Q2

1. We use MPI_Scatter and MPI_Reduce in this histogramming program. We use MPI_Scatter to scatter 2 million random numbers into equal-sized bins. For each MPI process, we calculate the total sum of all the numbers into the bin. Then we use MPI_Reduce to reduce all the sums in the bins(processes). Note that we specify the number of nodes and tasks per node of MPI by using the sbatch script so that we do not need to modify the code. We only need to read the world_size to determine how we scatter the data.

   2 nodes:

```
[zhang.yam@login-00 Question2]$ tail slurm-32509290.out
Process 94 has 20000 values with total sum 9989050479
Process 95 has 20000 values with total sum 10023012188
Process 96 has 20000 values with total sum 9950562739
Process 97 has 20000 values with total sum 9974712604
Process 98 has 20000 values with total sum 9946496853
Process 0 has 20000 values with total sum 10048107005
Process 99 has 20000 values with total sum 9982953575
Total sum = 1000168573420; Finished in 450.995554 ms
```

   4 nodes:

```
[zhang.yam@login-00 Question2]$ tail slurm-32509293.out
Process 94 has 20000 values with total sum 9997391974
Process 95 has 20000 values with total sum 9954943601
Process 96 has 20000 values with total sum 9998206867
Process 97 has 20000 values with total sum 9967121489
Process 98 has 20000 values with total sum 10028200492
Process 0 has 20000 values with total sum 9998338756
Process 99 has 20000 values with total sum 10023822890
Total sum = 999698172289; Finished in 705.474434 ms
```

   The running time of using 4 nodes is slower than using 2 nodes. The reason is that the communication among MPI processes is via network, so

the communications between nodes is slower than between processes on the same node. So the tasks on 4 nodes would be slower.

2. We just need to change the tasks per node option in our batch file. The running of 4 nodes is still slower. The main reason is still that the network between different nodes is slower.

   2 nodes:

```
[zhang.yam@login-00 Question2]$ tail slurm-32509294.out
Process 12 has 100000 values with total sum 49933464613
Process 13 has 100000 values with total sum 49958630083
Process 14 has 100000 values with total sum 49930650090
Process 15 has 100000 values with total sum 49848790469
Process 16 has 100000 values with total sum 49807387946
Process 17 has 100000 values with total sum 50037867092
Process 18 has 100000 values with total sum 50062749575
Process 0 has 100000 values with total sum 49814117690
Process 19 has 100000 values with total sum 50000294198
Total sum = 999818559416; Finished in 83.930658 ms
```

   4 nodes:

```
[zhang.yam@login-00 Question2]$ tail slurm-32509295.out
Process 12 has 100000 values with total sum 49908079077
Process 13 has 100000 values with total sum 49848806136
Process 14 has 100000 values with total sum 49960946730
Process 15 has 100000 values with total sum 49995989429
Process 16 has 100000 values with total sum 50077914644
Process 17 has 100000 values with total sum 49975249938
Process 18 has 100000 values with total sum 49949771812
Process 0 has 100000 values with total sum 49899487212
Process 19 has 100000 values with total sum 50121104829
Total sum = 999924951613; Finished in 134.265485 ms
```

3. From the test above, we could tell that the more nodes the more running time. Moreover, by comparing the a and b, we could also tell that, actually the more bins the more running. The main reason is also because of the scatter and the reduction process. The more bins, the more communication

needed between the processes on process scatter and reduction. It is similar to the Spark or Mapreduce shuffle process, which also needs a network to reduce data between different nodes. By proving that, we test the task on only one bin. Here is the result.

```
[zhang.yam@login-00 Question2]$ tail slurm-32509584.out
Process 0 has 2000000 values with total sum 1000485743027
Total sum = 1000485743027; Finished in 14.140449 ms
```