**Code**

**MPIAverge.java**

**import** mpi.MPI;

**import** java.util.\*;

**public** **class** MPIAverage {

**public** **static** **void** main(String args[]) {

*//Initialize and finalize*

MPI.Init(args);

**int** root = 0;

*//Rank and Size*

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

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

**int** arrsize = 8;

**int** chunksize = arrsize/size;

**if**(rank == root) {

System.out.println("\nSize of MPI Communicator : "+size);

}

System.out.println("\nThis process has rank "+rank);

**int** sendbuf[] = **new** **int**[arrsize];

**double** recdoublebuff[] = **new** **double**[size];

Random ran = **new** Random();

*// Generating random array in root process*

**if**(rank == root) {

**for**(**int** i=0;i<arrsize;i++) {

sendbuf[i] = ran.nextInt(100);

}

**for**(**int** i=0;i<arrsize;i++)

{

System.out.print(sendbuf[i]+" ");

}

}

**int** recvbuf[] = **new** **int**[chunksize];

*//scatter data*

MPI.COMM\_WORLD.Scatter(sendbuf, 0, chunksize, MPI.INT,

recvbuf, 0, chunksize, MPI.INT,

root);

*//Calc average of subset in every process*

System.out.println("\nProcess "+rank+" has data: \n");

**double** sum = 0.0;

**for** (**int** num: recvbuf) {

System.out.print(num+ " ");

sum += num;

}

**double** average[] = **new** **double**[1];

average[0] = (sum / recvbuf.length);

*//gater data to root process*

MPI.COMM\_WORLD.Gather(average, 0, 1, MPI.DOUBLE,

recdoublebuff, 0, 1, MPI.DOUBLE,

root);

*//display the doubled data*

**if**(rank == root) {

System.out.println("\nThe root process "+rank+" has data: ");

**for**(**int** i=0;i<size;i++)

System.out.print(recdoublebuff[i]+" ");

**double** tempsum = 0.0;

**for** (**double** num: recdoublebuff) {

tempsum += num;

}

System.out.println("\nThe average of the data is "+(tempsum / recdoublebuff.length));

}

MPI.Finalize();

}

}

*//javac -source 1.8 -target 1.8 -cp $MPJ\_HOME/lib/mpj.jar MPIAverage.java*

*//$MPJ\_HOME/bin/mpjrun.sh -np 4 MPIAverage*

**Output**

