Contribution: Yitong Li mainly finish the exercise 1 and Guoqing Liang mainly finish the exercise 2.

Exercise 1:

1.

Upload my code: lab3_ex1_template.cu to google cloud.

Use the following order to compile:

```
!nvcc lab3_ex1_template.cu
```

Use the following order to run:

```
! ./a.out 1024
```

The format is ./a.out datalength

```
[80] !nvcc lab3_ex1_template.cu
```

```
! ./a.out 1024

Copy memory to the GPU time elapsed 0.000040 sec
GPU Kernel time elapsed 0.000031 sec
Copy the GPU memory back to the CPU time elapsed 0.000023 sec
Results correct!
```

2.

(1). N times in1[i] + in2[i]

(2). Read the in1[i] and in2[i], 2N

3.

(1). 1024 threads and (256 + 1024 -1)/256 = 4 blocks

TPB = 256, so the number of block is 4. For each add operation, use 1 thread to calculate.

(2). Follow the order:

```
!/usr/local/cuda-11/bin/nv-nsight-cu-cli
/content/drive/MyDrive/Assignment2/a.out 1024
```

Get the results.

According to the screenshot:

```
Section: Occupancy
------------------------------------------------------------------------ --------------- ------------------------------
Block Limit SM                                                                    block                             16
Block Limit Registers                                                             block                             16
Block Limit Shared Mem                                                            block                             16
Block Limit Warps                                                                 block                              4
Theoretical Active Warps per SM                                                    warp                             32
Theoretical Occupancy                                                                 %                            100
Achieved Occupancy                                                                    %                          23.44
Achieved Active Warps Per SM                                                       warp                           7.50
------------------------------------------------------------------------ --------------- ------------------------------
```

The Achieved Occupancy is 23.44%

4.

(1). It still work just as the following screenshot:

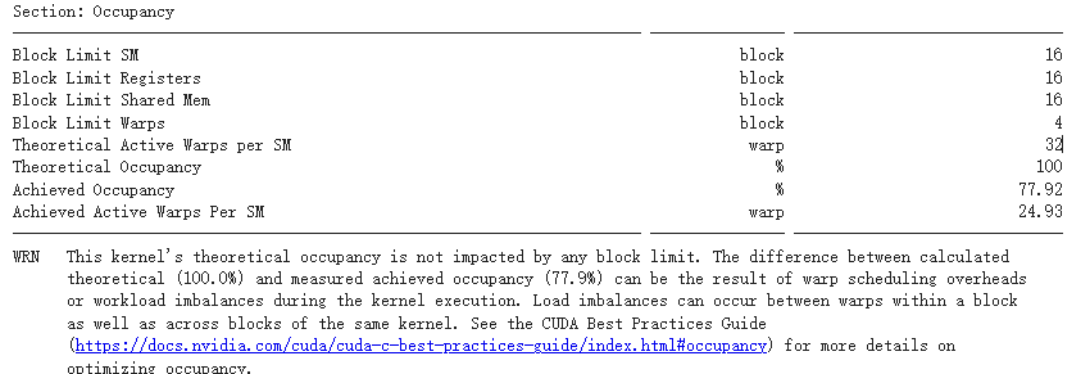
```
! ./a.out 131070

Copy memory to the GPU time elapsed 0.000845 sec
GPU Kernel time elapsed 0.000075 sec
Copy the GPU memory back to the CPU time elapsed 0.000872 sec
Results correct!
```

(2).same as the 3.(1)

131070 threads

(256 + 131070 -1)/256 = 512 blocks

(3).as the screenshot:



Section: Occupancy

Block Limit SM	block	16
Block Limit Registers	block	16
Block Limit Shared Mem	block	16
Block Limit Warps	block	4
Theoretical Active Warps per SM	warp	32
Theoretical Occupancy	%	100
Achieved Occupancy	%	77.92
Achieved Active Warps Per SM	warp	24.93

WRN This kernel's theoretical occupancy is not impacted by any block limit. The difference between calculated theoretical (100.0%) and measured achieved occupancy (77.9%) can be the result of warp scheduling overheads or workload imbalances during the kernel execution. Load imbalances can occur between warps within a block as well as across blocks of the same kernel. See the CUDA Best Practices Guide (https://docs.nvidia.com/cuda/cuda-c-best-practices-guide/index.html#occupancy) for more details on optimizing occupancy.

