CS 365, Spring 2017 — Feb 20th — Exam 1 Name: _____

Question 1. [10 points] Assume there is a grid of h rows and w columns of integer values. Using pseudo-code, briefly sketch a *sequential* algorithm (i.e., *not* parallel) that will find the maximum row and column sums in the grid. For example, consider the following example grid:

```
4 8 6 2 5 5
4 0 7 0 4 8
2 8 7 4 3 6
2 2 9 0 5 9
5 2 5 6 0 3
3 2 1 6 6 0
```

In this example, the maximum row sum is 30 (the first and third rows both have this sum), and the maximum column sum is 35 (the third column has this sum.)

Question 2. [30 points] Using pseudo-code, sketch a *parallel* algorithm for the problem described in Question 1. Your parallel algorithm should devide the overall h by w grid into an N (rows) by M (columns) grid of processors, assigning a smalller local grid region to each processor. Make sure your pseudo-code shows

- How each processor determines which portion of the overall grid it will work on
- How the local results computed by each processor are combined to form an overall solution

Question 3. [60 points] Implement the parallel algorithm you sketched in Question 2 using MPI. To get started, see the instructions on the exam web page:

`https://ycpcs.github.io/cs365-spring2017/assign/exam01.html`

Edit the code in `maxrowcol.c`. To run the program use the command

`./runpar` *filename* N M

where *filename* is an input file, N is the number of rows of processes, and M is the number of columns of processes. The input file contains a grid of integers.

The output of the program should include two lines indicating the maximum row and column sums in the overall grid of integers, in the format

`Maximum row sum is` X
`Maximum column sum is` Y

where X and Y are the maximum row and column sums.

Some hints and suggestions:

- Code to read the input data into a `Grid` object is provided
- A helper function `divide_work` is provided to help divide up the work
- A helper function `find_max` is provided to find the maximum value in an array of `int` values
- Arrays `row_sums` and `col_sums` are provided for storing the sum of the rows/columns of the local grid region
- Arrays `global_row_sums` and `global_col_sums` are provided for storing the global row/column sums (of the overall global grid); most likely, only the root process will use these
- `MPI_Reduce` can be used on an array of values if you pass a value of the `count` parameter that is greater than 1
- Only one process (e.g., the root process) should print the final results

Example test commands and their expected outputs:

`./runpar test1.dat 2 2`	`./runpar test2.dat 2 2`	`./runpar test3.dat 2 2`
Expected output:	Expected output:	Expected output:
`Maximum row sum is 30` `Maximum column sum is 35`	`Maximum row sum is 162` `Maximum column sum is 119`	`Maximum row sum is 551` `Maximum column sum is 1004`