Programming assignment: Part 2
Diagonal matrix multiplication (DMM) in CUDA

Overview: In the part 2 of the programming assignment, you need to write CUDA version of the DMM program that you had optimized and multi-threaded for CPU in part 1.

Deliverables: Similar to part 1, you are required to submit a report along with your code. The report should include details about the optimizations you considered while implementing. Optionally, you can also include what further optimizations are possible to reduce the runtime of your implementation.

What/how to submit: Clone the GitLab repository on your machine and follow its README.md for instructions on how to compile and run the programs. All your code has to be implemented in "header/". DO NOT MODIFY main.cu.

Submit your assignment as a zip file named as per the last five digits of your SR number (e.g., 15964.zip). It should contain two directories – i.e., "headers/" that would contain your implementation and "reports/". Name your reports as "reports/Report_Part2.pdf".

Where to run/test your CUDA program: You all would have gotten access to the GPU server of the department. You may GPUs in that server to run your program. However, expect the GPUs to be very busy during the end of the semester. Therefore, in the below you point to a GPU simulator called GPGPU-sim. This is a software that runs on the CPU but *emulates* a GPU. It runs CUDA application out-of-the-box. You can install this simulator in your local machine and do your homework.

Prerequisites:
In order to compile CUDA code (whether for a GPU or GPGPU-Sim), you will need to install the CUDA compiler (nvcc). You can download and install this from the official NVIDIA website (https://developer.nvidia.com/cuda-11.0-download-archive). You do NOT need to install the driver if you plan to use GPGPU-sim. GPGPU-Sim currently works with CUDA versions 4.5 - 11.0.

Instructions for using GPGPU-sim:
Clone GPGPU-sim using the link below-
https://github.com/gpgpu-sim/gpgpu-sim_distribution
Run your code for the **SM7_TITANV** configuration. You can find the configuration file at : gpgpu-sim_distribution/configs/tested-cfgs/
You can use GPGPU-sim profiler to performance optimize your CUDA application.
If you are running on the GPU hardware, look for NSight and/or NVprof tools.