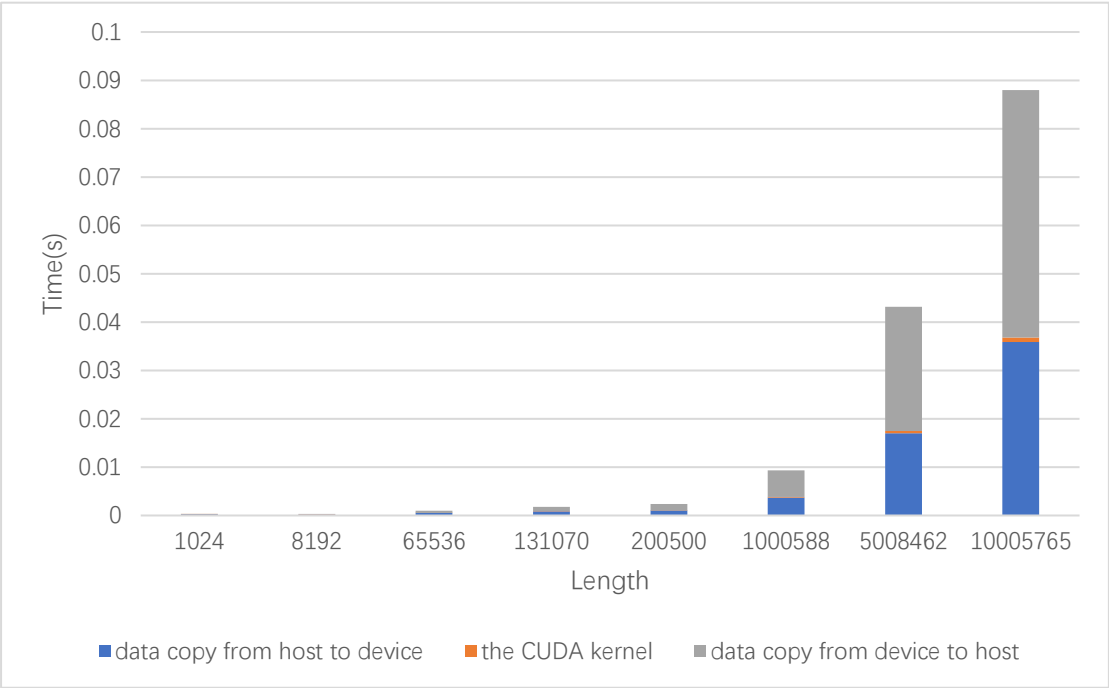
The Achieved Occupancy is 77.92%

5.

Test 8 different length in total. The results are following:

inputlength	data copy from host to device	the CUDA kernel	data copy from device to host
1024	0.000233	0.000035	0.000024
8192	0.000218	0.000039	0.00008
65536	0.000508	0.000038	0.000444
131070	0.000838	0.000082	0.000855
200500	0.001025	0.000043	0.001344
1000588	0.003731	0.000125	0.005476
5008462	0.017011	0.00049	0.025677
10005765	0.035892	0.000933	0.051195

Use it, I drawn the stacked bar chart:



Exercise 2：

1.

machine learning, graphic processing, data science.

2.

Addition: numARows * numBColumns * (numAColumns - 1)

Multiplication: numARows * numBColumns * numAColumns

3.

numARows *numAColumns*numBRows*numBColumns

4.

(1).Blocks = ((256 + 16 - 1)/ 16) * ((256 + 16 - 1)/ 16) = 256

Threads = Blocks * Grids = 128 * 128 = 16384

(2).P

```
  Section: Occupancy
  ------------------------------------------------------------------------ --------------- ------------------------------
  Block Limit SM                                                                    block                             16
  Block Limit Registers                                                             block                              4
  Block Limit Shared Mem                                                            block                             16
  Block Limit Warps                                                                 block                              4
  Theoretical Active Warps per SM                                                    warp                             32
  Theoretical Occupancy                                                                 %                            100
  Achieved Occupancy                                                                    %                          42.82
  Achieved Active Warps Per SM                                                       warp                          13.70
  ------------------------------------------------------------------------ --------------- ------------------------------
```

Achieved Occupancy = 42.82%

5.

(1) Yes

(2).Blocks = ((511 + 16 - 1)/ 16) * ((4094 + 16 - 1)/ 16) = 8192
Threads = Blocks * Grids = 8192 * 256 = 2097152

