import mpi.MPI;

import java.util.Arrays;

public class DistributedSumMPI {

public static void main(String[] args) {

MPI.Init(args);

int rank = MPI.COMM\_WORLD.Rank();

int size = MPI.COMM\_WORLD.Size();

int N = 100; // Total elements

int[] array = new int[N];

int elementsPerProcess = N / size;

int[] subArray = new int[elementsPerProcess];

if (rank == 0) {

// Initialize the array with values

for (int i = 0; i < N; i++) {

array[i] = i + 1;

}

}

// Scatter array among processes

MPI.COMM\_WORLD.Scatter(array, 0, elementsPerProcess, MPI.INT,

subArray, 0, elementsPerProcess, MPI.INT, 0);

// Compute partial sum

int localSum = Arrays.stream(subArray).sum();

System.out.println("Processor " + rank + " calculated partial sum: " + localSum);

// Gather all partial sums at root

int[] totalSum = new int[1];

int[] localSumArr = new int[1];

localSumArr[0] = localSum;

MPI.COMM\_WORLD.Reduce(localSumArr, 0, totalSum, 0, 1, MPI.INT, MPI.SUM, 0);

if (rank == 0) {

System.out.println("Final sum of array: " + totalSum[0]);

}

MPI.Finalize();

}

}

Output:

javac -cp .:/home/nishant/Downloads/mpj-v0\_44/lib/mpj.jar DistributedSumMPI.java

export MPJ\_HOME=/home/nishant/Downloads/mpj-v0\_44

/home/nishant/Downloads/mpj-v0\_44/bin/mpjrun.sh -np 4 -cp .:/home/nishant/Downloads/mpj-v0\_44/lib/mpj.jar DistributedSumMPI

MPJ Express (0.44) is started in the multicore configuration

Processor 1 calculated partial sum: 950

Processor 2 calculated partial sum: 1575

Processor 3 calculated partial sum: 2200

Processor 0 calculated partial sum: 325

Final sum of array: 5050