CO316-Assignment 6

Hardik Rana -16CO138
Harshal Shinde -16CO223

QUESTION 1 - HELLO WORLD PROGRAM - VERSION1

Given below is the output.Here number of threads is 4,hence “Hello World !” is printed 4 times.

```
hardik@ubuntu:~/Desktop$ cd Assignment6/
hardik@ubuntu:~/Desktop/Assignment6$ cd Q1/
hardik@ubuntu:~/Desktop/Assignment6/Q1$ mpicc Q1.c -o Q1
hardik@ubuntu:~/Desktop/Assignment6/Q1$ mpirun -np 4 ./Q1
Hello world from process of Rank 1 out of 4, running on ubuntu
Hello world from process of Rank 3 out of 4, running on ubuntu
Hello world from process of Rank 2 out of 4, running on ubuntu
Hello world from process of Rank 0 out of 4, running on ubuntu
hardik@ubuntu:~/Desktop/Assignment6/Q1$
```

QUESTION 2 - DAXPY LOOP

Given below is the output.Here speedup is shown by increasing number of threads in output.

```
hardik@ubuntu:~/Desktop/Assignment6$ cd Q2
hardik@ubuntu:~/Desktop/Assignment6/Q2$ mpicc Q2.c -o Q2
hardik@ubuntu:~/Desktop/Assignment6/Q2$ mpirun -np 2 ./Q2
Time taken by 2 processes : 0.585933 seconds
Speedup for 2 processes : 6.637285
hardik@ubuntu:~/Desktop/Assignment6/Q2$ mpirun -np 4 ./Q2
Time taken by 4 processes : 0.716989 seconds
Speedup for 4 processes : 3.867776
hardik@ubuntu:~/Desktop/Assignment6/Q2$ mpirun -np 8./Q2
--------------------------------------------------------------------------
Open MPI has detected that a parameter given to a command line
option does not match the expected format:

  Option: np
  Param:  8./Q2

This is frequently caused by omitting to provide the parameter
to an option that requires one. Please check the command line and try again.
--------------------------------------------------------------------------
hardik@ubuntu:~/Desktop/Assignment6/Q2$ mpirun -np 8 ./Q2
Time taken by 8 processes : 0.851927 seconds
Speedup for 8 processes : 2.484324
hardik@ubuntu:~/Desktop/Assignment6/Q2$ mpirun -np 16 ./Q2
Time taken by 16 processes : 1.243893 seconds
Speedup for 16 processes : 1.790567
hardik@ubuntu:~/Desktop/Assignment6/Q2$ mpirun -np 32 ./Q2
Time taken by 32 processes : 6.018744 seconds
Speedup for 32 processes : 4.223333
hardik@ubuntu:~/Desktop/Assignment6/Q2$
```

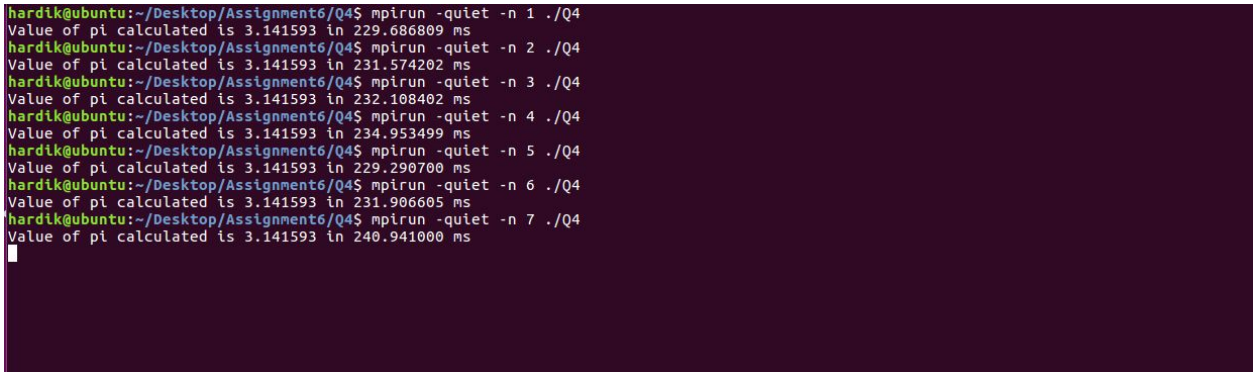
QUESTION 3 - HELLO WORLD PROGRAM - VERSION2

