Parallel programming lab 2

*Work on the problem assigned to your group until 11:30. If you finish early, move on to the next problem. Your group will be asked to spend 5 minutes at the end of class discussing the problem, your approach/solution, any challenges you ran into, and anything that you learned today.*

# Group 1.

The [weasel program](https://en.wikipedia.org/wiki/Weasel_program) is a classic example of using genetic algorithms, showing how random mutations and selective breeding (using a fitness function) can lead to the emergence of complexity, in this case a line from Shakespeare’s Hamlet. (Conceptually, the same approach applies to optimizing parameter choices of a computational model.) An important property of genetic algorithms is that the fittest individual of a generation can be less fit than the parents; why is this important?

Download a serial implementation from:

<https://github.com/ramcdougal/cbb750/blob/master/weaselprogram.py>

Create a parallel version using MPI. Your code should reproduce the serial output identically, regardless of how many processes are used. (Test with 1, 2, 3, and 8 processes.)

*Now move on to the task for group 2.*

# Group 2.

For last week’s homework, you made a parallel version of the deterministic reaction-diffusion simulation

<https://github.com/ramcdougal/cbb750/blob/master/propagating-signal.py>

using Python’s multiprocessing module with shared memory. Parallelize this again using MPI. Note that at most time steps, the only data that needs to be exchanged are the values at the borders between data to be processed by each rank.

Test your code with 1, 2, 3, and 8 processes. Be sure your code generates the correct graph in each case.

*Now move on to the task for group 1.*