**Lab 3: Collective Communication in MPI**

1.)

#include <mpi.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

void 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);

long fact;

long i, n;

long rec;

long arr[100], facts[100];

if (rank == 0)

{

n = size;

printf("Enter the numbers: ");

for (i = 0; i < n; ++i)

{

scanf("%ld", &arr[i]);

}

}

MPI\_Scatter(arr, 1, MPI\_LONG, &rec, 1, MPI\_LONG, 0, MPI\_COMM\_WORLD);

printf("Process [%d] received = %ld.\n", rank, rec);

fact = 1;

for (i = 2; i <= rec; ++i)

{

fact \*= i;

}

MPI\_Gather(&fact, 1, MPI\_LONG, facts, 1, MPI\_LONG, 0, MPI\_COMM\_WORLD);

if (rank == 0)

{

printf("Sum of factorials = ");

long sum = 0;

for (i = 0; i < n; ++i)

{

sum += facts[i];

printf("%ld %s", facts[i], (i != n-1)?"+ ":" ");

}

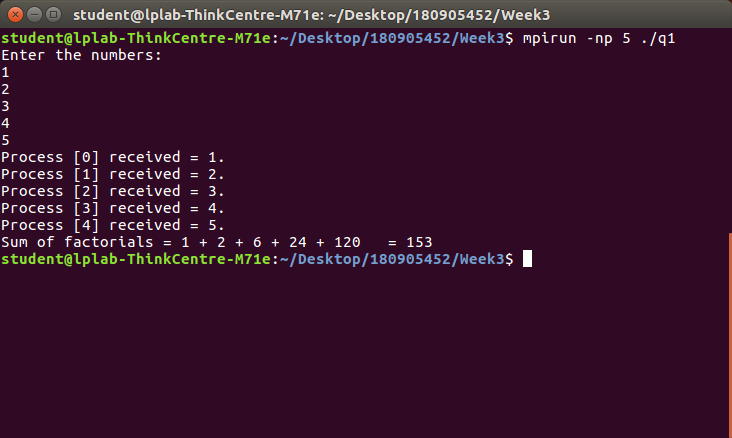
printf(" = %ld\n", sum);

}

MPI\_Finalize();

}

Output:



2.)

#include <mpi.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.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);

long fact;

long i, n;

long rec;

long arr[100], facts[100];

if (rank == 0)

{

n = size;

printf("Enter the numbers: ");

for (i = 0; i < n; ++i)

{

scanf("%d", &arr[i]);

}

}

MPI\_Scatter(arr, 1, MPI\_LONG, &rec, 1, MPI\_LONG, 0, MPI\_COMM\_WORLD);

printf("Process [%d] received = %d.\n", rank, rec);

fact = 1;

for (i = 2; i <= rec; ++i)

{

fact \*= i;

}

MPI\_Gather(&fact, 1, MPI\_LONG, facts, 1, MPI\_LONG, 0, MPI\_COMM\_WORLD);

if (rank == 0)

{

printf("Sum of factorials = ");

long sum = 0;

for (i = 0; i < n; ++i)

{

sum += facts[i];

printf("%ld %s", facts[i], (i != n-1)?"+ ":" ");

}

printf(" = %ld\n", sum);

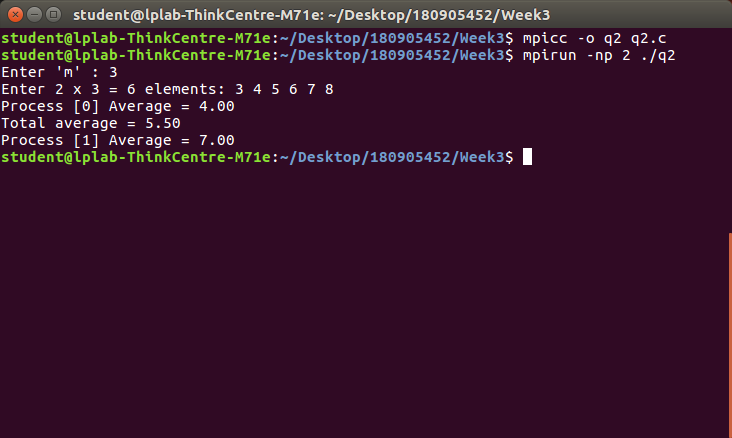
}

MPI\_Finalize();

return 0;

}

Output:



3.)

