Final Year B. Tech., Sem VII 2022-23

High Performance Computing Lab

PRN: 2020BTECS00206

Full Name: SAYALI YOGESH DESAI

Batch: B4

Assignment No. 10

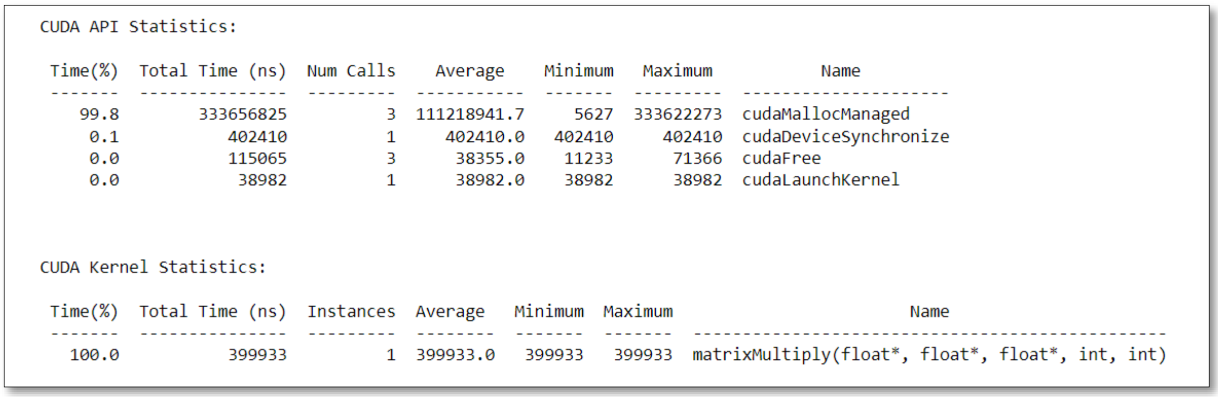
---

1. Implement Matrix-matrix Multiplication using global memory in CUDA C. Analyze and tune the program for getting maximum speed up. Do Profiling and state what part of the code takes the huge amount of time to execute.

```
#include <stdio.h>
void initWith(float num, float *a, int SIZE)
{
  for(int i = 0; i < SIZE; ++i)
  {
    a[i] = num;
  }
}

__global__
void matrixMultiply(float *result, float *a, float *b, int N, int SIZE)
{
  int start = blockIdx.x * blockDim.x + threadIdx.x;
  int stride = gridDim.x * blockDim.x;
   for(int i = start; i < SIZE; i += stride)
  {
    int row = i / N;
    float sum = 0
    for (int j = 0; j < N; j++)
    {
      sum += a[row * N + j] * b[N * j + row];
    }
    result[i] = sum;
void checkElementsAre(float target, float *array, int SIZE)
{
  for(int i = 0; i < SIZE; i++)
  {
    if(array[i] != target)
    {
      printf("FAIL: array[%d] - %0.0f does not equal %0.0f\n", i, array[i], target);
      exit(1);
    }
  }
  printf("SUCCESS! All values multiplied correctly.\n");
}
int main()
{
  const int N = 1024;
  const int SIZE = N * N; // sqaure matrix
  size_t size = SIZE * sizeof(float);
  float *a;
  float *b;
  float *c;

  cudaMallocManaged(&a, size);
  cudaMallocManaged(&b, size);
  cudaMallocManaged(&c, size);

  initWith(3, a, SIZE);
  initWith(4, b, SIZE);
  initWith(0, c, SIZE);

  matrixMultiply<<<1, 1>>>(c, a, b, N, SIZE);
  cudaDeviceSynchronize();

  checkElementsAre(12288, c, SIZE);

  cudaFree(a);
  cudaFree(b);
  cudaFree(c);
}
```

8*8 Matrix:

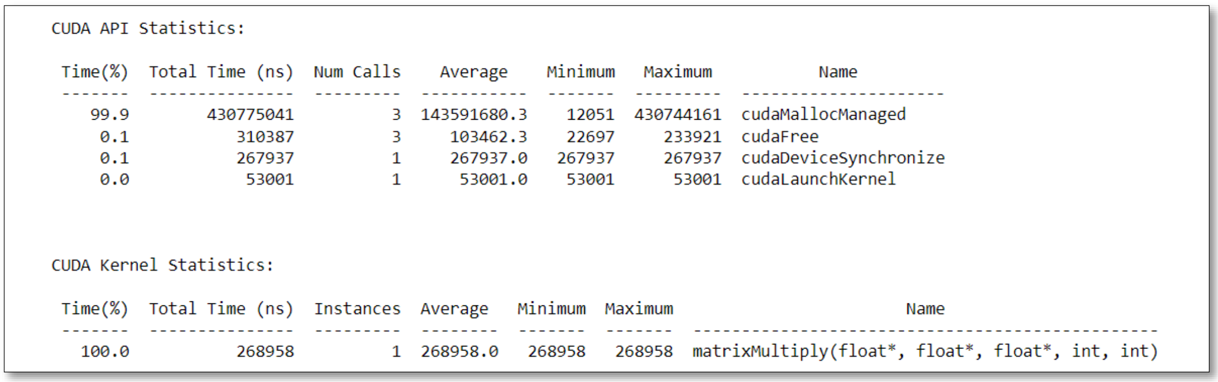
Serial Execution Time: 399933ns


```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum   Maximum           Name
 -------  ---------------  ---------  -----------  -------  ---------  ---------------------
    99.8        333656825          3  111218941.7     5627  333622273  cudaMallocManaged
     0.1           402410          1     402410.0   402410     402410  cudaDeviceSynchronize
     0.0           115065          3      38355.0    11233      71366  cudaFree
     0.0            38982          1      38982.0    38982      38982  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                      Name
 -------  ---------------  ---------  --------  -------  -------  --------------------------------------------------
   100.0           399933          1  399933.0   399933   399933  matrixMultiply(float*, float*, float*, int, int)
```

