CSCD 439
Lab2

Michael Peterson

4) Did you notice any duplicated memory access in the kernel?
Yes, every element in the input array is getting read by two different threads.

5) Please explain how each thread is mapped to an unique spot(index) of the output array? Can you draw some conclusions when mapping thread to data with regards to whether you should think the input array first or the result array first?
It looks like a good strategy is to have insure that writes to a given location should only happen from one thread.

6) Is there any data read/write conflict in the kernel for each thread? That is, in you kernel, are there any two threads reading a same element at the same, or is it possible to happen? Every element in the input is read by two different threads.

7) In you kernel, are there any two threads writing a same spot at the same, or is that case possible to happen? In this solution, concurrent writes to the same location is not possible.

8) how many thread blocks are generated by the kernel?

```
blocksize * (int)ceil ((double)  n / (double) blocksize)
```