

Vanessa Lobo
180905316
CSE-A
Roll no: 37

LAB-4

Q1:

```
#include <mpi.h>
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[])
{
    int rank, size;
    int fact=1, factsum;

    MPI_Init(&argc, &argv);

    MPI_Errhandler_set(MPI_COMM_WORLD, MPI_ERRORS_RETURN);
    //triggering an error
    //int errcode=MPI_Comm_rank(size, &rank);
    int errcode=MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    for(int i=1; i<=rank+1; i++){
        fact=fact*i;
    }

    MPI_Scan(&fact, &factsum, 1, MPI_INT, MPI_SUM, MPI_COMM_WORLD);

    if(rank==size-1){
        printf("Rank: %d, The sum of factorial is %d\n", rank, factsum);
    }

    if(errcode!=MPI_SUCCESS){
        int errclass;
        char errstring[1024];
        int len;
        MPI_Error_class(errcode, &errclass);
        MPI_Error_string(errcode, errstring, &len);
        printf("Error class: %d\n", errclass);
        printf("Error string: %s\n", errstring);
    }

    MPI_Finalize();
    return 0;
}
```

```
student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpicc -o q1 q1.c
student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpicc -o q1 q1.c
student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpirun -n 4 ./q1
Rank: 3, The sum of factorial is 33
student@lplab-Lenovo-Product:~/Desktop/180905316/4$
```

After triggering error

```
student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpicc -o q1 q1.c
student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpirun -n 4 ./q1
Error class: 5
Error string: Invalid communicator, error stack:
PMPI_Comm_rank(110): MPI_Comm_rank(comm=0x0, rank=0x7ffc9c855d0) failed
PMPI_Comm_rank(68): Invalid communicator
Error class: 5
Error string: Invalid communicator, error stack:
PMPI_Comm_rank(110): MPI_Comm_rank(comm=0x0, rank=0x7ffd42c5d250) failed
PMPI_Comm_rank(68): Invalid communicator
Error class: 5
Error string: Invalid communicator, error stack:
PMPI_Comm_rank(110): MPI_Comm_rank(comm=0x0, rank=0x7ffc33a738f0) failed
PMPI_Comm_rank(68): Invalid communicator
Error class: 5
Error string: Invalid communicator, error stack:
PMPI_Comm_rank(110): MPI_Comm_rank(comm=0x0, rank=0x7fff10a775d0) failed
PMPI_Comm_rank(68): Invalid communicator
student@lplab-Lenovo-Product:~/Desktop/180905316/4$
```

Q2:

```
#include <mpi.h>
#include <stdio.h>
#include <stdlib.h>
```

```
int main(int argc, char *argv[])
{
```

```
    int rank, size;
    MPI_Init(&argc, &argv);
    MPI_Errhandler_set(MPI_COMM_WORLD, MPI_ERRORS_RETURN);
```

```
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
```

```
    float x, y, area, pi;
```

```
    x = (float)(rank+1)/size;
```

```
    y = 4.f/(1+x*x);
```

```
    area = (1/(float)size)*y;
```

```
    //triggering an error
```

```
    //int errcode=MPI_Reduce(&area, &pi, 1, MPI_FLOAT, MPI_SUM, 6,
MPI_COMM_WORLD);
```

```
    int errcode=MPI_Reduce(&area, &pi, 1, MPI_FLOAT, MPI_SUM, 0,
MPI_COMM_WORLD);
```

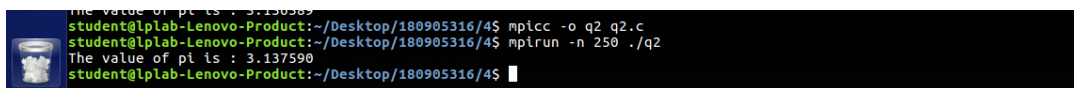
```
    if (rank==0){
        printf("The value of pi is : %f\n", pi);
    }
```

```

    if(errcode!=MPI_SUCCESS){
        int errclass;
        char errstring[1024];
        int len;
        MPI_Error_class(errcode, &errclass);
        MPI_Error_string(errcode, errstring, &len);
        printf("Error class: %d\n", errclass);
        printf("Error string: %s\n", errstring);
    }

    MPI_Finalize();
    return 0;
}

