Date : 29/01/2026

PPL WEEK4 SUBMISSION

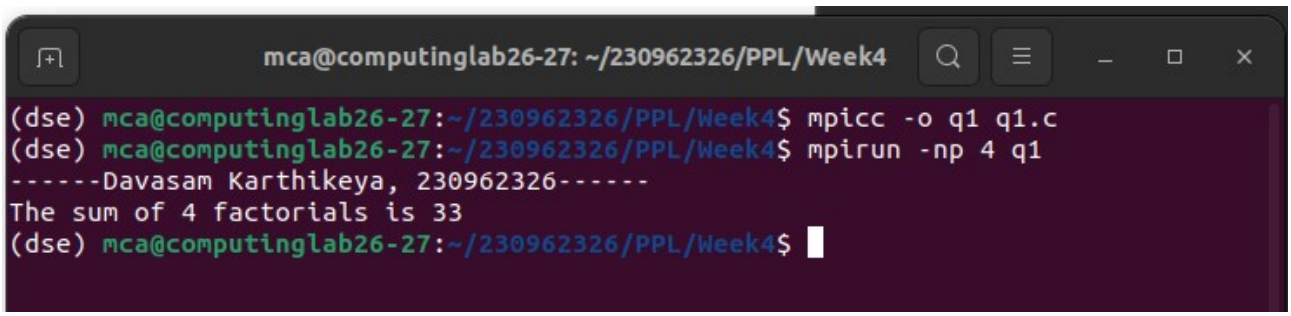
Name: Davasam Karthikeya Section: AIMLB Reg No: 230962326

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

Code:-

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

void mpi_error_to_str(int err_code, int rank) {
    if(err_code == MPI_SUCCESS)return;
    char err_str[MPI_MAX_ERROR_STRING];
    int result_len;
    MPI_Error_string(err_code, err_str, &result_len);
    printf("MPI Error, from %d: %s \n", rank, err_str);
}

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

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

    if(rank == 0)printf("------Davasam Karthikeya, 230962326------\n");

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

    int err = MPI_Scan(&fact, &recvBuf, 1, MPI_INT, MPI_SUM, MPI_COMM_WORLD);

    mpi_error_to_str(err, rank);

    if(rank == size-1)
        printf("The sum of %d factorials is %d \n", size, recvBuf);

