TSN3151: Laboratory 04

Before you proceed, you should have completed Laboratory 03 where you are able to send a "token" (message) starting from process 0 and ending at process 0. In order to ensure that you have done it correctly, at each process, increment the token (assuming that you sent an integer) and make sure they it passes through all the other processes.

The main purpose of this laboratory is to (1) allow for input from the user, (2) an exercise using a very common parallel computation which is the parallel pi computation, and (3) tracing of a MPI program code (which you can validate by running the code).

Part 1: Code Tracing

Suppose that MPI_COMM_WORLD consists of the three processes 0, 1, and 2, and suppose the following code is executed (note: you will need to add headers, etc. before you can compile it):

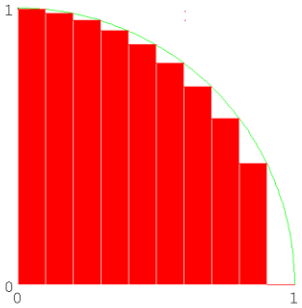
```
int x, y, z;
switch (my_rank){
      case 0: x=0; y=1; z=2;
            MPI_Bcast(&x, 1, MPI_INT, 0, MPI_COMM_WORLD);
            MPI_Send(&y, 1, MPI_INT, 2, 43, MPI_COMM_WORLD);
            MPI_Bcast(&z, 1, MPI_INT, 1, MPI_COMM_WORLD);
            break;
      case 1: x=3; y=4; z=5;
            MPI_Bcast(&x, 1, MPI_INT, 0, MPI_COMM_WORLD);
            MPI_Bcast(&y, 1, MPI_INT, 1, MPI_COMM_WORLD);
            break;
      case 2: x=6; y=7; z=8;
            MPI_Bcast(&z, 1, MPI_INT, 0, MPI_COMM_WORLD);
            MPI_Recv(&x, 1, MPI_INT, 0, 43, MPI_COMM_WORLD, &status);
            MPI_Bcast(&y, 1, MPI_INT, 1, MPI_COMM_WORLD);
            break;
}
printf("%d: x=%d y=%d z=%d\n", rank, x, y, z);
```

What are the values of x, y and z on each process after the code has been executed? Do this without compiling and running the code. Once you are confident on the answer, compile and run it to validate your answer.

If you are unable to trace it, add in a few `printf()` for each process (rank) and observer the output.

Part 2: Parallel Pi Computation

This part is about a common parallel computation to determine the value of pi. The algorithm I would like you to explore is to "compute the value of pi using integration" (in other words, calculate the area under the graph as illustrated below).



The model is based on the fact that the area of a circle is defined by $A = \pi r^2$. If r is 1, then the area of a circle of radius 1 (diameter 2) is π. From the graph above, we are basically computing $\pi/4$ as it is a quarter of a circle. Hence the area under the graph above has to be multiplied by 4 to get the value of π. Area under the graph is defined by $1/(1+x^2)$, therefore π is $4/(1+x^2)$.

Write a serial program to calculate Pi using this method.

Modify the above into an MPI code where the master process (rank 0) asks the user for the number of intervals. The master should then broadcast (recall your MPI routine for broadcast) this number to all of the other processes. Each process then adds up every nth interval, and finally, the results from each process is reduced (recall your MPI reduce routine) to obtain the results.

You can find the code for this online but the main objective here is to be able to code the MPI program yourself. Run your code with varying number of intervals, from 10 to 10,000 and observe the results. You can also find a lecture I gave before on this topic, with an animation explaining how this works here: https://youtu.be/BlJyek3wgcM?t=195

Test your program and record your results here.

10 intervals, π =
100 intervals, π =
1,000 intervals, π =
10,000 intervals, π =