PARALLEL AND DISTRIBUTED COMPUTING LAB

REPORT

NAME: S Shyam Sundaram

REG NO: 19BCE1560

PROGRAMMING ENVIRONMENT: MPI

PROBLEM: MPI

DATE: 13th October, 2021

HARDWARE CONFIGURATION:

CPU NAME	:	Intel core i5 – 1035G1 @ 1.00 Ghz
Number of Sockets:	:	1
Cores per Socket	:	4
Threads per core	:	1
L1 Cache size	:	320KB
L2 Cache size	:	2MB
L3 Cache size (Shared):		6MB
RAM	:	8 GB

QUESTION

Write an MPI program that calculates the sum of numbers read from a text file. Note: Text file is named num.txt.

CODE

```
#include <mpi.h>
#include<stdio.h>
#define MAXSIZE 100 //the text file has just 100 numbers
int main(int argc, char**argv)
{
   printf("Name: Shyam Sundaram\nReg num: 19BCE1560\n\n");
   MPI_Init(&argc,&argv);
   float startwtime,endwtime,totalTime;
   int myid;
   long int s,s0,startIndex,endIndex;
   long int i=0;
   float sum,part_sum;
   float data[MAXSIZE];
   MPI_Comm_rank(MPI_COMM_WORLD, &myid);

   int numprocs;
   MPI_Comm_size(MPI_COMM_WORLD, &numprocs);
```

```
int namelen;
char processor_name[MPI_MAX_PROCESSOR_NAME];
MPI_Get_processor_name(processor_name, &namelen);

FILE *fp;

fprintf(stderr,"Process %d is on %s\n",myid,processor_name);
fflush(stderr);

FILE *fptr;
int num;
fptr=fopen("num.txt","r");

while((num=getw(fptr))!=EOF)
{
   data[i++]=num/1.0;
}
fclose(fptr);

if(myid==0) //master
{
   s=(int)(MAXSIZE/numprocs);
   s0=s+(MAXSIZE%numprocs);
   printf("s=%ld,s0=%ld \n",s,s0);
}
else //worker
{
   s=(int)(MAXSIZE/numprocs);
   s0=s+(MAXSIZE%numprocs);
   startIndex = s0+(myid-1)*s;
   endIndex=startIndex+s;
   }
//Broadcasting
MPI_Bcast(&s,1,MPI_FLOAT,0,MPI_COMM_WORLD);
MPI_Bcast(&s0,1,MPI_FLOAT,0,MPI_COMM_WORLD);

//Time
totalTime=0;
if(myid==0)
{
   startwtime=MPI_Wtime();
}
//Partial Sum
```

```
    if(myid==0)
    {
        //master partial sum
        for(i=0;i<s0;i++)
        {
            part_sum = part_sum+data[i];
        }
        printf("Partial Sum from master = %f from processor %d\n", part_sum,myid);
    }
    else
    {
        //worker partial sum
        for(i=startIndex;i<endIndex;i++)
        {
            part_sum=part_sum+data[i];
        }
        printf("Partial Sum from worker = %f from processor %d\n", part_sum,myid);
    }

    //Global Sum
    MPI_Reduce(&part_sum,&sum,1, MPI_FLOAT,MPI_SUM,0,MPI_COMM_WORLD);

    if(myid==0)
    {
        endwtime=MPI_Wtime();
        totalTime=endwtime-startwtime;
        printf("Global Sum is %f\n", sum);
        printf("Execution time %f sec\n",totalTime);
    }

    MPI_Finalize();
}
```

