CMPE 297 Lab#2

Due: Friday, Sep 2, 5:15pm
Total Score: /100

Instructor: Hyeran Jeon

Computer Engineering Department, San Jose State University

Group ID	// identical to your board id			
Member 1	Name		Student ID	
Member 2	Name		Student ID	

In this Lab, you will complete a CUDA-version Vector Add code by translating kernel code and inserting memory operations. Follow the steps below. Submit the completed code to Canvas before leaving the classroom.

Step 1: Download skeleton code

- Open web browser (*epiphany*), go to Canvas→Labs→Lab2 and download following file
  - lab2.tar.bz2
- Open *terminal* and type following commands
  - cd "*folder that you downloaded the files"*
  - tar –xvjf lab2.tar.bz2

Step 2: Complete the code

- Open cmpe297_vecAdd.cu with *gedit* or *vi*
- Locations that you need to modify are marked with ***"// FILL HERE:"***.
  - vectorAdd function should be translated to a CUDA kernel
  - GPU memory allocation for input/output vectors A, B and C should be inserted
    - Use the following names for the GPU-side memory pointers
      - ***d_A, d_B, and d_C. (already declared in the code)***
  - Input vectors A and B should be copied from host to device
  - Set values of two integer variables, **blocksPerGrid** and **threadsPerBlock** so that you can run vectorAdd kernel with ***4 blocks of 256 threads***

  - vectorAdd function call should be changed to kernel call invocation statement
  - Output vector C should be copied back from device to host
- Refer to the lecture slide
  - To read the lecture slide in the Jetson board, you should download pdf version lecture slide

Step 3: Compile vectorAdd

- Type following command to compile cmpe297_vecAdd.cu
  - make
- You can also compile the code by simply running the following command:
  - nvcc –o vectorAdd cmpe297_vecAdd.cu

Step 4: Run vectorAdd

- Type following command to run vectorAdd
  - ./vectorAdd
- If the following sentence appears, you are done!
  - "Test PASSED"

Step 5: Check how many warps are used for the vectorAdd

- Add the two lines
  - `int warpid = threadIdx.x >>5;`
  - `printf("block id = %d, warp id = %d\n", blockIdx.x, warpid);`
- Compile the code again by giving the following commands and rerun vectorAdd
  - make clean; make; ./vectorAdd
- By seeing the printfs, fill the following table and submit this hard copy Lab sheet when you leave the classroom

Question	Your answer
# warps per thread block	
Total warps used in the kernel	

Step 6: Submit the completed cmpe297_vecAdd.cu file to Canvas

  - One copy per group