Quiz Questions for Module 3

1. If we need to use each thread to calculate one output element of a vector addition, what would be the expression for mapping the thread/block indices to data index:
   (A) i=threadIdx.x + threadIdx.y;
   (B) i=blockIdx.x + threadIdx.x;
   (C) i=blockIdx.x*blockDim.x + threadIdx.x;
   (D) i=blockIdx.x * threadIdx.x;

   Answer: (C)

   Explanation: This is the case we covered in Lecture 2.3.

2. We want to use each thread to calculate two (adjacent) output elements of a vector addition. Assume that variable i should be the index for the first element to be processed by a thread. What would be the expression for mapping the thread/block indices to data index of the first element?
   (A) i=blockIdx.x*blockDim.x + threadIdx.x +2;
   (B) i=blockIdx.x*threadIdx.x*2
   (C) i=(blockIdx.x*blockDim.x + threadIdx.x)*2
   (D) i=blockIdx.x*blockDim.x*2 + threadIdx.x

   Answer: (C)

   Explanation: Every thread covers two adjacent output elements. The starting data index is simply twice the global thread index. Another way to look at it is that all previous blocks cover (blockIdx.x*blockDim.x)*2. Within the block, each thread covers 2 elements so the beginning position for a thread is threadIdx.x.

3. We want to use each thread to calculate two output elements of a vector addition. Each thread block processes 2*blockDim.x consecutive elements that form two sections. All threads in each block will first process a section, each processing one element. They will then all move to the next section, again each processing one element. Assume that variable i should be the index for the first element to be processed by a thread. What would be the expression for mapping the thread/block indices to data index of the first element?
   (A) i=blockIdx.x*blockDim.x + threadIdx.x +2;
   (B) i=blockIdx.x*threadIdx.x*2
   (C) i=(blockIdx.x*blockDim.x + threadIdx.x)*2
   (D) i=blockIdx.x*blockDim.x*2 + threadIdx.x

   Answer: (D)

   Explanation: Each previous block covers (blockIdx.x*blockDim.x)*2. The beginning elements of the threads are consecutive in this case so just add threadIdx.x to it.

4. For a vector addition, assume that the vector length is 8000, each thread calculates one output element, and the thread block size is 1024 threads. The programmer configures the kernel launch to have a minimal number of thread blocks to cover all output elements. How many threads will be in the grid?
   (A) 8000
   (B) 8196
   (C) 8192
   (D) 8200

   Answer: (C)

   Explanation: ceil(8000/1024)*1024 = 8 * 1024 = 8192. Another way to look at it is the minimal multiple of 1024 to cover 8000 is 1024*8 = 8192.