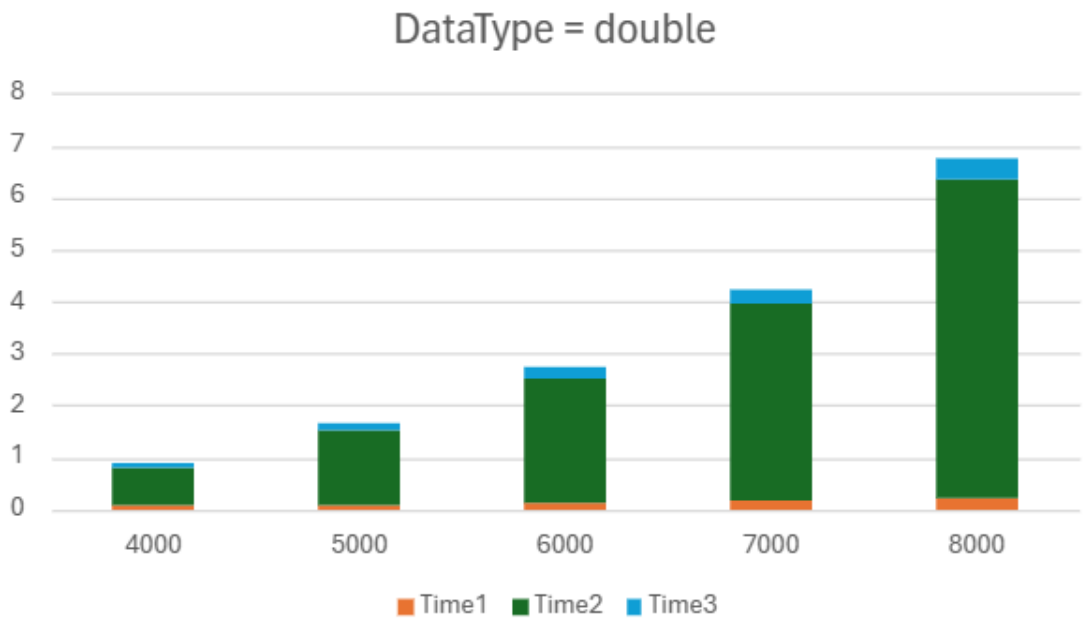
(3).

```
Section: Occupancy
------------------------------------------------------------------------------ --------------- ------------------------------
Block Limit SM                                                                 block                              16
Block Limit Registers                                                          block                               4
Block Limit Shared Mem                                                         block                              16
Block Limit Warps                                                              block                               4
Theoretical Active Warps per SM                                                warp                               32
Theoretical Occupancy                                                          %                                 100
Achieved Occupancy                                                             %                               99.17
Achieved Active Warps Per SM                                                   warp                            31.73
------------------------------------------------------------------------------ --------------- ------------------------------
```

Achieved Occupancy = 99.17%

6.

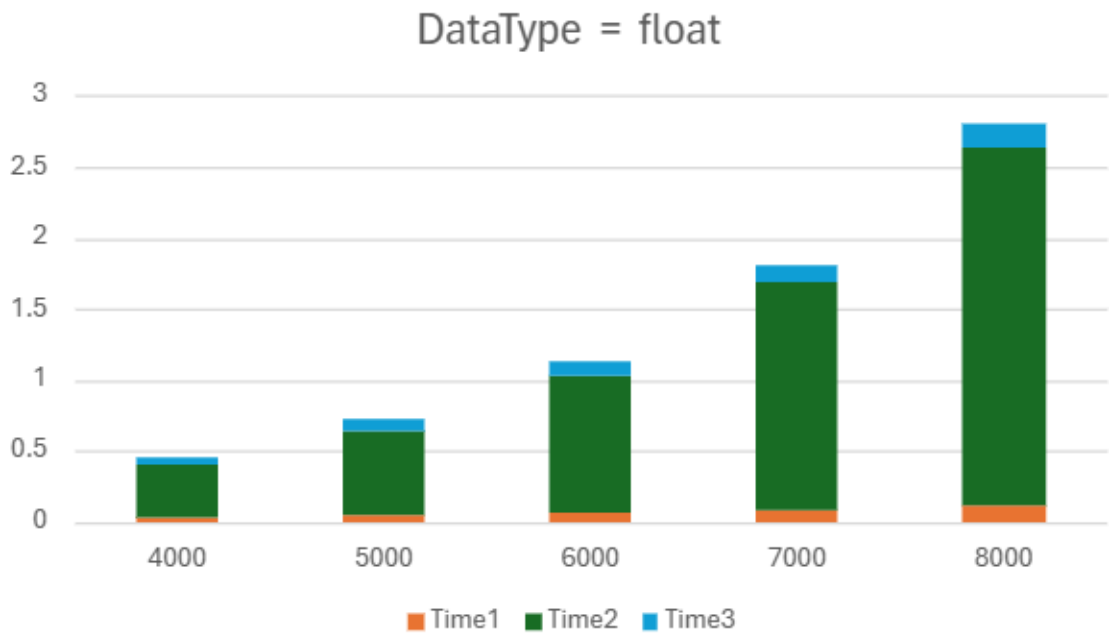
MatrixSize	Time1	Time2	Time3
4000	0.055313	0.739178	0.09681
5000	0.082686	1.448493	0.128007
6000	0.125639	2.407707	0.219407
7000	0.164025	3.780909	0.290298
8000	0.218979	6.153248	0.395821



MatrixSize: Rows and Cols for MatrixA and MatrixB
Time1: data copy from host to device
Time2: the CUDA kernel
Time3: data copy from device to host

7.

MatrixSiz	Time1	Time2	Time3
4000	0.026751	0.375341	0.041759
5000	0.042856	0.60271	0.069076
6000	0.062812	0.961197	0.111522
7000	0.084788	1.598097	0.127727
8000	0.110325	2.531368	0.163684



Time consumption for all three steps are much shorter after changing datatype from ‘double’ to ‘float’.