High-Performance Computing Assignment (MPI)

Team Members

Keerti Chaudhary: 181CO226

Trivedi Naman Manishbhai: 181CO156

Q1

Given below is the output.

```
hp@hp-HP-Pavilion-Laptop-15-cc1xx: ~/Desktop/hpc/Assignment-3/Q1
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q1$ mpicc -o hello ./q1.c
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q1$ mpirun -n 4 ./hello
Hello world from process of Rank 0 out of 4, running on hp-HP-Pavilion-Laptop-15-cc1xx
Hello world from process of Rank 1 out of 4, running on hp-HP-Pavilion-Laptop-15-cc1xx
Hello world from process of Rank 2 out of 4, running on hp-HP-Pavilion-Laptop-15-cc1xx
Hello world from process of Rank 3 out of 4, running on hp-HP-Pavilion-Laptop-15-cc1xx
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q1$
```

The number of threads is 4. Hence “Hello World !” is printed four times.

Q2

Given below is the output

```
hp@hp-HP-Pavilion-Laptop-15-cc1xx: ~/Desktop/hpc/Assignment-3/Q2
q2.c
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q2$ mpicc -o daxpy
./q2.c
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q2$ mpirun -n 4 ./d
axpy
No. of processes: 4
Time taken for multiprocessor calculation: 0.000647
Time taken for uniprocessor implementation: 0.001982
3.063249 times faster
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q2$
```

Time taken is shown in the output.

Q3

```
hp@hp-HP-Pavilion-Laptop-15-cc1xx: ~/Desktop/hpc/Assignment-3/Q3
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q3$ mpicc -o hello ./q3.c
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q3$ mpirun -n 4 ./hello
Hello world from 1
Hello world from 2
Hello world from 3
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q3$
```

“Hello, world” message is sent to the master process from the other three processes.

Q4

```
Hello world     Rank : 2        Number received : 845488520 from rank 1

Hello world     Rank : 1        Number received : 551618186 from rank 0
```

Q5

```
hp@hp-HP-Pavilion-Laptop-15-cc1xx: ~/Desktop/hpc/Assignment-3/Q5
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q5$ mpicc -o calc ./q5.c
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q5$ mpirun -n 4 ./calc
Calculated value of pi: 3.141593
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q5$
```

The value of pi is calculated with four threads.

Q6

```
~/5SEM/HPC/MPI_Assignment/6  main !3 ?2
> mpirun -n 4 a.out
MultiProc Time: 13.6696 milliseconds.
~/5SEM/HPC/MPI_Assignment/6  main !3 ?2
```

Q7

```
hp@hp-HP-Pavilion-Laptop-15-cc1xx: ~/Desktop/hpc/Assignment-3/Q7
q7.c
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q7$ mpicc -o reduce ./q7.c
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q7$ mpirun -n 4 ./reduce
Sum: 500000500000
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q7$
```

The Sum of N = 1000000, is shown in the output.

Q8

```
hp@hp-HP-Pavilion-Laptop-15-cc1xx: ~/Desktop/hpc/Assignment-3/Q8
hp@hp-HP-Pavilion-Laptop-15-cc1xx: ~/Desk...   hp@hp-HP-Pavilion-Laptop-15-cc1xx: ~/Desk...
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q8$ mpicc -o col ./q8.c -lm
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q8$ mpirun -n 4 ./col
The old array:
48.000000 5.000000 40.000000 96.000000
The new array:
6.928203 2.236068 6.324555 9.797959 hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assi
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q8$
```

The square root of the old array is calculated by scattering its elements into different processes, summing, the square root there, and gathering it back at the master process.

Q9

```
hp@hp-HP-Pavilion-Laptop-15-cc1xx: ~/Desktop/hpc/Assignment-3/Q9
hp@hp-HP-Pavilion-Laptop-15-cc1xx: ~/Desk...   hp@hp-HP-Pavilion-Laptop-15-cc1xx: ~/Desk...
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q9$ mpicc -o derived ./q9.c
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q9$ mpirun -n 4 ./derived
Collective Communication Rank : 1 Structure : 1 - 49 50 - 49.000000 50.000000 51.000000
52.000000
Point-to-point communication Rank : 1 Structure : a - 97 98 - 97.000000 98.000000 99.000
000 100.000000
Collective Communication Rank : 2 Structure : 1 - 49 50 - 49.000000 50.000000 51.000000
52.000000
Point-to-point communication Rank : 2 Structure : a - 97 98 - 97.000000 98.000000 99.000
000 100.000000
Collective Communication Rank : 0 Structure : 1 - 49 50 - 49.000000 50.000000 51.000000
52.000000
Collective Communication Rank : 3 Structure : 1 - 49 50 - 49.000000 50.000000 51.000000
52.000000
Point-to-point communication Rank : 3 Structure : a - 97 98 - 97.000000 98.000000 99.000
000 100.000000
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q9$
```

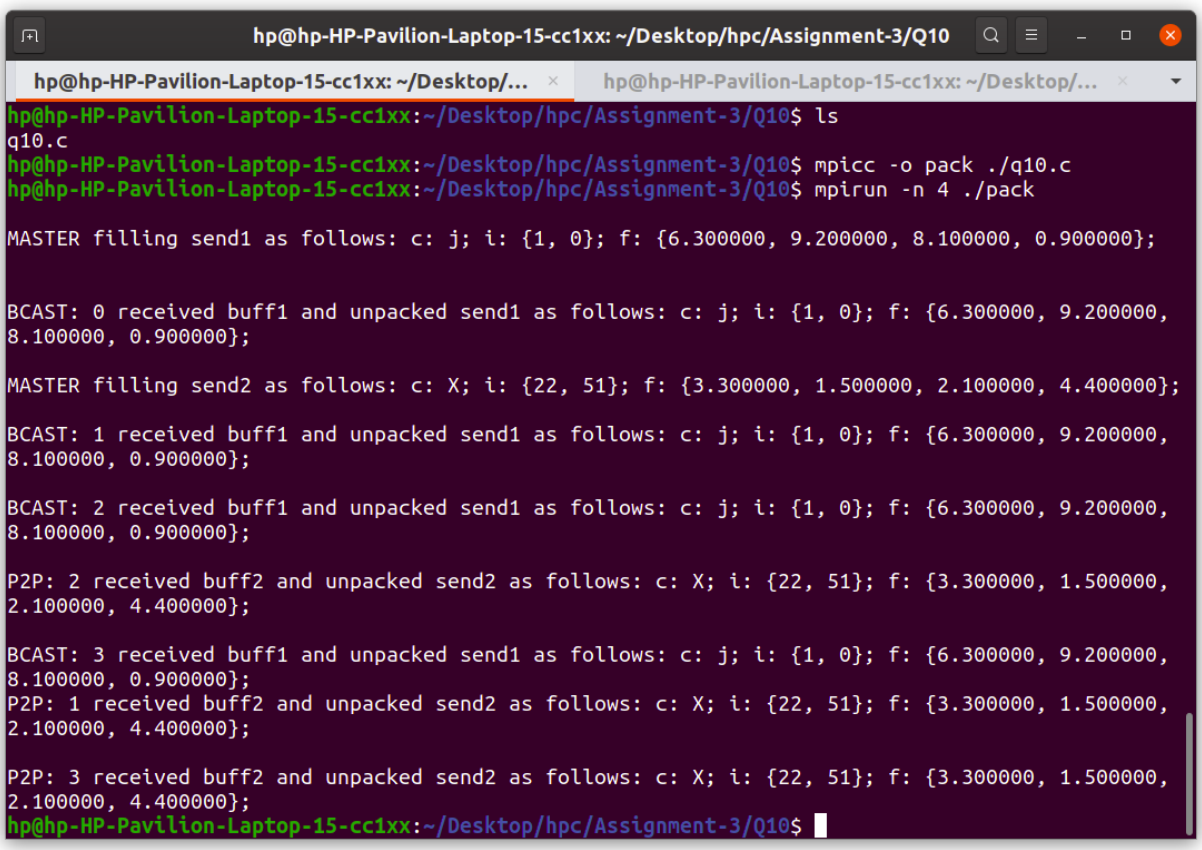
The following code snippet is used to fill the structure

```
struct dd get_filled_struct(char key)

{   struct dd temp;

    temp.c = key;

    for(int i=0;i<4;++i)
```

```
{   if(i<2)
        {
                temp.iA[i] = key + i;
        }
        temp.fA[i] = key + i;
    } return temp;
}
```

The derived datatype was created and broadcasted by the master process, whose output is shown prefixed with *Collective Communication.* Also, the master process individually sends the data type to the other process, whose output is prefixed by *Point-to-point communication.*

Q10



The creation of a derived datatype to help communicate compound data structures is done using the Pack and Unpack functions in MPI.

Q11

```
hp@hp-HP-Pavilion-Laptop-15-cc1xx: ~/Desktop/hpc/Assignment-3/Q11
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q11$ mpirun -n 4 ./derived
Sent Matrix
0 1 2 3
4 5 6 7
8 9 10 11
12 13 14 15
Received Matrix
0 1 2 3
0 5 6 7
0 0 10 11
0 0 0 15
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q11$
```

An indexed derived datatype which taken only the upper triangle of a matrix is declared. The above-shown matrix is sent by the master process using this derived datatype to process 1.

The matrix received by process 1 is an upper triangular matrix as shown in the output.

Q12

```
hp@hp-HP-Pavilion-Laptop-15-cc1xx: ~/Desktop/hpc/Assignment-3/Q12
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q12$ ls
q12.c
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q12$ mpicc -o matrix ./q12.c
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q12$ mpirun -n 4 ./matrix
dotProduct = 799400.100000
hp@hp-HP-Pavilion-Laptop-15-cc1xx:~/Desktop/hpc/Assignment-3/Q12$
```

Here matrix is processed in chunks and each thread performs optimizations and reductions and the final result is calculated by the master thread who then prints it out.

Q14

```
naman@naman-Lenovo-ideapad-530S-15IKB:~/code$ mpicc try.c
naman@naman-Lenovo-ideapad-530S-15IKB:~/code$ ./a.out
Matrix size:
3
Time: 0.0000s
naman@naman-Lenovo-ideapad-530S-15IKB:~/code$ ^C
naman@naman-Lenovo-ideapad-530S-15IKB:~/code$ ./a.out
Matrix size:
15
Time: 0.0000s
naman@naman-Lenovo-ideapad-530S-15IKB:~/code$ ./a.out
Matrix size:
10000
^C
naman@naman-Lenovo-ideapad-530S-15IKB:~/code$ ./a.out
Matrix size:
100
Time: 0.0041s
naman@naman-Lenovo-ideapad-530S-15IKB:~/code$
```

The cannon's algorithm was implemented. The output is shown for different matrices. The matrices are filled using the following code snippet.

```
for(i=0;i<Nl;i++)
        for(j=0;j<Nl;j++) {
        A[i*Nl+j]=5-(int)( 10.0 * rand() / ( RAND_MAX + 1.0 ) );
        }
```

The input of the dimension of the matrix is taken from the user.