SANSKAAR PATNI
180905134 CSE C-23
PCAP LAB 3

1. CODE

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

int main(int argc,char *argv[]){
	int rank,size,N,A[10],B[10],c,i,sum=0,fact=1;
	MPI_Init(&argc, &argv);
	MPI_Comm_rank(MPI_COMM_WORLD, &rank);
	MPI_Comm_size(MPI_COMM_WORLD, &size);
	if (rank == 0)
	{
		N=size;
		fprintf(stdout, "Input %d numbers: \n", size);
		fflush(stdout);
		for (int i = 0;i < size;i++)
			scanf("%d", &A[i]);
	}
	MPI_Scatter(A,1,MPI_INT,&c,1,MPI_INT,0,MPI_COMM_WORLD);
	fprintf(stdout,"In process %d, Recieved %d\n",rank,c);
	fflush(stdout);
	for(i=1;i<=c;i++){
		fact*=i;
	}
	MPI_Gather(&fact,1,MPI_INT,B,1,MPI_INT,0,MPI_COMM_WORLD);
	if(rank==0){
		fprintf(stdout,"In process Factorial Array %d:\n",rank);
		fflush(stdout);
		for(i=0;i<N;i++)
		{
			fprintf(stdout,"%d ",B[i]);
			sum+=B[i];
			fflush(stdout);
		}
			fprintf(stdout,"\nSum of all factorials of original array elements is
%d\n",sum);
			fflush(stdout);
	}
	MPI_Finalize();
	return 0;
}
```

SCREENSHOT

