CMPS 224/396AA: GPU COMPUTING
ASSIGNMENT 5

In this assignment, you will implement a histogram operation using atomic operations, and optimize it using privatization, shared memory, and thread coarsening.

Instructions

1. Place the files provided with this assignment in a single directory. The files are:
   - `main.cu`: contains setup and sequential code
   - `kernel.cu`: where you will implement your code (you should only modify this file)
   - `common.h`: for shared declarations across `main.cu` and `kernel.cu`
   - `timer.h`: to assist with timing
   - `Makefile`: used for compilation

2. Edit `kernel.cu` where TODO is indicated as follows:
   - Histogram with privatization and shared memory only:
     - o Implement the kernel (`histogram_private_kernel`):
       - Declare a private copy of the histogram in shared memory and initialize it to 0
       - Have each thread load a single image pixel and atomically update the corresponding histogram bin count in shared memory
       - Commit the non-zero bin counts to the global copy of the histogram in parallel
     - o Implement the host code (`histogram_gpu_private`):
       - Launch the grid (Note: the image has already been copied to global memory for you and the global bins have already been initialized to 0)
   - Histogram with privatization, shared memory, and thread coarsening:
     - o Implement the kernel (`histogram_private_coarse_kernel`):
       - Similar to the previous implementation, but each thread loads multiple image pixels based on a coarsening factor (make sure the loads are coalesced)
     - o Implement the host code (`histogram_gpu_private_coarse`):
       - Similar to the previous implementation, but remember to take the coarsening factor into consideration when selecting the number of thread blocks in the grid

3. Compile your code by running: `make`

4. Test your code by running: `./histogram`
   - If you are using the HPC cluster, do not forget to use the submission system. Do not run on the head node!
   - For testing on different input sizes, you can provide your own values for the input dimensions as follows: `./histogram <height> <width>`

5. You are also provided with a file called `questions.txt` which contains questions about the assignment. Answer the questions in the file after implementing your kernel.

Submission

Submit your modified `kernel.cu` and `questions.txt` files via Moodle by the due date. Do not submit any other files or compressed folders.