MPI – Part 1

1. Run the program `1c-hello-name.c` on multiple nodes. Make sure you see different "processor names". (You can find the code in the file `MPI-1-code.tgz`.)

2. Write an MPI program where each of p processes generates 20 random integers in the range $]-100, 100[$. Then, process 0 must print the single largest integer generated and the rank(s) of the process(es) containing it. Use of `MPI_MAXLOC` is not allowed. Run with $p >= 8$.

3. Write your own implementation of `MPI_Bcast` using `MPI_Send` and `MPI_Recv`. Your routine, `myMPI_Bcast`, will take the same arguments as the original `MPI_Bcast`; that is

```
int myMPI_Bcast( void *buffer, int count, MPI_Datatype datatype,
                 int root, MPI_Comm comm );
```

3.b Write also an MPI program where the root process initializes a vector of three elements with random values and uses your routine to broadcast the contents of the vector to the other processes.

4. Write your own implementation of `MPI_Reduce` using `MPI_Send` and `MPI_Recv`. Your routine, `myMPI_Reduce`, will have the following signature:

```
int myMPI_Reduce( const void *sendbuf, void *recvbuf, int count,
                  int root, MPI_Comm comm);
```

   This is a simplified version of the original `MPI_Reduce` which assumes double precision data (`double`) and performs the equivalent to the reduction operation `MPI_MIN`.

4.b Write also an MPI program where each process generates one random value and uses your routine to obtain the minimum among them.