PPL Week-4

Rachit Takate – 230962294 – B54

I. Write a MPI program using N processes to find 1! + 2! + ….. + N!. Use scan. Also, handle different errors using error handling routines.

```c
#include <stdio.h>
#include <math.h>
#include <mpi.h>
#define MCW MPI_COMM_WORLD

int errHandler(int errCode, int rank) {
    char errStr[MPI_MAX_ERROR_STRING];
    int errClass, errLen;

    switch (errCode) {
    case MPI_SUCCESS:
        return 0;
    default:
        MPI_Error_class(errCode, &errClass);
        MPI_Error_string(errCode, errStr, &errLen);
        printf("P%d: %s\n", rank, errStr);
        return 1;
    }
}

int main(int argc, char *argv[]) {
    int rank, size, len, sum, fact = 1;
    int stat;
    MPI_Init(&argc, &argv);

    MPI_Comm_rank(MCW, &rank);
    MPI_Comm_size(MCW, &size);
    MPI_Comm_set_errhandler(MCW, MPI_ERRORS_RETURN);

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

    stat = MPI_Scan(&fact, &sum, 1, MPI_INT, MPI_SUM, MCW);
    errHandler(stat, rank);

    if(!rank)
        printf("Rachit 230962294\n");
    else if(rank == size - 1)
        printf("Sum of %d factorials: %d\n", size, sum);

    MPI_Finalize();
    return 0;
}
```

Output


```
(dse) mca@computinglab26-26:~/230962294/PP Lab/Week4$ mpicc Q1.c -o Q1
(dse) mca@computinglab26-26:~/230962294/PP Lab/Week4$ mpirun -np 5 ./Q1
Sum of 5 factorials: 34
Rachit 230962294
```

II. Write a MPI program to read a 3 X 3 matrix. Enter an element to be searched in the root process. Find the number of occurrences of this element in the matrix using three processes.

```c
#include <stdio.h>
#include <mpi.h>
#define MCW MPI_COMM_WORLD

int main(int argc, char* argv[]) {
    int rank, val, final, count = 0;
    int mat[3][3], arr[3];
    int stat;
    MPI_Init(&argc, &argv);

    MPI_Comm_rank(MCW, &rank);

    if(!rank) {
        for(int i = 0; i < 3; i++)
            for(int j = 0; j < 3; j++)
                scanf("%d", &mat[i][j]);

        scanf("%d", &val);
    }

    MPI_Bcast(&val, 1, MPI_INT, 0, MCW);
    MPI_Scatter(mat, 3, MPI_INT, arr, 3, MPI_INT, 0, MCW);

    for(int i = 0; i < 3; i++)
        if(arr[i] == val)
            count++;

    MPI_Reduce(&count, &final, 1, MPI_INT, MPI_SUM, 0, MCW);

    if(!rank) {
        printf("Amount: %d\n", final);
        printf("Rachit 230962294\n");
    }

    MPI_Finalize();
    return 0;
}
```

Output

III. Write a MPI program to read 4 X 4 matrix and display the following output using four processes.

```
I/p matrix:          O/p matrix:
1  2  3  4           1  2  3  4
1  2  3  1           2  4  6  5
1  1  1  1           3  5  7  6
2  1  2  1           5  6  9  7
```

```c
#include <stdio.h>
#include <mpi.h>
#define MCW MPI_COMM_WORLD

int main(int argc, char* argv[]) {
    int rank;
    int mat[4][4], arr[4], sumArr[4];
    int stat;
    MPI_Init(&argc, &argv);
```

```
        MPI_Comm_rank(MCW, &rank);

        if(!rank)
            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, arr, 4, MPI_INT, 0, MCW);
        MPI_Scan(arr, sumArr, 4, MPI_INT, MPI_SUM, MCW);

        MPI_Gather(sumArr, 4, MPI_INT, mat, 4, MPI_INT, 0, MPI_COMM_WORLD);

        if(!rank) {
            print("Result:\n");
            for(int i = 0; i < 4; i++) {
                for(int j = 0; j < 4; j++)
                    printf("%d ", mat[i][j]);
                printf("\n");
            }
            printf("Rachit 230962294\n");
        }

        MPI_Finalize();
        return 0;
}
```

Output

IV. Write a MPI program to read a word of length N. Using N processes including the root get output word with the pattern as shown in example. Display the resultant output word in the root. Input - PCAP & output - PCCAAAPPPP

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <mpi.h>
#define MCW MPI_COMM_WORLD

int main(int argc, char* argv[]) {
    int rank, size, len;
    char lett, inp[32], *subs, *final;
    int stat;
    MPI_Init(&argc, &argv);

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

```c
    if(!rank) {
        scanf("%s", inp);
        len = strlen(inp);
        final = calloc(size * len, sizeof(char));
    }

    MPI_Bcast(&len, 1, MPI_INT, 0, MCW);
    MPI_Scatter(inp, 1, MPI_CHAR, &lett, 1, MPI_CHAR, 0, MCW);

    subs = calloc(len, sizeof(char));
    for(int i = 0; i < len; i++)
        if(i <= rank)
            subs[i] = lett;

    MPI_Gather(subs, len, MPI_CHAR, final, len, MPI_CHAR, 0, MCW);

    if(!rank) {
        printf("Res: ");
        for(int i = 0; i < size * len; i++)
            final[i] ? printf("%c", final[i]) : 0;

        printf("\nRachit 230962294\n");
    }

    MPI_Finalize();
    return 0;
}
```

Output