```
student@selab-19: ~/180905134/Lab3
student@selab-19:~/180905134/Lab3$ mpicc -o first first.c
student@selab-19:~/180905134/Lab3$ mpirun -n 4 ./first
Input 4 numbers:
1
2
3
4
In process 0, Recieved 1
In process Factorial Array 0:
1 2 6 24
Sum of all factorials of original array elements is 33
In process 1, Recieved 2
In process 2, Recieved 3
In process 3, Recieved 4
student@selab-19:~/180905134/Lab3$
```

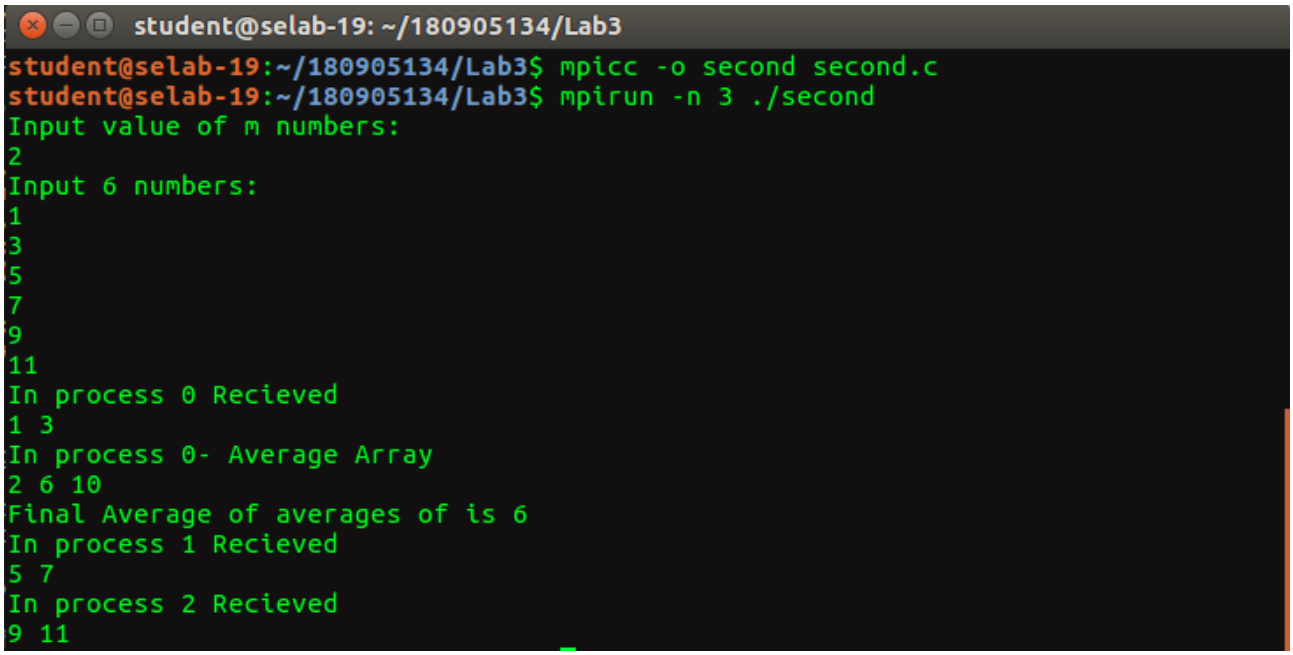
2. CODE

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

int main(int argc,char *argv[]){
        int rank,size,N,m,A[50],B[50],X[10],i,finalAvg=0,avg=0,sum=0;
        MPI_Init(&argc, &argv);
        MPI_Comm_rank(MPI_COMM_WORLD, &rank);
        MPI_Comm_size(MPI_COMM_WORLD, &size);
        if (rank == 0)
        {
                N=size;
                fprintf(stdout, "Input value of m numbers: \n");
                fflush(stdout);
                scanf("%d",&m);
                fprintf(stdout, "Input %d numbers: \n", size*m);
                fflush(stdout);
                for (int i = 0;i < size*m;i++)
                        scanf("%d", &A[i]);
        }
        MPI_Bcast(&m,1,MPI_INT,0,MPI_COMM_WORLD);
        MPI_Scatter(A,m,MPI_INT,X,m,MPI_INT,0,MPI_COMM_WORLD);
        fprintf(stdout,"In process %d Recieved \n",rank);
        for (int i = 0;i < m;i++)
                        printf("%d ",X[i]);
        printf("\n");
        fflush(stdout);
        for(i=0;i<m;i++){
                sum+=X[i];
        }
        avg=sum/m;
        MPI_Gather(&avg,1,MPI_INT,B,1,MPI_INT,0,MPI_COMM_WORLD);
        if(rank==0){
                fprintf(stdout,"In process %d- Average Array\n",rank);
                fflush(stdout);
                sum=0;
                for(i=0;i<N;i++)
```

```
		{
			fprintf(stdout,"%d ",B[i]);
			sum+=B[i];
			fflush(stdout);
		}
		finalAvg=sum/N;
		fprintf(stdout,"\nFinal Average of averages of is %d\n",finalAvg);
		fflush(stdout);
	}
	MPI_Finalize();
	return 0;
}
```