```

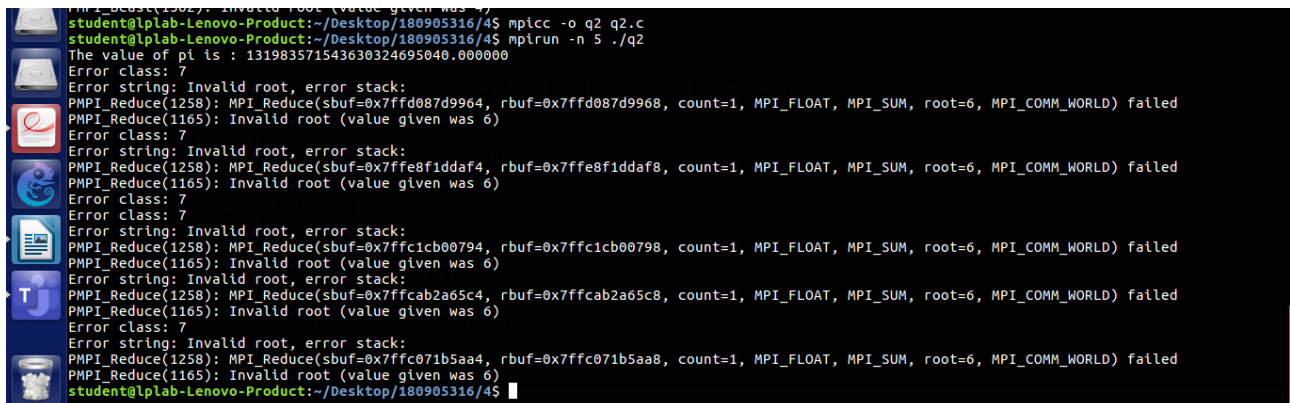


```

The value of pi is : 3.137590
student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpicc -o q2 q2.c
student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpirun -n 250 ./q2
The value of pi is : 3.137590
student@lplab-Lenovo-Product:~/Desktop/180905316/4$

```

After triggering error



```

PMPI_Reduce(1165): Invalid root (value given was 6)
Error class: 7
Error string: Invalid root, error stack:
PMPI_Reduce(1258): MPI_Reduce(sbuf=0x7ffd087d9964, rbuf=0x7ffd087d9968, count=1, MPI_FLOAT, MPI_SUM, root=6, MPI_COMM_WORLD) failed
PMPI_Reduce(1165): Invalid root (value given was 6)
Error class: 7
Error string: Invalid root, error stack:
PMPI_Reduce(1258): MPI_Reduce(sbuf=0x7ffe8f1ddaf4, rbuf=0x7ffe8f1ddaf8, count=1, MPI_FLOAT, MPI_SUM, root=6, MPI_COMM_WORLD) failed
PMPI_Reduce(1165): Invalid root (value given was 6)
Error class: 7
Error string: Invalid root, error stack:
PMPI_Reduce(1258): MPI_Reduce(sbuf=0x7ffc1cb00794, rbuf=0x7ffc1cb00798, count=1, MPI_FLOAT, MPI_SUM, root=6, MPI_COMM_WORLD) failed
PMPI_Reduce(1165): Invalid root (value given was 6)
Error string: Invalid root, error stack:
PMPI_Reduce(1258): MPI_Reduce(sbuf=0x7ffcab2a65c4, rbuf=0x7ffcab2a65c8, count=1, MPI_FLOAT, MPI_SUM, root=6, MPI_COMM_WORLD) failed
PMPI_Reduce(1165): Invalid root (value given was 6)
Error class: 7
Error string: Invalid root, error stack:
PMPI_Reduce(1258): MPI_Reduce(sbuf=0x7ffc071b5aa4, rbuf=0x7ffc071b5aa8, count=1, MPI_FLOAT, MPI_SUM, root=6, MPI_COMM_WORLD) failed
PMPI_Reduce(1165): Invalid root (value given was 6)
student@lplab-Lenovo-Product:~/Desktop/180905316/4$

```

Q3:

```

#include <mpi.h>
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{
    int rank, size;
    MPI_Init(&argc, &argv);
    MPI_Errhandler_set(MPI_COMM_WORLD, MPI_ERRORS_RETURN);

    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    int mat[3][3];
    int row[3];
    int ele;
    int count=0;

```

```

int sumcount;

if(rank==0){
    printf("Enter elements of a 3x3 matrix\n");
    for(int i=0; i<3; i++){
        for(int j=0; j<3; j++){
            scanf("%d", &mat[i][j]);
        }
    }
    printf("Enter element to be searched\n");
    scanf("%d", &ele);
}

//triggering an error
//int errcode=MPI_Bcast(&ele, 1, MPI_INT, 4, MPI_COMM_WORLD);
MPI_Bcast(&ele, 1, MPI_INT, 0, MPI_COMM_WORLD);
MPI_Scatter(&mat, 3, MPI_INT, &row, 3, MPI_INT, 0, MPI_COMM_WORLD);

for(int i=0; i<3; i++){
    if(row[i]==ele){
        count++;
    }
}

printf("Rank: %d, occurrence count: %d\n", rank, count);
MPI_Reduce(&count, &sumcount, 1, MPI_INT, MPI_SUM, 0, MPI_COMM_WORLD);
if(rank==0){
    printf("Rank: %d, Total occurrence count: %d\n", rank, sumcount);
}
if(errcode!=MPI_SUCCESS){
    int errclass;
    char errstring[1024];
    int len;
    MPI_Error_class(errcode, &errclass);
    MPI_Error_string(errcode, errstring, &len);
    printf("Error class: %d\n", errclass);
    printf("Error string: %s\n", errstring);
}
MPI_Finalize();
return 0;
}

