Question1.

Part1, file: a1q1p1.c

Compile Code:

mpicc -o a1q1p1.x a1q1p1.c -lm

mpirun -np 40 ./a1q1p1.x

Thoughts:

- Each process creates a local_data, an array has 3 elements, [0] stores eggholder value, [1] [2] store value of x and y respectively.

- Each process finds out the min value in its own local

- Using MPI_Gather, each process sends its min value to process 0, then process 0 to find out the min value among all processes.

Output:

```
[yliu327@mcsl assignment]$ nano alqlpl.c
[yliu327@mcsl assignment]$ mpicc -o alqlpl.x alqlpl.c -lm
[yliu327@mcsl assignment]$ mpirun -np 40 ./alqlpl.x
After Gathering data at root process:
[-954.097411, 478.516242, 434.196932]
[-953.678524, 484.396885, 435.219603]
[-955.950479, 511.083326, 404.038273]
[-952.372932, 510.094942, 402.970599]
[-944.685882, 474.413423, 430.449419]
[-953.031724, 478.088532, 433.272057]
[-954.159855, 480.358501, 431.172982]
[-950.982207, 480.948641, 430.981022]
[-953.369046, 511.556048, 405.863005]
[-953.554691, 481.903769, 432.048864]
[-954.607152, 480.130598, 435.718939]
[-957.992250, 511.604856, 404.377886]
[-954.180800, 486.175117, 436.694047]
[-956.192027, 480.460628, 431.042831]
[-957.323794, 511.879063, 402.823050]
[-959.103092, 511.863870, 404.364229]
[-955.679331, 511.572855, 405.321188]
[-953.691936, 510.581808, 403.808028]
[-952.897211, 476.752676, 431.992778]
[-955.593827, 484.126621, 434.813834]
[-954.422096, 478.371551, 434.013779]
[-952.730216, 483.382370, 433.481577]
[-954.033970, 478.723701, 429.396081]
[-955.508294, 510.986421, 402.637293]
[-953.820072, 476.296304, 431.793553]
[-948.052197, 509.191490, 400.874065]
[-956.156467, 511.257930, 402.683522]
[-954.917007, 511.015437, 404.387650]
[-954.665008, 479.188003, 434.804957]
[-952.397904, 482.933596, 438.401733]
[-955.368770, 483.961043, 434.265007]
[-955.795872, 484.071143, 434.738483]
[-952.793739, 510.584259, 404.191523]
[-958.194584, 511.599909, 403.633442]
[-954.617285, 483.267230, 433.482768]
[-953.269930, 480.682227, 436.385525]
[-959.004049, 511.947398, 404.818173]
[-950.556015, 483.739823, 439.328927]
[-952.148924, 482.128488, 437.348084]
[-956.535384, 511.178811, 403.060673]
Min Value is -959.103092, x is 511.863870, y is 404.364229:
[yliu327@mcsl assignment]$
```

Part2 files: a1q1p2, currently runTime is 10s for easy testing.

mpicc -o a1q1p2.x a1q1p2.c -lm

mpirun -np 40 ./a1q1p2.x

Thoughts:

- Base ideas are the same to part, each process finds out min and gather in process0

- gapTime = difftime(current_time, start_time); to set the running time

Output:

```
40 total processes failed to start
[yliu327@mcsl assignment]$ mpicc -o a1q1p2.x a1q1p2.c -lm
[yliu327@mcsl assignment]$ mpirun -np 40 ./a1q1p2.x
After Gathering data at root process:
[-959.630325, 511.999692, 404.321933]
[-959.616582, 511.993530, 404.182315]
[-959.596467, 511.987761, 404.171800]
[-959.557892, 511.978983, 404.313724]
[-959.562345, 511.979753, 404.306037]
[-959.583111, 511.998372, 404.444146]
[-959.604457, 511.989845, 404.262597]
[-959.634455, 511.999386, 404.291517]
[-959.624163, 511.995270, 404.206987]
[-959.537503, 511.986519, 404.444442]
[-959.599304, 511.995585, 404.075355]
[-959.559073, 511.986952, 404.038662]
[-959.638035, 511.999662, 404.195361]
[-959.573909, 511.997146, 404.453180]
[-959.581148, 511.982610, 404.239036]
[-959.626691, 511.997006, 404.287198]
[-959.609576, 511.993172, 404.141983]
[-959.551701, 511.974876, 404.151469]
[-959.594914, 511.986529, 404.209746]
[-959.598082, 511.998914, 404.415712]
[-959.506022, 511.961011, 404.242745]
[-959.560371, 511.985768, 404.050990]
[-959.595174, 511.991061, 404.108081]
[-959.634022, 511.998649, 404.273213]
[-959.563873, 511.997623, 403.983568]
[-959.607762, 511.994150, 404.119223]
[-959.633274, 511.998464, 404.186552]
[-959.546208, 511.993203, 403.974832]
[-959.624901, 511.996973, 404.159457]
[-959.566149, 511.978112, 404.195132]
[-959.570560, 511.990630, 404.407005]
[-959.558788, 511.975879, 404.200129]
[-959.556975, 511.983106, 404.369115]
[-959.611837, 511.995394, 404.335508]
[-959.560211, 511.979404, 404.116519]
[-959.585684, 511.989280, 404.350289]
[-959.485461, 511.985119, 403.914567]
[-959.599137, 511.997780, 404.056754]
[-959.490827, 511.959137, 404.294519]
[-959.548882, 511.988060, 404.433457]
Min Value is -959.638035, x is 511.999662, y is 404.195361:
[yliu327@mcsl assignment]$
```

Part 3, file: a1q1p3.c

mpicc -o a1q1p3.x a1q1p3.c -lm

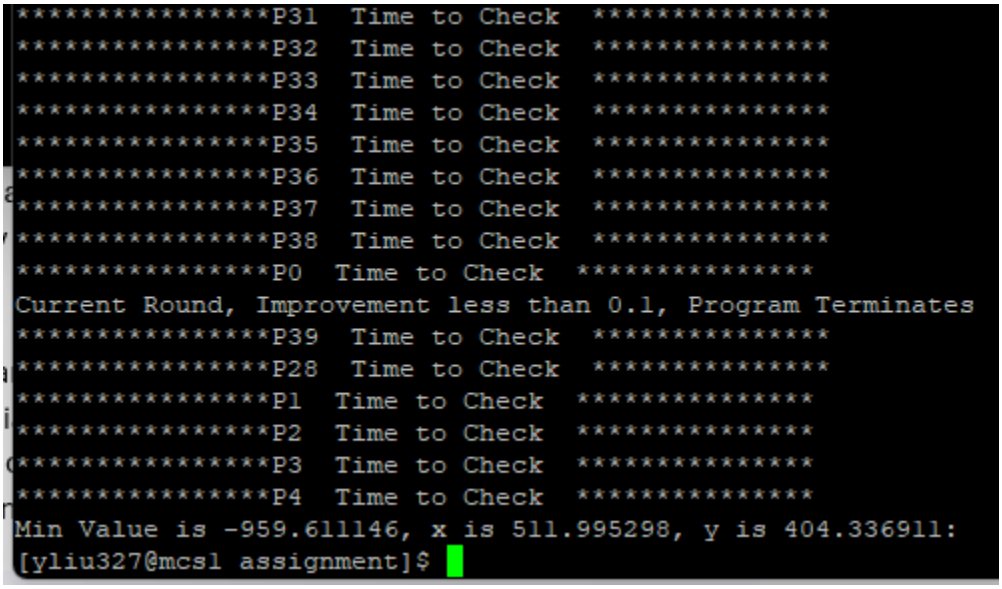
mpirun -np 40 ./a1q1p3.x

Thoughts:

- Base concepts are the same.

- Add a 5-sec check and flag which tells the loop break or continue

Output:



Question2. File: a1q2.c

mpicc -o a1q2.x a1q2.c

mpirun -np 40 ./a1q2.x

In my understanding from the hint and question description, file means in process 0 it can be generating a 16x16 matrix, and send each block, 4x4, to itself and p1, p2 and p3.

Thoughts:

- Cut 16x16 block into 4x4

- Send 4x4 for each process, using code below to get starting row and col.

```
start_row = (my_rank / block_size + time) * block_size;
start_col = (my_rank % block_size) * block_size;
```

- time = n/block_size to get total looping times, 4 times means there are 4 rows for 4 processes to get 16x16 matrix.

- Use MPI_Barrier(MPI_COMM_WORLD); to sync each process.

Output:

```
Row 2, Process 1 received:
132.0 133.0 134.0 135.0
148.0 149.0 150.0 151.0
164.0 165.0 166.0 167.0
180.0 181.0 182.0 183.0

Row 3, Process 1 received:
196.0 197.0 198.0 199.0
212.0 213.0 214.0 215.0
228.0 229.0 230.0 231.0
244.0 245.0 246.0 247.0

Row 1, Process 3 received:
76.0 77.0 78.0 79.0
92.0 93.0 94.0 95.0
108.0 109.0 110.0 111.0
124.0 125.0 126.0 127.0

Row 2, Process 3 received:
140.0 141.0 142.0 143.0
156.0 157.0 158.0 159.0
172.0 173.0 174.0 175.0
188.0 189.0 190.0 191.0

Row 3, Process 3 received:
204.0 205.0 206.0 207.0
220.0 221.0 222.0 223.0
236.0 237.0 238.0 239.0
252.0 253.0 254.0 255.0

[yliu327@mcsl assignment]$
```