Given below is the output.Here “Hello world” message is sent to master process from the other 7 processes.

```
hardik@ubuntu:~/Desktop/Assignment6$ cd Q3
hardik@ubuntu:~/Desktop/Assignment6/Q3$ mpicc Q3.c -o Q3
hardik@ubuntu:~/Desktop/Assignment6/Q3$ mpirun -np 8 ./Q3
Hello world from 1
Hello world from 2
Hello world from 3
Hello world from 4
Hello world from 5
Hello world from 6
Hello world from 7
hardik@ubuntu:~/Desktop/Assignment6/Q3$
```

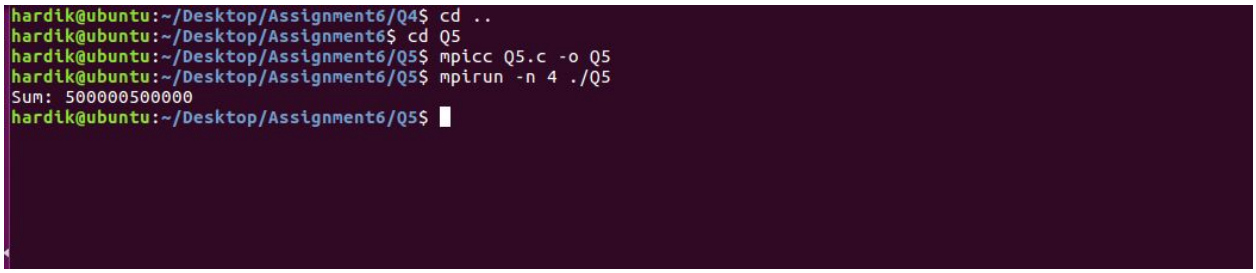
QUESTION 4 - CALCULATION OF PI

Given below is the output.Here value of pi is calculated. The time taken by different number of processes is shown in the output.



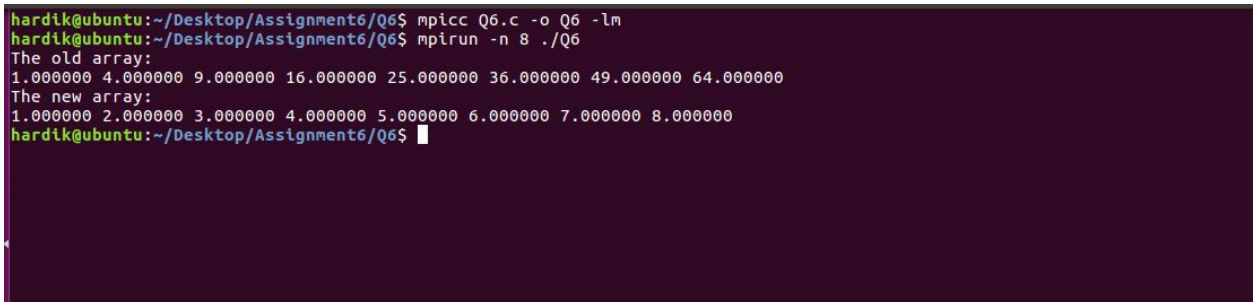
QUESTION 5 - REDUCTION OPERATION

Given below is the output.Here sum of N = 1000000, is shown in the Output.



QUESTION 6 - COLLECTIVE COMMUNICATION-SCATTER-GATHER

Given below is the output.Here the square root of the old array is calculated by scattering its elements into different processes, calculating square root there and gathering it back at the master process.