```

```

student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpicc -o q3 q3.c
student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpirun -n 3 ./q3
Enter elements of a 3x3 matrix
1 2 1
3 2 1
3 4 1
Enter element to be searched
1
Rank: 0, occurrence count: 2
Rank: 0, Total occurrence count: 4
Rank: 1, occurrence count: 1
Rank: 2, occurrence count: 1
student@lplab-Lenovo-Product:~/Desktop/180905316/4$

```

After triggering error:

```
PMPI_Bcast(1562): Invalid root (value given was 4)
student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpicc -o q3 q3.c
student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpirun -n 3 ./q3
Enter elements of a 3x3 matrix
1 2 3
4 5 6
1 2 3
Enter element to be searched
1
Rank: 0, occurrence count: 1
Rank: 0, Total occurrence count: 2
Error class: 7
Error string: Invalid root, error stack:
PMPI_Bcast(1600): MPI_Bcast(buf=0x7ffffc180e19c, count=1, MPI_INT, root=4, MPI_COMM_WORLD) failed
PMPI_Bcast(1562): Invalid root (value given was 4)
Rank: 1, occurrence count: 0
Error class: 7
Error string: Invalid root, error stack:
PMPI_Bcast(1600): MPI_Bcast(buf=0x7ffe9c26882c, count=1, MPI_INT, root=4, MPI_COMM_WORLD) failed
PMPI_Bcast(1562): Invalid root (value given was 4)
Rank: 2, occurrence count: 1
Error class: 7
Error string: Invalid root, error stack:
PMPI_Bcast(1600): MPI_Bcast(buf=0x7ffe9c26882c, count=1, MPI_INT, root=4, MPI_COMM_WORLD) failed
PMPI_Bcast(1562): Invalid root (value given was 4)
student@lplab-Lenovo-Product:~/Desktop/180905316/4$
```

Q4:

```
#include <mpi.h>
#include <stdio.h>
#include <stdlib.h>
```

```
int main(int argc, char *argv[])
{
    int rank, size;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    int mat[4][4];
    int res[4][4];

    int row[4];
    int buf[4];

    if(rank==0){
        printf("Enter elements of a 4x4 matrix\n");
        for(int i=0; i<4; i++){
            for(int j=0; j<4; j++){
                scanf("%d", &mat[i][j]);
            }
        }
    }
}
```

```

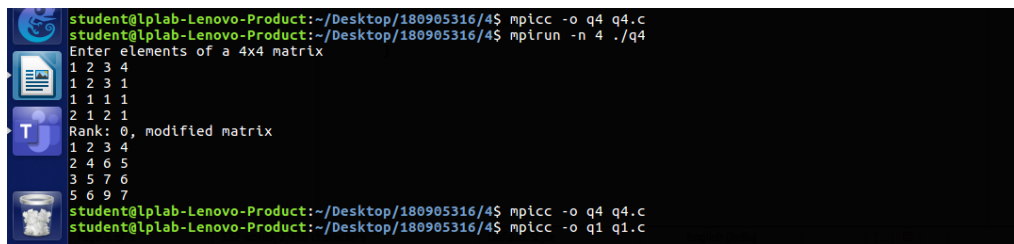
}

MPI_Scatter(&mat, 4, MPI_INT, &row, 4, MPI_INT, 0, MPI_COMM_WORLD);
MPI_Scan(&row, &buf, 4, MPI_INT, MPI_SUM, MPI_COMM_WORLD);
MPI_Gather(&buf, 4, MPI_INT, &res, 4, MPI_INT, 0, MPI_COMM_WORLD);

if(rank==0){
    printf("Rank: %d, modified matrix\n", rank);

    for(int i=0; i<4; i++){
        for(int j=0; j<4; j++){
            printf("%d ", res[i][j]);
        }
        printf("\n");
    }
}
MPI_Finalize();
return 0;
}

```



A terminal window showing the compilation and execution of an MPI program. The user is at a prompt on a system named 'student@lplab-Lenovo-Product'. The terminal shows the following sequence of commands and output:

```

student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpicc -o q4 q4.c
student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpirun -n 4 ./q4
Enter elements of a 4x4 matrix
1 2 3 4
1 2 3 1
1 1 1 1
2 1 2 1
Rank: 0, modified matrix
1 2 3 4
2 4 6 5
3 5 7 6
5 6 9 7
student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpicc -o q4 q4.c
student@lplab-Lenovo-Product:~/Desktop/180905316/4$ mpicc -o q1 q1.c

```

The output shows that the program was compiled successfully, executed with 4 processes, and produced the expected modified matrix output for rank 0.