```
#include <iostream>
#include <ostream>
#include <sstream>  //for stringstream/ostringstream/istringstream

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

using std::cout;
using std::endl;
using std::ostringstream;

int main(int argc, char **argv) {
    int     comm_sz;    /* number of processes  */
    int     my_rank;    /* process rank         */

    //Initialize the command line arguments on every process
    MPI_Init(&argc, &argv);

    //Get the number of processes in MPI_COMM_WORLD, and put
    //it in the 'comm_sz" variable; ie., how many processes
    //are running this program (the same as what you put in the
    //-np argument).
    MPI_Comm_size(MPI_COMM_WORLD, &comm_sz);

    //Get the rank of this particular process in MPI_COMM_WORLD,
    //and put it in the 'my_rank' variable -- ie., what number
    //is this process
    MPI_Comm_rank(MPI_COMM_WORLD, &my_rank);

    int array_size = 100;
    double* array = new double[array_size];

    if (my_rank == 0) {
        srand48(time(NULL));
        for (int i = 0; i < array_size; i++) {
            array[i] = drand48();
        }
    } else {
        for (int i = 0; i < array_size; i++) {
            array[i] = 0;
        }
    }

    ostringstream oss;  /*Writing what we want to say to a string stream
                          and then writing it's contents to cout will
                          prevent interleaving of print statements to
                          the root process */
    oss << "[process: " << my_rank << "]:";
    for (int i = 0; i < array_size; i++) {
        oss << " " << array[i];
    }
    cout << oss.str() << endl;

    MPI_Barrier(MPI_COMM_WORLD);

    MPI_Bcast(array /*the data we're broadcasting*/,
              array_size /*the data size */,
              MPI_DOUBLE /*the data type */,
              0 /*the process we're broadcasting from */,
              MPI_COMM_WORLD);

    oss.str("");
    oss.clear();
```

```
    oss << "[process: " << my_rank << "]:";
    for (int i = 0; i < array_size; i++) {
        oss << " " << array[i];
    }
    cout << oss.str() << endl;


    MPI_Finalize();
}
```