SCREENSHOT



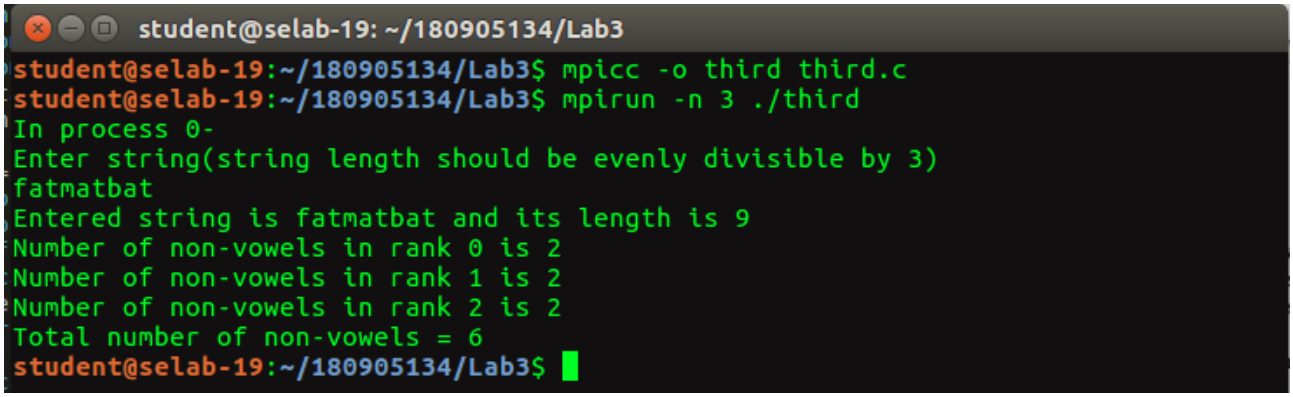
3.CODE

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

int main(int argc,char *argv[])
{
	int rank,size,N,sum=0,len,b[10];
	char arr1[100],arr2[10];
	MPI_Init(&argc,&argv);
	MPI_Comm_rank(MPI_COMM_WORLD,&rank);
	MPI_Comm_size(MPI_COMM_WORLD,&size);
	MPI_Status status;
	if(rank==0)
	{
		N=size;
		fprintf(stdout,"In process %d-\n",rank);
		fprintf(stdout, "Enter string(string length should be evenly divisible by %d)\n",size);
```

```
                fflush(stdout);
                scanf("%s",arr1);
                len=strlen(arr1);
                printf("Entered string is %s and its length is %d\n",arr1,len);
        }
        MPI_Bcast(&len,1,MPI_INT,0,MPI_COMM_WORLD);
        int d=len/size;
        MPI_Scatter(arr1,d,MPI_CHAR,arr2,d,MPI_CHAR,0,MPI_COMM_WORLD);
        int nonVowelCount=0;
        for(int j=0;j<len;j++)
        {
                if(arr2[j]!='a' && arr2[j]!='e' && arr2[j]!='i' && arr2[j]!='o' && arr2[j]!='u' &&
arr2[j]>97 && arr2[j]<123)
                {
                        nonVowelCount++;
                }
        }
        MPI_Gather(&nonVowelCount,1,MPI_INT,b,1,MPI_INT,0,MPI_COMM_WORLD);
        if(rank==0)
        {
                for(int i=0;i<N;i++)
                {
                        fprintf(stdout, "Number of non-vowels in rank %d is %d\n",i,b[i]);
                        fflush(stdout);
                        sum=b[i]+sum;
                }
        fprintf(stdout, "Total number of non-vowels = %d\n",sum);
        fflush(stdout);
        }
        MPI_Finalize();
        return 0;
}
```

SCREENSHOT



4.CODE

```
#include "mpi.h"
#include <stdio.h>
#include <string.h>
int main(int argc,char *argv[])
{
        int rank,size,N,l;
```

```
	char arr1[100],arr2[100],a[100],b[100],res[100];
	MPI_Init(&argc,&argv);
	MPI_Comm_rank(MPI_COMM_WORLD,&rank);
	MPI_Comm_size(MPI_COMM_WORLD,&size);
	MPI_Status status;
	if(rank==0)
	{
		N=size;
		fprintf(stdout,"In process %d-\n",rank);
		fprintf(stdout,"Enter two same length strings\n(string length should be evenly
divisble by %d)\n",N);
		fflush(stdout);
		scanf("%s",arr1);
		scanf("%s",arr2);
		l=strlen(arr1);
	}
	MPI_Bcast(&l,1,MPI_INT,0,MPI_COMM_WORLD);
	int d=l/size;
	MPI_Scatter(arr1,d,MPI_CHAR,a,d,MPI_CHAR,0,MPI_COMM_WORLD);
	MPI_Scatter(arr2,d,MPI_CHAR,b,d,MPI_CHAR,0,MPI_COMM_WORLD);
	char temp[100];
	for(int j=0;j<d*2;j+=2)
	{
		temp[j]=a[j/2];
		temp[j+1]=b[j/2];
	}
	temp[d*2]='\0';
	MPI_Gather(temp,d*2,MPI_CHAR,res,d*2,MPI_CHAR,0,MPI_COMM_WORLD);
	if(rank==0)
	{
		fprintf(stdout,"Result = %s\n",res);
		fflush(stdout);
	}
}
```

SCREENSHOT