AP(IT) race conditions

Simon Rogers, February 2017

Main lab for week 5. See also week5a if you finish this one within the hour.

Background

Last week you built a threaded system to find the minimum and maximum of a two-dimensional array. You probably did option 1 (doing an extra min or max in main) to collate the results from the multiple threads. Now you should do the same with a shared object with the aim of producing a *race condition*.

Tasks

- Implement a system for finding min and max using a shared object. The object should contain two double attributes, one for the global minimum and one for the global maximum. Give each min max thread a reference to this object. Have each thread compare with the values in each object *as it searches* (i.e. don't just compare once it has found the min and max). i.e. as you loop over the array in one of the threads, check each value against the value stored in the global option. This means that once all the threads have finished the shared object should have the global values in it.
- At the bottom of your main method, write a simple pair of for loops that loop over the original array and find the mimimum and maximum. Compare these values with those in the global object. What would you expect if there was a race condition? Think about why it's hard to make a race condition here.
- Change your code to try and make a race condition more likely.