Name _____

CS/EE 147
GPU Architecture and Parallel Programming

Sample Midterm

Name _____

1. [10 points] For the following basic reduction kernel code fragment, if the block size is 2048 and warp size is 32, how many warps in a block will have divergence during the iteration where stride is equal to 1? Stride equal to 32? Stride equal to 128?

```
for (unsigned int stride = 1; stride <= blockDim.x; stride *= 2)
{
    __syncthreads();
    if (threadIdx.x % stride == 0) {
       partialSum[2*threadIdx.x]+= partialSum[2*threadIdx.x + stride];
    }
}
```

1: _____ 32: _____ 128: _____

2. [20 points] In a parallel Reduction implementation, each thread loads two input elements from global memory to shared memory. The input elements are stored in:
`__shared__ float partialSum[2*blockDim.x];`
For simplicity, you do not need to do boundary condition checking. Answer the following.
(hint: start with the thread indexing first.)

   Complete the code for a thread to naïvely load two input elements in an uncoalesced manner:

   unsigned int start = 2*blockIdx.x*blockDim.x;

   unsigned int t = _____

   partialSum[t] = input[start + _____]

   partialSum[t+1] = input[start + _____]

   Complete the code for a thread to load two input elements in a coalesced manner:

   unsigned int start = 2*blockIdx.x*blockDim.x;

   unsigned int t = _____

   partialSum[t] = input[start + _____]

   partialSum[blockDim.x+t] = input[start + _____]

3. [10 points] You need to write a kernel that operates on an image of size 400x900. You would like to allocate one thread to each pixel. You would like the thread blocks to be square and to use

Name _____

the maximum number of threads per block possible on the device (assume 1536 thread limit and 8 thread block limit).

a. [4 points] What would you select as the grid and block dimensions?

b. [6 points] Assuming next that we use blocks of size 16x16, how many warps would experience thread divergence?

4. [20 points] For the below Vector Add kernel answer the following questions. Assume a vector size of *n*, block size of 256 threads and (*n*-1)/256+1 thread blocks.

```
__global__ void VecAdd(int n,const float *A,const float *B,float* C) {
      for (unsigned int i = 0; i<n; i++){
            C[i] = A[i] + B[i];
      }
}
```

a. [4 points] Does the kernel produce the desired results?

b. [6 points] How many additions are performed in this VecAdd compared to the VecAdd you implemented for assignment 1? (Do NOT count `i++` in the loop statement as an addition.)

c. [6 points] How many total *loads* to memory are performed in this VecAdd compared to the VecAdd you implemented in assignment 1?

d. [4 points] Would this parallel VecAdd kernel run faster, or a serial implementation of Vector Add?

Name _____

5. [20 points] For the following, explain how it could harm performance and possible ways the program can be modified to reduce this effect. Please be specific.
    a. [10 points] The application needs to access global memory to get one value for every operation.
    How this harms performance:

    Technique/change that could reduce this effect:

    b. [10 points] Control divergence: (aka Warp divergence)
    How this harms performance:

    Technique/change that could reduce this effect:

Name _____

1. DMA (Direct Memory Access) hardware transfers data between

    a. virtual addresses
    b. memory mapped addresses
    c. mythical addresses
    d. physical addresses
    e. imaginary addresses

2. cudaMalloc allocates memory in:

    a. host memory
    b. pinned memory
    c. device memory
    d. shared memory
    e. persistent memory

3. cudaHostAlloc allocates memory in:

    a. shared memory
    b. pinned memory
    c. device memory
    d. global memory
    e. persistent memory

4. If your CUDA application has a single stream, can you concurrently copy data and execute a kernel?

    a. Yes
    b. No

5. Each CUDA stream is a _____ of operations

    a. Event
    b. Command
    c. Heap
    d. Queue
    e. Stack

6. Commands (aka Events) in CUDA streams can be processed out-of-order.

    a. True
    b. False

7. Which of the following is false?

    a. Events in CUDA streams are processed in FIFO order
    b. The OS can accidentally swap out a page that is being transferred by DMA
    c. Kernel launches are CUDA events
    d. Copies are CUDA events
    e. All CUDA API calls are CUDA events

For the following questions answer whether it is (a) true or (b) false.

8. Virtual memory addresses are translated to physical memory addresses using page tables. T

9. All warps in a thread block must execute the same instruction simultaneously. F

10. Warps can finish executing at different times. T

11. In the GPU, a scheduler exists to pick warps to issue for execution. T

12. PTX is an intermediate representation for GPU code. T

13. GPUs can boot an Operating System. F

14. Which of the following is true?

    a. Instructions in a warp are processed out-of-order
    b. Warps consist of multiple thread blocks
    c. All warps in an SM share the same program counter
    d. All threads in a warp share the same program counter
    e. GPUs do not use program counters to access instruction