Assignment 1

The Red/Blue computation simulates two interactive flows: an *n* by *n* board is initialized so cells have one of three colors: red, white, and blue, where white is empty, red moves right, and blue moves down. (The board may be initialized with 1/3 cells in read, 1/3 in white and 1/3 in blue and colors should be interleaved and spread across the board. You need to write a separate function *board_init* to initialize the board.) Colors wraparound to the opposite side when reaching the edge. In the first half step of an iteration, any red color can move right one cell if the cell to the right is unoccupied (white); on the second half step, any blue color can move down one cell if the cell below it is unoccupied (white); the case where red vacates a cell (first half) and blue moves into it (second half) is okay. Viewing the board as overlaid with *t* by *t* tiles (*t* divides *n*), the computation terminates if any tile's colored squares are more than *c%* one color (blue or red).

- § Use MPI to write a solution to the Red/Blue computation.
- § Assume the processes are organized as a one-dimensional linear array.
- § It is fine to use a square grid, i.e. *n* by *n* grid for *n* being grid size.
- § For simplicity assume *n* is divisible by *t*.
- § Each process will hold a number of $k \times n/t$ rows for $k \geq 0$.
- § For the purpose of load balancing, processes should have roughly the same number of rows and the difference not greater than n/*t* rows.
- § Your program must produce correct results for *nprocs* being greater than or equal to one.
- § Your program needs to ask for 4 user defined parameters (integers) as inputs: grid size *n*, tile size *t*, terminating threshold *c*, and maximum number of iterations *max_iters*.
- § Your program needs to print out which tile (or tiles if more than one) has the colored squares more than *c%* one color (blue or red).

Your submissions will be marked on accuracy of results, suitability of the parallelism applied, and quality of your reports.
The **report** should be concise and clear (2-3 A4 pages) and should contain the following sections:

a. Problem definition and Requirements
b. Parallel algorithm design
c. Implementation and Testing
d. Manual (e.g. how to run the program, input and output)

You MUST attempt this assignment individually.

Submission Requirements

Your submission must be made by **11:59pm on Friday, 15 April, 2016**.

1. Create **a tar file** which contains your **reports**, ***makefiles*** and **any source files (e.g., .c and .h files)**. **DO NOT INCLUD ANY OBJECT OR BINARY FILES.**
2. Submit *only* the **.tar** file.

NB: NO LATE SUMISSIONS WILL BE PERMITTED

Failure to follow these submission requirements can result in loss of marks.

Marking Scheme

Correctness	10
Report	2
Code Quality (algorithm efficiency, comments, readability, portability, etc.)	3
Total	**15**