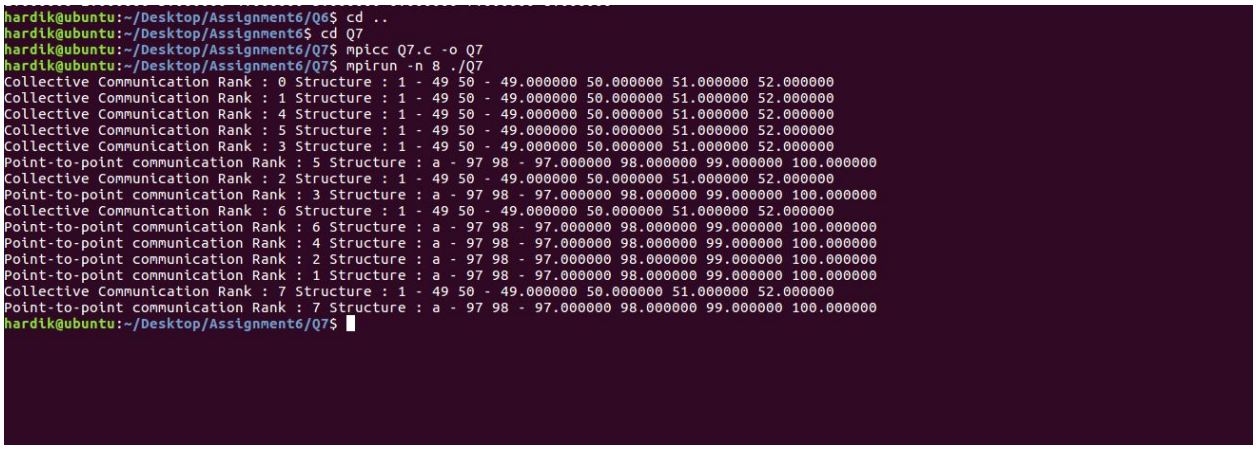


QUESTION 7 - MPI DERIVED DATATYPES

The following code snippet is used to fill the structure

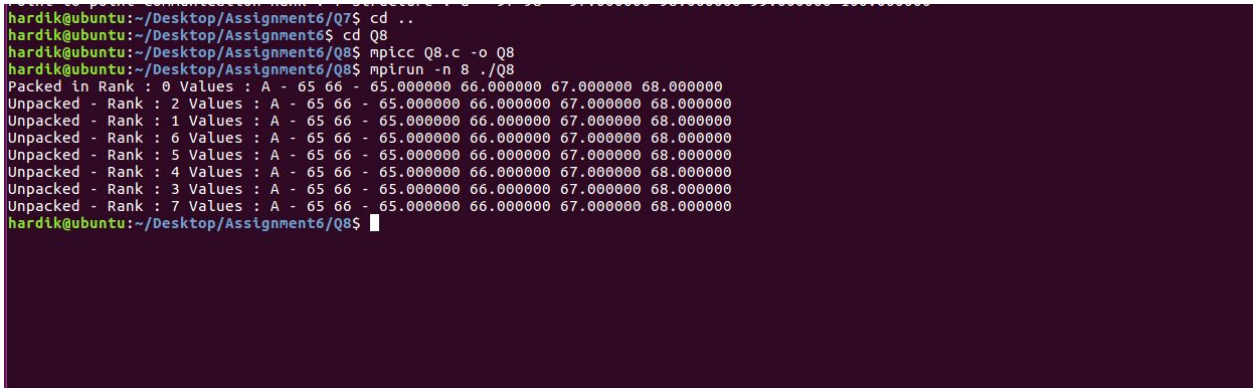
```
struct dd get_filled_struct(char key)
    {
                struct dd temp;
                 temp.c = key;
                 for(int i=0;i<4;++i)
                {
                       if(i<2)
                      {
                         temp.iA[i] = key + i;
                      }
                        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 send the datatype to the other process, whose output is prefixed by **Point-to-point communication**



QUESTION 8 - PACK AND UNPACK

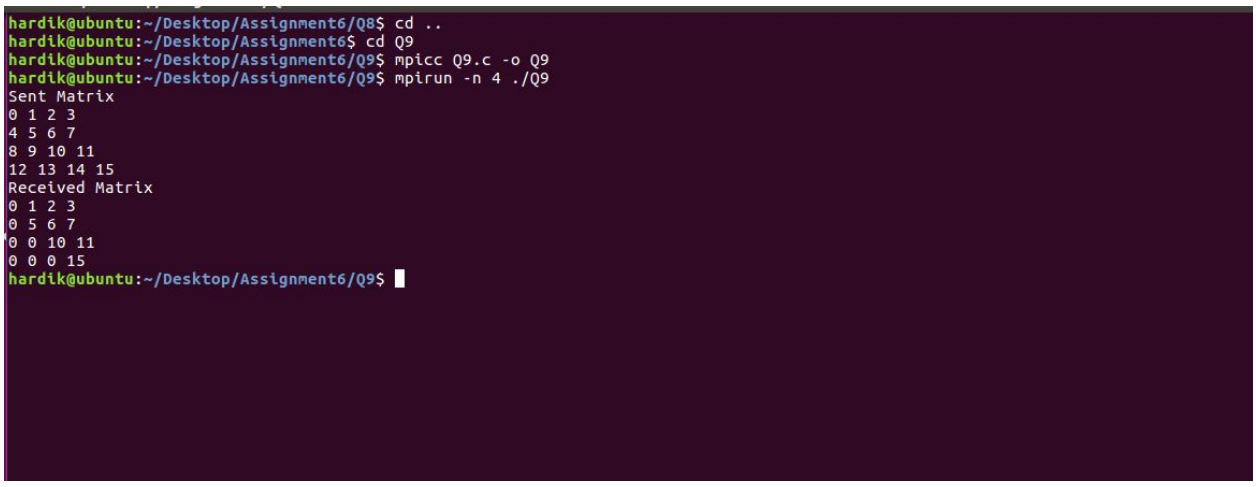
Given below is the output.Here The same problem statement in Q7, is done using the Pack and Unpack functions in MPI.



QUESTION 9 - DERIVED DATATYPE - INDEXED

Given below is the output.Here an indexed derived datatype which taken

only the upper triangle of a matrix is declared. The below shown matrix is send my 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.



QUESTION 10 - MATRIX MULTIPLICATION USING CANNON'S ALGORITHM

Given below is the output.
