Started on	Thursday, September 22, 2022, 2:35 PM
State	Finished
Completed on	Thursday, September 22, 2022, 3:02 PM
Time taken	26 mins 35 secs

Question **1**
Complete
Points out of 1.00

Assume that we want to use each thread to calculate two (adjacent) output elements of a vector addition. Assume that variable i should be initialized with the index for the first element to be processed by a thread. Which of the following should be used for such initialization to allow correct, coalesced memory accesses to these first elements in the following statement?

if(i<n) C[i] = A[i] + B[i];

Select one:

- a. None of the answers
- [✓ b. int i=(blockIdx.x*blockDim.x + threadIdx.x)*2;]
- c. int i=(threadIdx.x*blockDim.x + blockIdx.x)*2;
- [✓ d. int i=(blockIdx.x*blockDim.x)*2 + threadIdx.x;]
- e. int i=(threadIdx.x*blockDim.x)*2 + blockIdx.x;

Question **2**
Complete
Points out of 1.00

Assume that we want to use each thread to calculate two (adjacent) output elements of a vector addition. Assume that variable i should be initialized with the index for the first element to be processed by a thread. what would be the correct statement for each thread to process the second element?

Select one:

- [✓ a. None of the answers]
- b. if (i+threadIdx.x < n) C[i+threadIdx.x] = A[i+threadIdx.x] + B[i+threadIdx.x];
- [✓ c. if (i+1<n) C[i+1] = A[i+1] + B[i+1];]
- d. if (i<n) C[i+1] = A[i+1] + B[i+1];
- e. if (i+blockIdx.x < n) C[i+blockIdx.x] = A[i+blockIdx.x] + B[i+blockIdx.x];

Question **3**

Complete

Points out of 1.00

Assume the following simple matrix multiplication kernel

```
 __global__ void MatrixMulKernel(float* M, float* N, float* P, int Width)
 {
   int Row = blockIdx.y*blockDim.y+threadIdx.y;
   int Col = blockIdx.x*blockDim.x+threadIdx.x;
   if ((Row < Width) && (Col < Width)) {
     float Pvalue = 0;
     for (int k = 0; k < Width; ++k) {
           Pvalue += M[Row*Width+k] * N[k*Width+Col];
     }
     P[Row*Width+Col] = Pvalue;
   }
  }
```

Based on the judging criterion in Lecture 6.2, which of the following is true?

Select one:

- a. M[Row*Width+k] and N[k*Width+Col] are coalesced but P[Row*Width+Col] is not
- [✓ b. M[Row*Width+k] is not coalesced but N[k*Width+Col] and P[Row*Width+Col] both are]
- c. None of the answers
- d. M[Row*Width+k], N[k*Width+Col] and P[Row*Width+Col] are all coalesced
- [✓ e. M[Row*Width+k] is coalesced but N[k*Width+Col] and P[Row*Width+Col] are not]

Question **4**

Complete

Points out of 1.00

Assume the following simple matrix multiplication kernel

```
 __global__ void MatrixMulKernel(float* M, float* N, float* P, int Width)
 {
   int Row = blockIdx.y*blockDim.y+threadIdx.y;
   int Col = blockIdx.x*blockDim.x+threadIdx.x;
   if ((Row < Width) && (Col < Width)) {
     float Pvalue = 0;
     for (int k = 0; k < Width; ++k) {
           Pvalue += M[Row*Width+k] * N[k*Width+Col];
     }
     P[Row*Width+Col] = Pvalue;
   }
  }
```

For the tiled single-precision matrix multiplication kernel of the above, assume that each thread block is 32x32 and the system has a DRAM bust size of 128 bytes. How many DRAM bursts will be delivered to the processor as a result of loading one M-matrix element by a thread block (one k step)? Keep in mind that each single precision floating point number is four bytes.

Select one:

- [✓ a. 32]
- b. None of the answers
- c. 64
- d. 128
- e. 16

Question **5**

Complete

Points out of 1.00

If a CUDA device's SM (streaming multiprocessor) can take up to 1,536 threads and up to 8 thread blocks. Which of the following block configuration would result in the most number of threads in each SM?

Select one:

- a. 1024 threads per block
- b. 64 threads per block
- c. 128 threads per block
- [✓ d. 512 threads per block]
- e. None of the answers

Question **6**

Complete

Points out of 1.00

Assume the following simple matrix multiplication kernel

```
__global__ void MatrixMulKernel(float* M, float* N, float* P, int Width)
{
  int Row = blockIdx.y*blockDim.y+threadIdx.y;
  int Col = blockIdx.x*blockDim.x+threadIdx.x;
  if ((Row < Width) && (Col < Width)) {
    float Pvalue = 0;
    for (int k = 0; k < Width; ++k) {
          Pvalue += M[Row*Width+k] * N[k*Width+Col];
    }
    P[Row*Width+Col] = Pvalue;
  }
}
```

If we launch the kernel with a block size of 16X16 on a 1000X1000 matrix, how many warps will be in the invalid range?

Select one:

- [✓ a. 508]
- b. 1008
- c. 1000
- [✓ d. None of the answers]
- e. 500

Question **7**

Complete

Points out of 1.00

If a CUDA device's SM (streaming multiprocessor) can have 48KBytes shared memory. It takes up to 2,560 threads and up to 32 thread blocks. Which of the following block configuration would result in the most number of threads in each SM for the tiled matrix multiplication computation with C ***double*** type of data?

Select one or more:

- a. 8*8
- [✓ b. 16*16]
- c. 4*4
- d. 32*32
- e. 64*64
- f. None of the answers