GPU Lab 2

Part 3: *GPU Optimization: The purpose of this exercise is to experiment with the optimization techniques discussed in the lecture today.*

Starting with the naïve implementation of matrix-matrix multiplication you implemented in **Part 2** of Lab 1, refactor your code to compute the product in sub-blocks. i.e. given a two-dimensional grid of thread-blocks, then the two-dimensional thread-block parameterized by (I, J) computes

$$(\mathbf{AB})_{ij} = \sum_{k=0}^{k=15} \mathbf{A}_{ik}\mathbf{B}_{kj}$$

$$\text{for } DI \leq i < D(I+1),\ DJ \leq j < D(J+1)\ ,$$

where the thread-block size is $D \times D$ with $D = 16$.

Determine the impact of using the following techniques to improve efficiency:

a. Shared memory for the sub-blocks of **A** and **B**.

b. Padding the shared memory arrays to avoid bank conflicts.

c. Bind textures array handles to the **A** and **B** arrays.

d. Unroll the inner-loop [HINT: the loop iterates exactly 16 times].

e. Experiment with how many 16×16 sub-matrix multiplies each thread-block performs.

f. Try different thread-block sizes, i.e. blocks of side $D = 2, 4, 8, 16$.

Tabulate the floating point performance (GFLOP/s) and bandwidth (GB/s) for each permutation that you try.