    MPI_Finalize();
    return 0;
}
```

Output:-



Date : 29/01/2026

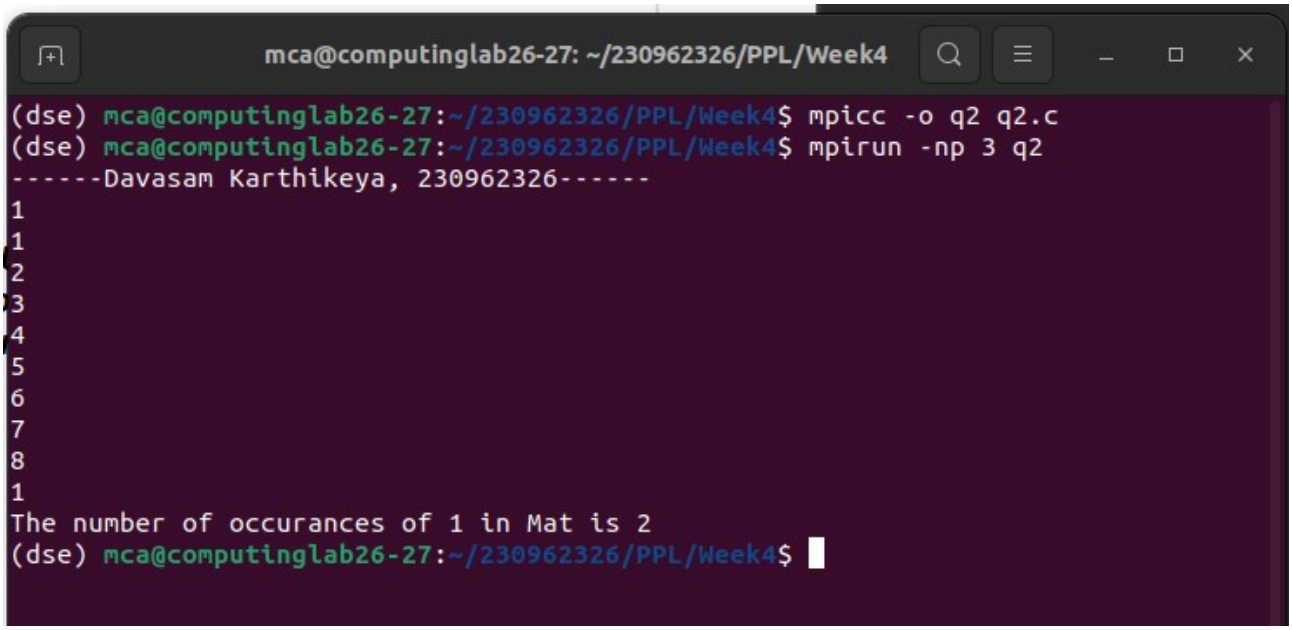
2. Write a MPI program to read a 3 X 3 matrix. Enter an element to be searched in the root process. Find the number of occurrences of this element in the matrix using three processes.

Code:-

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

void print_mpi_error(int err_code, int rank) {
    if(err_code == MPI_SUCCESS)return;
    char err_str[MPI_MAX_ERROR_STRING];
    int result_len;
    MPI_Error_string(err_code, err_str, &result_len);
    printf("MPI Error, from %d: %s \n", rank, err_str);
}

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

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

    int Mat[3][3], searchElem;
    if(rank == 0){
        printf("------Davasam Karthikeya, 230962326------\n");
        for(int i = 0; i < 3; i++){
            for(int j = 0; j < 3; j++){
                scanf("%d", &Mat[i][j]);
            }
        }
        scanf("%d", &searchElem);
    }
    int err = MPI_Bcast(&searchElem, 1, MPI_INT, 0, MPI_COMM_WORLD);
    print_mpi_error(err, rank);
    err = MPI_Scatter(&Mat, 3, MPI_INT, &Mat[0], 3, MPI_INT, 0, MPI_COMM_WORLD);
    print_mpi_error(err, rank);

    int count = 0, recvBuf;
    for(int i = 0; i< 3; i++){
        if(Mat[0][i] == searchElem)count++;
    }

    MPI_Reduce(&count, &recvBuf, 1, MPI_INT, MPI_SUM, 0, MPI_COMM_WORLD);

    if(rank == 0){
        printf("The number of occurances of %d in Mat is %d \n",
searchElem ,recvBuf);
    }

    MPI_Finalize();
    return 0;
}
```

Date : 29/01/2026

Output:-



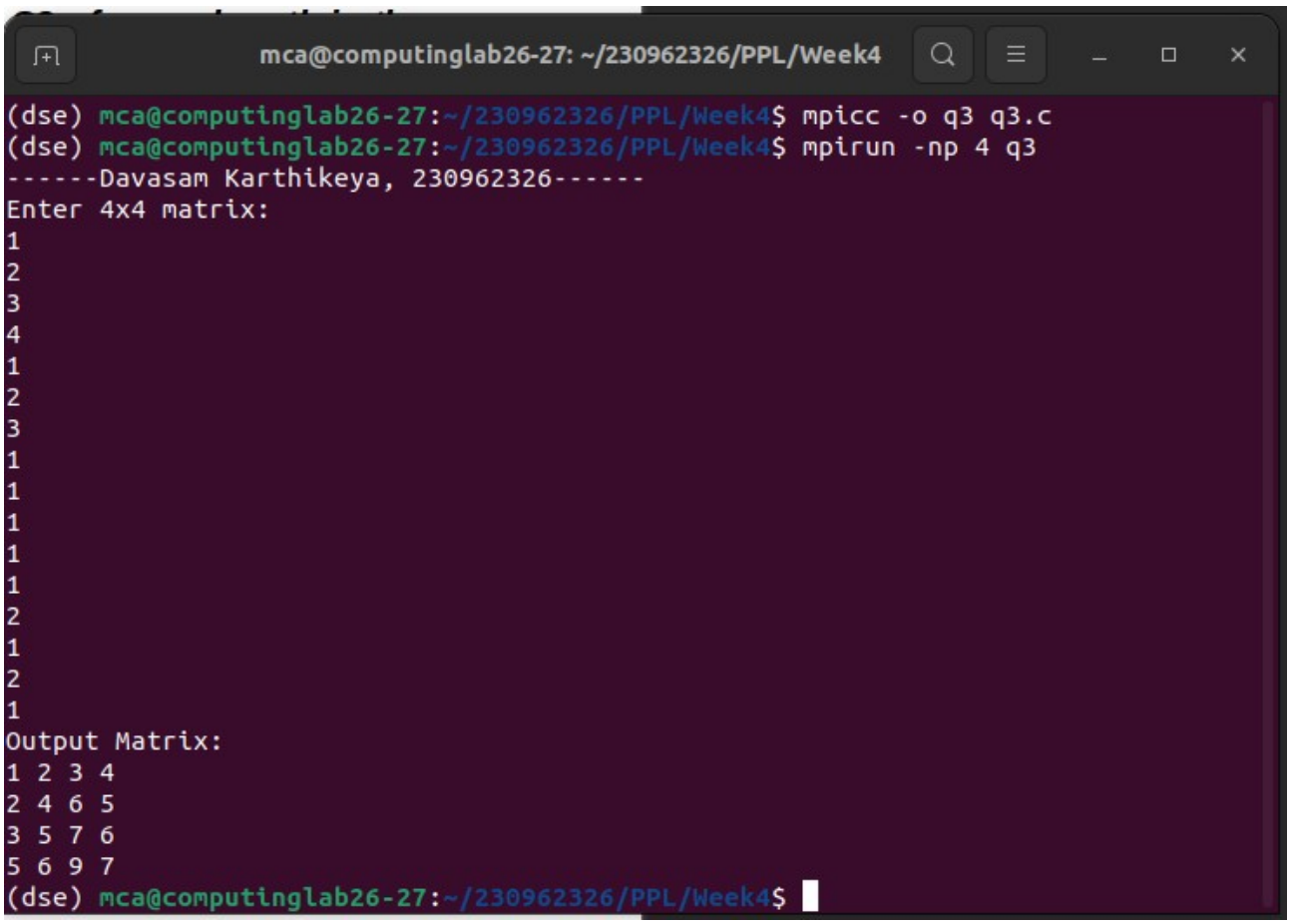
3. Write a MPI program to read 4 X 4 matrix and display the following output using four processes.