The cannon's algorithm was implemented. The output is shown for 3x3, 4x4 and 5x5 matrices respectively.

The matrices are filled using the following code snippet

```
for(int i = 0; i < N; ++i)
        for(int j = 0; j < N; ++j)
            arr[i][j] = i*N + j;
```

To change the dimension of the matrix do the following
1. #define N 4 : Change this macro in line 3 of the code
2. while running run : mpirun -n 25 a.out, where 25 is for a 5x5 matrix.
   or mpirun -n 16 a.out, where 16 is for a 4x4 matrix.

```
n 9 a.out
Operand matrix A
1 2 3
4 5 6
7 8 9
Operand matrix B
1 2 3
4 5 6
7 8 9
Resultant matrix C
30 36 42
66 81 96
102 126 150
```

```
n 16 a.out
Operand matrix A
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
Operand matrix B
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
Resultant matrix C
90 100 110 120
202 228 254 280
314 356 398 440
426 484 542 600
```

```
n 25 a.out
Operand matrix A
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
21 22 23 24 25
Operand matrix B
1 2 3 4 5
6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
21 22 23 24 25
Resultant matrix C
215 230 245 260 275
490 530 570 610 650
765 830 895 960 1025
1040 1130 1220 1310 1400
1315 1430 1545 1660 1775
```