b) Explain the importance of tag.

When process 1 wants to send many different types of messages, it can use ids called tags so that the receiver can differentiate the messages.

c) Write your analysis about Blocking Send and Receive. Whether it is advantageous?

The buffer passed to mpi send can be reused. Mpi receive returns when the receive buffer has been filled with valid data.

d) What is the need for Non-blocking Send and Receive?

The use of non - blocking receives may also avoid system buffering and memory-to-memory copying, as information is provided early on the location of the receive buffer