Code:-

```
#include <stdio.h>
#include <mpi.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 matrix[4][4];
    int row[4], scanRow[4];
    int result[4][4];

    if (rank == 0) {
        printf("------Davasam Karthikeya, 230962326------\n");
        printf("Enter 4x4 matrix:\n");
        for (int i = 0; i < 4; i++)
            for (int j = 0; j < 4; j++)
                scanf("%d", &matrix[i][j]);
    }

    MPI_Scatter(matrix, 4, MPI_INT, row, 4, MPI_INT, 0, MPI_COMM_WORLD);

    MPI_Scan(row, scanRow, 4, MPI_INT, MPI_SUM, MPI_COMM_WORLD);

    MPI_Gather(scanRow, 4, MPI_INT, result, 4, MPI_INT, 0, MPI_COMM_WORLD);

    if (rank == 0) {
        printf("Output Matrix:\n");
        for (int i = 0; i < 4; i++) {
```

```
            for (int j = 0; j < 4; j++)
                printf("%d ", result[i][j]);
            printf("\n");
        }
    }

    MPI_Finalize();
    return 0;
}
```

Output:-



4. 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.

Code:-

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

void print_mpi_error(int err_code, int rank) {
    if(err_code == 0)return;
    char err_str[MPI_MAX_ERROR_STRING];
    int result_len;
    MPI_Error_string(err_code, err_str, &result_len);
```

```
    printf("MPI Error, from %d: %s \n", rank, err_str);
}

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);

    char strIn[size + 1], ch;
    if (rank == 0) {
        printf("------Davasam Karthikeya, 230962326------\n");
        scanf("%s", strIn);
    }

    MPI_Scatter(strIn, 1, MPI_CHAR, &ch, 1, MPI_CHAR, 0, MPI_COMM_WORLD);

    char sendBuf[size];
    for (int i = 0; i < size; i++) {
        if (i < rank + 1)
            sendBuf[i] = ch;
        else
            sendBuf[i] = '\0';
    }

    char recvBuf[size * size];

    MPI_Gather(sendBuf, size, MPI_CHAR, recvBuf, size, MPI_CHAR, 0,
MPI_COMM_WORLD);

    if (rank == 0) {
        printf("Processed String: ");
        for (int i = 0; i < size * size; i++) {
            if (recvBuf[i] != '\0')
                printf("%c", recvBuf[i]);
        }
        printf("\n");
    }

    MPI_Finalize();
    return 0;
}
```

Output:-