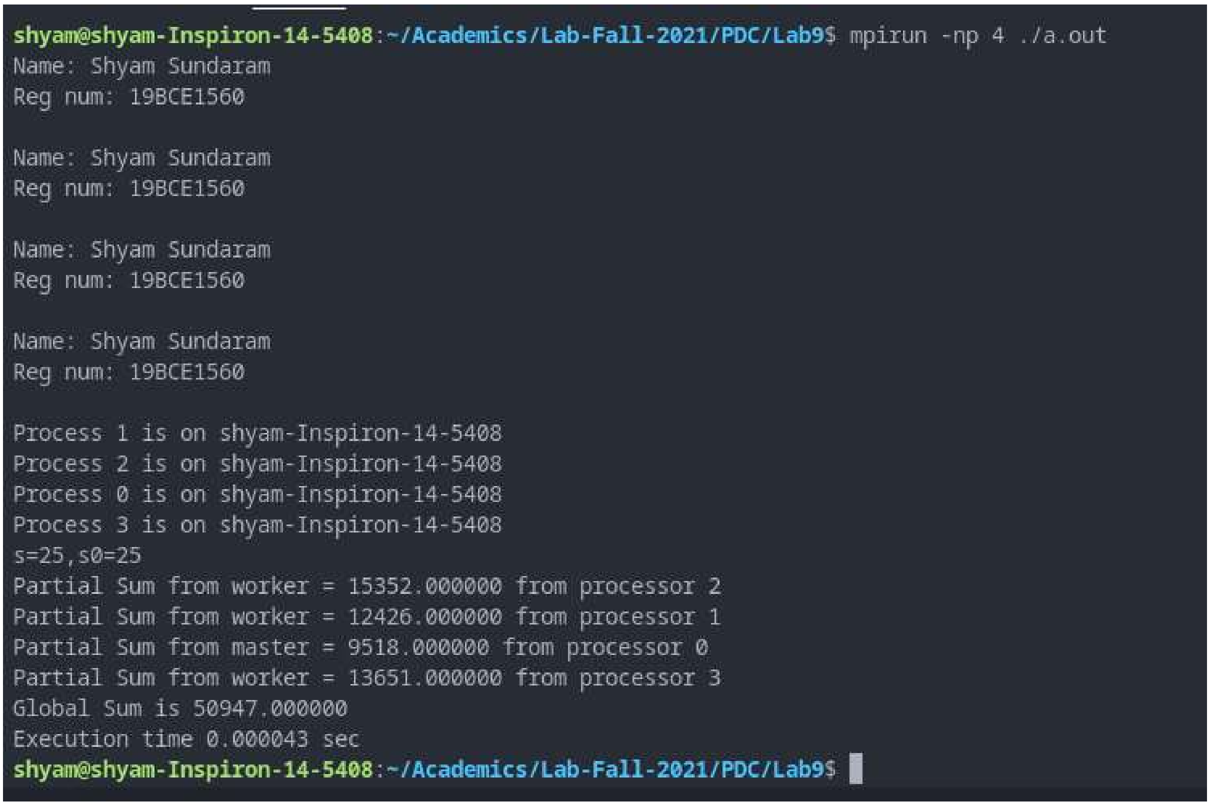
CODE (for write.c that created the num.txt file)

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

#define N 100

int main()
{
    FILE *fptr;
    int num;
    fptr=fopen("num.txt","w");
    srand(time(0));
    int upper=1000,lower=10;
    for(int i=0;i<N;++i)
    {
        num=i+1;//(rand()%(upper-lower+1))+lower;
        putw(num,fptr);
    }
    fclose(fptr);

    // fptr=fopen("num.txt","r");

    // while((num=getw(fptr))!=EOF)
    // {
    //     printf("%d\n",num);
    // }
    // fclose(fptr);
    return 0;

}
```

COMMANDS

```
mpicc mpisum.c
mpirun -np 4 ./a.out
```

OUTPUT



```
shyam@shyam-Inspiron-14-5408:~/Academics/Lab-Fall-2021/PDC/Lab9$ mpirun -np 4 ./a.out
Name: Shyam Sundaram
Reg num: 19BCE1560

Name: Shyam Sundaram
Reg num: 19BCE1560

Name: Shyam Sundaram
Reg num: 19BCE1560

Name: Shyam Sundaram
Reg num: 19BCE1560

Process 1 is on shyam-Inspiron-14-5408
Process 2 is on shyam-Inspiron-14-5408
Process 0 is on shyam-Inspiron-14-5408
Process 3 is on shyam-Inspiron-14-5408
s=25,s0=25
Partial Sum from worker = 15352.000000 from processor 2
Partial Sum from worker = 12426.000000 from processor 1
Partial Sum from master = 9518.000000 from processor 0
Partial Sum from worker = 13651.000000 from processor 3
Global Sum is 50947.000000
Execution time 0.000043 sec
shyam@shyam-Inspiron-14-5408:~/Academics/Lab-Fall-2021/PDC/Lab9$
```

OBSERVATION

Each worker calculated their respective sums and they are added at the end to produce the final sum of all numbers. The sum is verified to be correct.

CONCLUSION

We have found the sum of numbers read from a text file using MPI in C.