

NAME	JACOB JOHN
REGISTER NO.	16BCE2205
E-MAIL	jacob.john2016@vitstudent.ac.in

LAB ASSESSMENT #8

SCENARIO – 1

Write a 'C' Program to initialize an array of 100 elements in order to perform the sum of the elements sharing the load among 4 processes using MPI Send and MPI Recv.

Code:

```
/*Initialize an array of 100 elements
Perform the sum of the elements sharing the load among 4 processes using
MPI Send and MPI Recv*/

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

#define ARRAYSIZE 100
#define ROOT 0

int main( int argc, char** argv )
{
    int rank, fraction, PROCESSES;
    MPI_Status status;
    int i, dest, offset, source;
    float sum, mysum, tasks_sum;
    float data[100];
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &PROCESSES);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);

    fraction = (ARRAYSIZE / PROCESSES);
    if (rank == ROOT)
    {
        sum = 0;
        for(i = 0; i < ARRAYSIZE; i++){
            data[i] = i * 1.0;
        }
        offset = fraction;
        for (dest = 1; dest < PROCESSES; dest++){
            MPI_Send(&data[offset], fraction, MPI_FLOAT, dest, 0,
MPI_COMM_WORLD);
            printf("Sent %d elements to task %d offset= %d\n", fraction, dest,
offset);
        }
    }
}
```

```

        offset = offset + fraction;
    }
    offset = 0;
    for (i = 0; i < fraction; i++){
        mysum += data[i];
    }
    for (i = 1; i < PROCESSES; i++)
    {
        source = i;
        MPI_Recv(&tasks_sum, 1, MPI_FLOAT, source, 0, MPI_COMM_WORLD,
&status);
        sum += tasks_sum;
        printf(" Received partial sum %f from task %d\n", tasks_sum, i);
    }
    printf(" Final sum= %f \n", sum + mysum);
}

if (rank > ROOT)
{
    source = ROOT;
    MPI_Recv(&data[offset], fraction, MPI_FLOAT, source, 0, MPI_COMM_WORLD,
&status);
    for ( i=offset ; i <(offset+fraction); i++){
        tasks_sum += data[i];
    }
    dest = ROOT;
    MPI_Send(&tasks_sum, 1, MPI_FLOAT, dest, 0, MPI_COMM_WORLD);
}
MPI_Finalize();
}

```

The screenshot displays a macOS desktop environment. The top status bar shows the time as 8:11 PM on Thursday, September 14, 2017. The menu bar includes standard macOS applications (Code, File, Edit, Selection, View, Go, Debug, Terminal, Window, Help) and system status icons (network, battery, volume, etc.).

The central application is Visual Studio Code, which is open to a file named `array_send_rec.c`. The code is a C program that uses MPI for inter-process communication. It includes headers for `stdio.h`, `mpi.h`, and `stdlib.h`. It defines `ARRAYSIZE` as 100 and `ROOT` as 0. The `main` function initializes MPI, sets up variables for rank, fraction, and processes, and then sends data from the root process to other processes and receives data back. The terminal window at the bottom shows the execution of the program, displaying the distribution of data and the final sum.

The terminal output is as follows:

```
Jacobs-MacBook-Pro:Assignment_8 jacobjohn$ $HOME/opt/usr/local/bin/mpicc -o array ./array_send_rec.c
Jacobs-MacBook-Pro:Assignment_8 jacobjohn$ $HOME/opt/usr/local/bin/mpiexec -np 4 ./array
Sent 25 elements to task 1 offset= 25
Sent 25 elements to task 2 offset= 50
Sent 25 elements to task 3 offset= 75
Received partial sum 925.000000 from task 1
Received partial sum 1550.000000 from task 2
Received partial sum 2175.000000 from task 3
Final sum= 4950.000000
Jacobs-MacBook-Pro:Assignment_8 jacobjohn$
```

```
Jacobs-MacBook-Pro:Assignment_8 jacobjohn$ $HOME/opt/usr/local/bin/mpicc -o array ./array_send_rec.c
Jacobs-MacBook-Pro:Assignment_8 jacobjohn$ $HOME/opt/usr/local/bin/mpirun -np 4 ./array
Sent 25 elements to task 1 offset= 25
Sent 25 elements to task 2 offset= 50
Sent 25 elements to task 3 offset= 75
Received partial sum 925.000000 from task 1
Received partial sum 1550.000000 from task 2
Received partial sum 2175.000000 from task 3
Final sum= 4950.000000
```