#include <mpi.h>

#include <stdio.h>

#include <string.h>

int main(int argc, char\* argv[])

{

int rank,size;

int count = 0;

int b[100] = {0};

int i, n, l;

char str[100], c[100];

MPI\_Init(&argc, &argv);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);

if (rank == 0)

{

n = size;

printf("Enter the string: ");

scanf("%s", str);

l = strlen(str) / n;

}

MPI\_Bcast(&l, 1, MPI\_INT, 0, MPI\_COMM\_WORLD);

MPI\_Scatter(str, l, MPI\_CHAR, c, l, MPI\_CHAR, 0, MPI\_COMM\_WORLD);

count = 0;

for (i = 0; i < l; ++i)

{

if(c[i] =='a' || c[i] == 'e' || c[i] == 'i' || c[i] == 'o' || c[i] == 'u')

continue;

count+=1;

}

printf("Process %d Count = %d\n", rank, count);

fflush(stdout);

MPI\_Gather(&count, 1, MPI\_INT, b, 1, MPI\_INT, 0, MPI\_COMM\_WORLD);

if (rank == 0)

{

int tcount = 0;

for (i = 0; i < n; i++)

tcount += b[i];

printf("Total non vowels = %d\n", tcount);

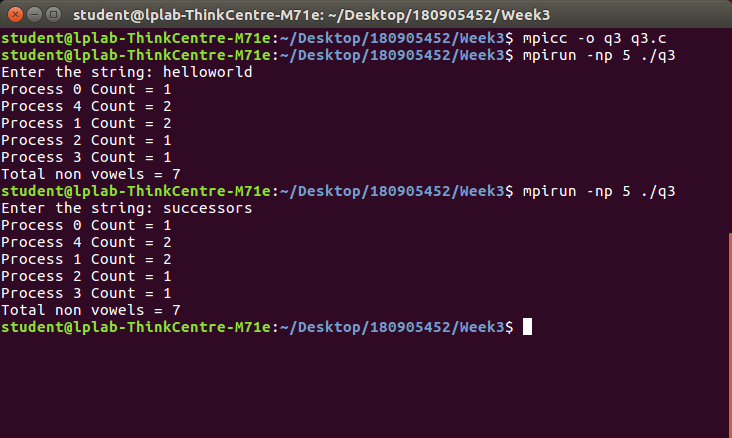
fflush(stdout);

}

MPI\_Finalize();

}

Output:



4.)

#include <mpi.h>

#include <stdio.h>

#include <string.h>

int main(int argc, char\* argv [])

{

int rank, size;

float avg = 0;

char b[100], str1[100], str2[100], c1[100], c2[100], concatted[100];

int i, j, m;

MPI\_Init(&argc, &argv);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);

if (rank == 0)

{

printf("Enter string 1: ");

scanf("%s", str1);

printf("Enter string 2: ");

scanf("%s", str2);

m = strlen(str1) / size;

}

MPI\_Bcast(&m, 1, MPI\_INT, 0, MPI\_COMM\_WORLD);

MPI\_Scatter(str1, m, MPI\_CHAR, c1, m, MPI\_CHAR, 0, MPI\_COMM\_WORLD);

MPI\_Scatter(str2, m, MPI\_CHAR, c2, m, MPI\_CHAR, 0, MPI\_COMM\_WORLD);

int t = 0;

for (t = 0; t <= 2 \* m; t += 2)

{

concatted[t] = c1[t/2];

concatted[t+1] = c2[t/2];

}

concatted[2\*m] = '\0';

MPI\_Gather(concatted, 2\*m, MPI\_CHAR, b, 2\*m, MPI\_CHAR, 0, MPI\_COMM\_WORLD);

if (rank == 0)

{

b[m\*size\*2] = '\0';

printf("Concatted: " b);

}

MPI\_Finalize();

}

Output:

