Name: Makara Phav

Student ID: 20130844

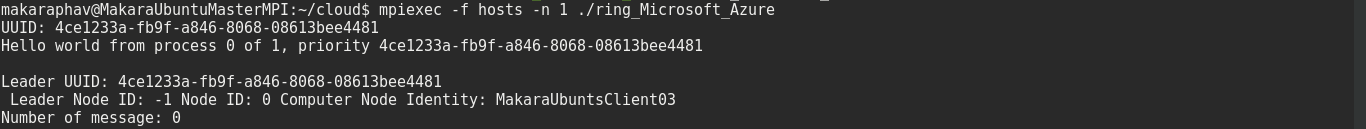
Report on First Programming Assignment:

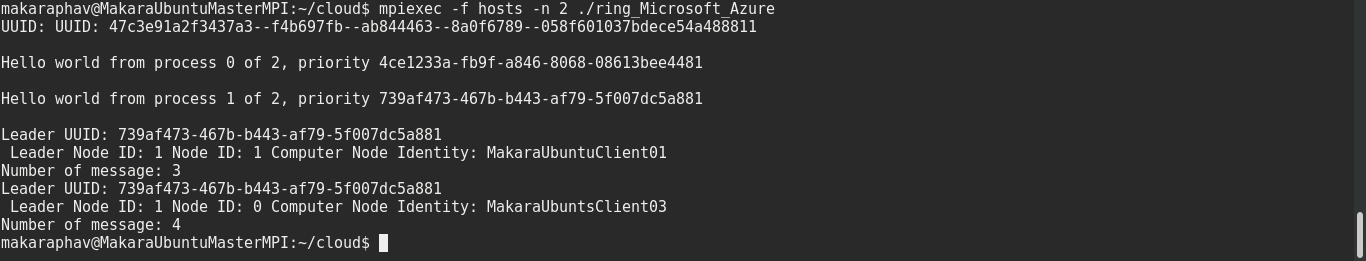
1-Source Code:

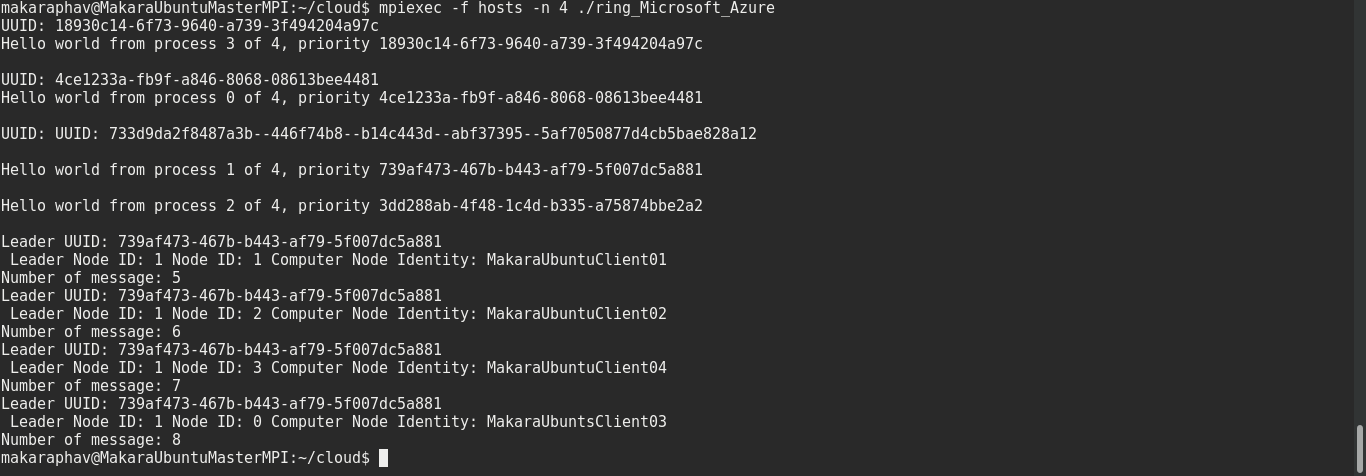
I include both source code and binary file inside the zip file.

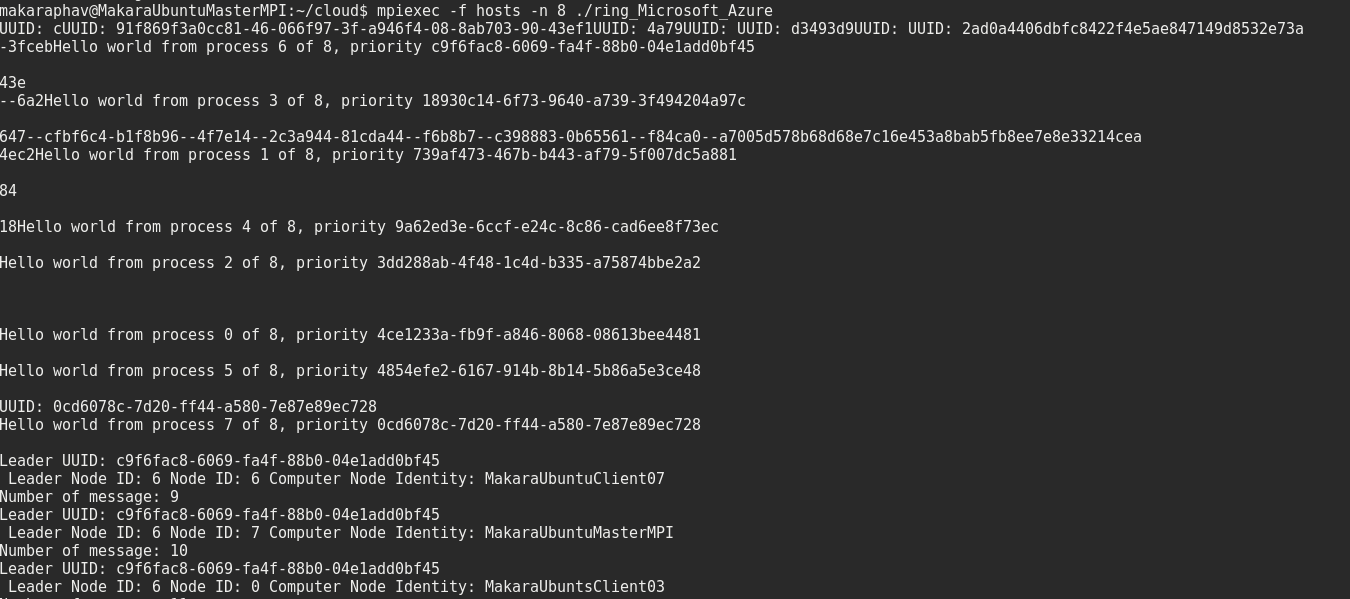
2- Screen Shoots:

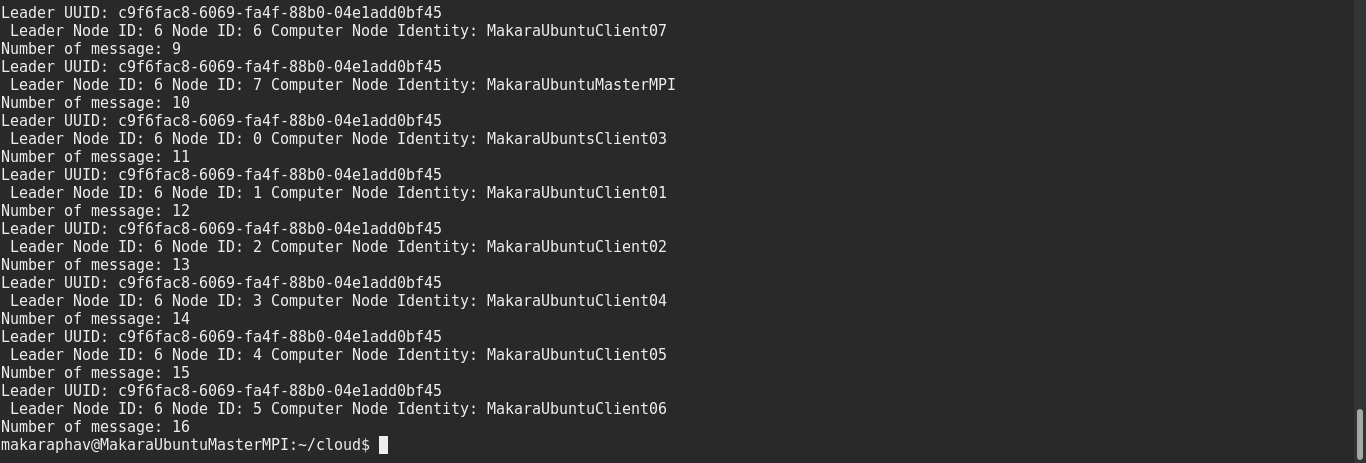
* Running on Microsoft Azure using UUID for electing the leader



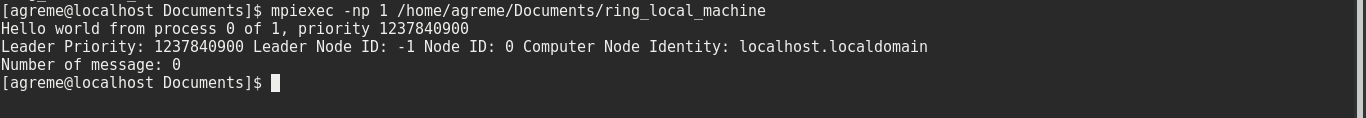


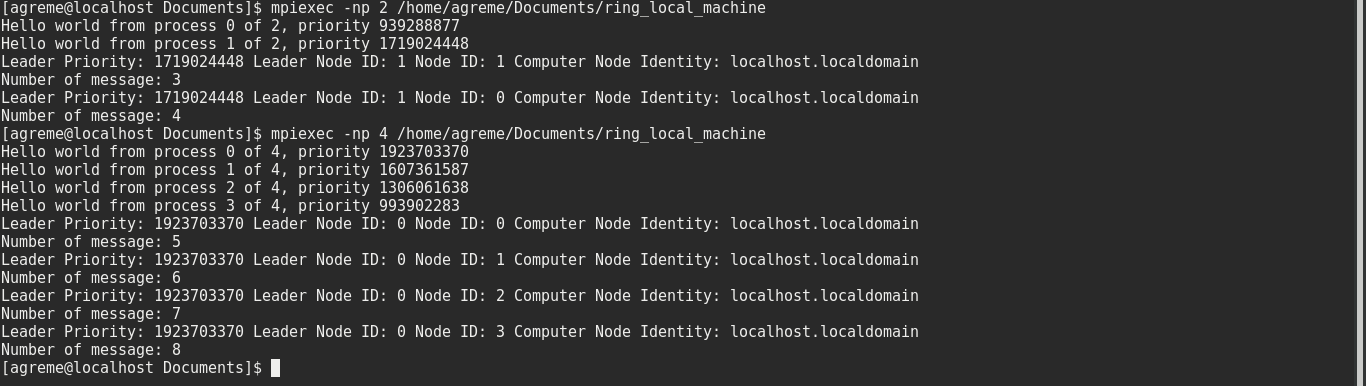


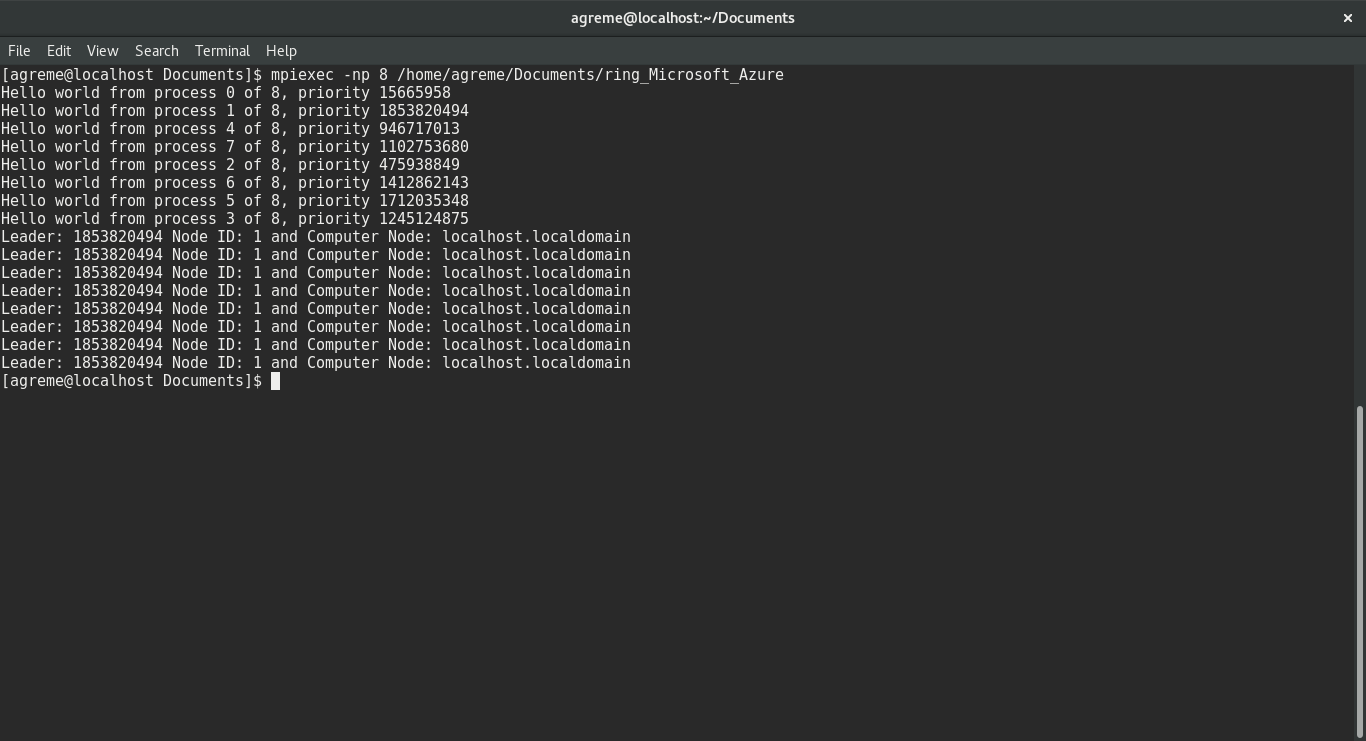




* Running on Local machine using random priority to choose leader







3- Number of Message Passing

|  |  |  |  |
| --- | --- | --- | --- |
| Number of Processor | Best case | Worst case | Average case |
| 1 | 0 | 0 | 0 |
| 2 | 7 | 7 | 7 |
| 4 | 26 | 26 | 26 |
| 8 | 100 | 100 | 100 |

4- Algorithm Implementation:

As it is described in the programming assignment document that we only send the message from node with index i to node with index (i+1) % size. So, I design a really basic leader election algorithm as follow. I will described my algorithm with pseudo code first:

while(!terminated){

if (ring\_size == 1){

my\_leader = my\_rank;

terminated = true;

break;

}

if (message\_terminated ){

terminated  = true

my\_leader  = message\_leader

My\_max = message\_max;

}

Else if (my\_max > message\_max){

my\_max  = message\_max;

my\_leader  = message\_leader;

terminated = false;

}

Else if (my\_max == message\_max){

if (message\_ leader == my\_leader)

terminated = true;

Else

my\_leader  = my\_rank;

terminated = false;

}

send (my\_message, destination);

receive (message);

}

[Noted]:

1 - Assuming that my\_max, my leader and terminated are variable belong to computer node

2 - Assuming that message\_max, message\_terminated, message\_leader belong to incoming message

3- Message is already pack with message\_max, message\_terminated and message\_leader

Firstly, all of the node will send out the massage until the maximum max\_value is found. So, at least, it needs around n message to pass to get it for each node. However, the first node who receives the maximum max id is not break the loop yet it will need to send the message to person to the right that it already receive the max value. I modify the algorithm to send more message to ensure the true leader was elected when a node has the same max priority, it once again need to send our message with terminated state to the other nodes. So, the rest of the node will end up send one more message than the previous one because the rest of the nodes is waiting for the terminated state with true leader priority. As a result, the reason that my algorithm always send out the same amount of messages despite the case of the arrangement of the priority in each machine node. Therefore, in big o notation we, can get the following performance from the above algorithm:

|  |  |  |  |
| --- | --- | --- | --- |
| Number of Processors | Best | Average | Worst |
| N | O(N^2) | O(N^2) | O(N^2) |