CEPARCO 2nd Term 2024-2025
GPU Project

Group assignment (Histogram counting):

Group 2	Atomic operations
Group 6	Shared memory concept
Group 7	Cooperative group

CUDA programming project specifications: **Histogram counting**

Input: A vector containing 32-bit integers; 10 bins for the histogram.
Process: Each integer *N* in the vector is processed using modulo 10 (N %10). The resulting value determines the corresponding histogram bin, which is then incremented accordingly.
Output #1: The final count of values in each of the 10 histogram bins.
Output #2: Video recording of the assigned CUDA concept (upload on Youtube; Should be "unlisted" and NOT "YouTube Kids"). Link to be placed in Github.
Example: The number 25 maps to bin 5 since 25 % 10=5, so bin 5 is incremented.

Note:
1.) Write the kernel using the specified method in (1) C program; (2) CUDA C program using Colab platform. Place your group number and group members in the first cell.
2.) CUDA program should use Unified memory, pre-fetching and memadvise.
3.) Time the kernel portion only with vector size of 2^{28} 32-bit integers.
4.) For each kernel, execute at least 30 times and get the average execution time.
5.) For the data, initialize each vector with values of your choice. Please document this value.
6.) Check the correctness of your output. Thus, if the C version is your "sanity check answer key," then the output of the CUDA version must be checked with the C version and output correspondingly (i.e., CUDA kernel output is correct).
7.) Place your project in Github (ensure I can access your Github).
8.) The GitHub repository contains the following:

a.) Link to your YouTube video. The video should be between 12-15 minutes discussing the CUDA concept assigned to the group. Make sure that the discussion should be in layman's term. You may use your project to illustrate the CUDA concept. Make sure that you are seen in the video. It is not necessary for all group members to be in the video.

b.) Readme section (inline and not link) containing the following report:

i.) screenshot of the program output with execution time
ii.) screenshot of the program output with correctness check (C)
iii.) screenshot of the program output, including correctness check (CUDA)
iv.) comparative table of execution time as well as analysis of the performance the kernel (how many times faster, why is it faster, etc.)
v.) Discuss the problems encountered and solutions made, unique methodology used, AHA moments, etc.

Rubric:

C program with correct implementation and output	10
Cuda program using the assigned CUDA concept with correct implementation and output	30
Screenshot	10
Comparative result	10
Analysis of result	10
Video presentation	30

Group assignment (Hadamard product):

Group 4	2D/3D variable concept
Group 8	2D/3D variable with shared memory

CUDA programming project specifications: **Hadamard product using 2D variables**

Input: Three matrices (X,Y,Z) of floating-point type.
Process: Element-wise multiplication with matrices *X* and *Y* and place the result in *Z*
Output #1: Output of matrix Z.
Output #2: Video recording of the assigned CUDA concept (upload on Youtube; Should be "unlisted" and NOT "YouTube Kids"). Link to be placed in Github.

Note:
1.) Write the kernel using the specified method in (1) C program; (2) CUDA C program using Colab platform. Place your group number and group members in the first cell.
2.) CUDA program should use Unified memory, pre-fetching and memadvise.
3.) Time the kernel portion only. For each version, time the process/kernel for vector size *r* = {1024x1024, 2048*2048, 4096*4096} and threads per block {8x8, 16x16, 32x32}
4.) For each kernel, execute at least 30 times and get the average execution time.
5.) For the data, initialize each vector with values of your choice. Please document this value.
6.) Check the correctness of your output. Thus, if the C version is your "sanity check answer key," then the output of the CUDA version must be checked with the C version and output correspondingly (i.e., CUDA kernel output is correct).
7.) Place your project in Github (ensure I can access your Github).
8.) The GitHub repository contains the following:

a.) Link to your YouTube video. The video should be between 12-15 minutes discussing the CUDA concept assigned to the group. Make sure that the discussion should be in layman's term. You may use your project to illustrate the CUDA concept. Make sure that you are seen in the video. It is not necessary for all group members to be in the video.

b.) Readme section (inline and not link) containing the following report:

i.) screenshot of the program output with execution time
ii.) screenshot of the program output with correctness check (C)
iii.) screenshot of the program output, including correctness check (CUDA)
iv.) comparative table of execution time as well as analysis of the performance various size variants (how many times faster, why is it faster, etc.)
v.) Discuss the problems encountered and solutions made, unique methodology used, AHA moments, etc.

Rubric:

C program with correct implementation and output	10
Cuda program using the assigned CUDA concept with correct implementation and output	30
Screenshot	10
Comparative result	10
Analysis of result	10
Video presentation	30

Group assignment (1D convolution):

Group 1	Multiple data transfer method Implement using 1.) Unified memory 2.) Prefetching of data with memory advice 3.) Data initialization as a CUDA kernel 4.) Old method of transferring data between CPU and memory (memCUDA malloc + CUDAmemcpy)
Group 3	Shared memory concept
Group 5	Concepts of stream

CUDA programming project specifications: **1D convolution**

Input: Two vectors: input vector *in* and output vector *out*.
Process: 1D convolution is defined as out[i] = (in[i] + in[i+1] + in[i+2] / 3.0f
Output #1: First and last 20 elements of vector *out*.
Output #2: Video recording of the assigned CUDA concept (upload on Youtube; Should be "unlisted" and NOT "YouTube Kids"). Link to be placed in Github.

Note:
1.) Write the kernel using the specified method in (1) C program; (2) CUDA C program using Colab platform. Place your group number and group members in the first cell.
2.) Group 3 and 5: CUDA program should use Unified memory, pre-fetching and memadvise.
3.) Time the kernel portion only with vector size of 2^{28} floating point.
4.) For each kernel, execute at least 30 times and get the average execution time.
5.) For the data, initialize each vector with values of your choice. Please document this value.
6.) Check the correctness of your output. Thus, if the C version is your "sanity check answer key," then the output of the CUDA version must be checked with the C version and output correspondingly (i.e., CUDA kernel output is correct).
7.) Place your project in Github (ensure I can access your Github).
8.) The GitHub repository contains the following:

a.) Link to your YouTube video. The video should be between 12-15 minutes discussing the CUDA concept assigned to the group. Make sure that the discussion should be in layman's term. You may use your project to illustrate the CUDA concept. Make sure that you are seen in the video. It is not necessary for all group members to be in the video.

b.) Readme section (inline and not link) containing the following report:

i.) screenshot of the program output with execution time
ii.) screenshot of the program output with correctness check (C)
iii.) screenshot of the program output, including correctness check (CUDA)
iv.) comparative table of execution time as well as analysis of the performance of various kernel implementation metod (how many times faster, why is it faster, etc.).

Group 1: include comparative table as well as analysis for various data transfer time.

v.) Discuss the problems encountered and solutions made, unique methodology used, AHA moments, etc.

Rubric:

C program with correct implementation and output	10
Cuda program using the assigned CUDA concept with correct implementation and output	30
Screenshot	10
Comparative result	10
Analysis of result	10
Video presentation	30