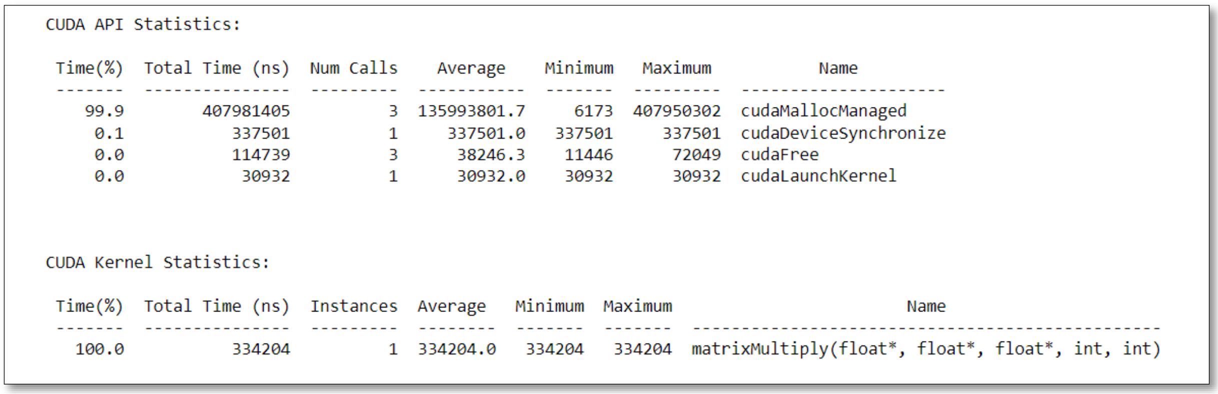
Parallel Execution Time:

Number of blocks	Thread per blocks	Time (in ns)	Speedup
16	512	268958	1.4869
16	1024	334204	1.1966
32	512	374204	1.0687
32	1024	332125	1.2041

Number of blocks: 16, Thread per blocks: 512, Execution Time: 268958

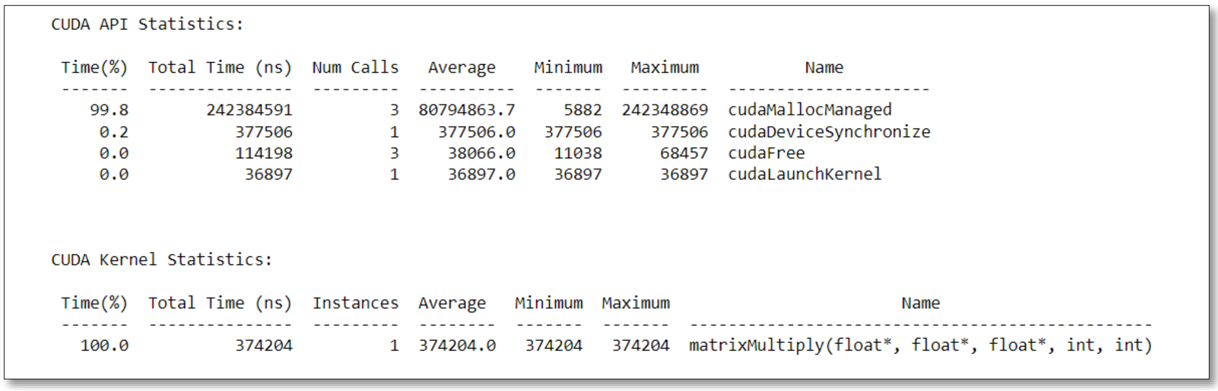

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum   Maximum           Name
 -------  ---------------  ---------  -----------  -------  ---------  ---------------------
    99.9        430775041          3  143591680.3    12051  430744161  cudaMallocManaged
     0.1           310387          3     103462.3    22697     233921  cudaFree
     0.1           267937          1     267937.0   267937     267937  cudaDeviceSynchronize
     0.0            53001          1      53001.0    53001      53001  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                      Name
 -------  ---------------  ---------  --------  -------  -------  --------------------------------------------------
   100.0           268958          1  268958.0   268958   268958  matrixMultiply(float*, float*, float*, int, int)
```

Number of blocks: 16, Thread per blocks: 1024, Execution Time: 334204



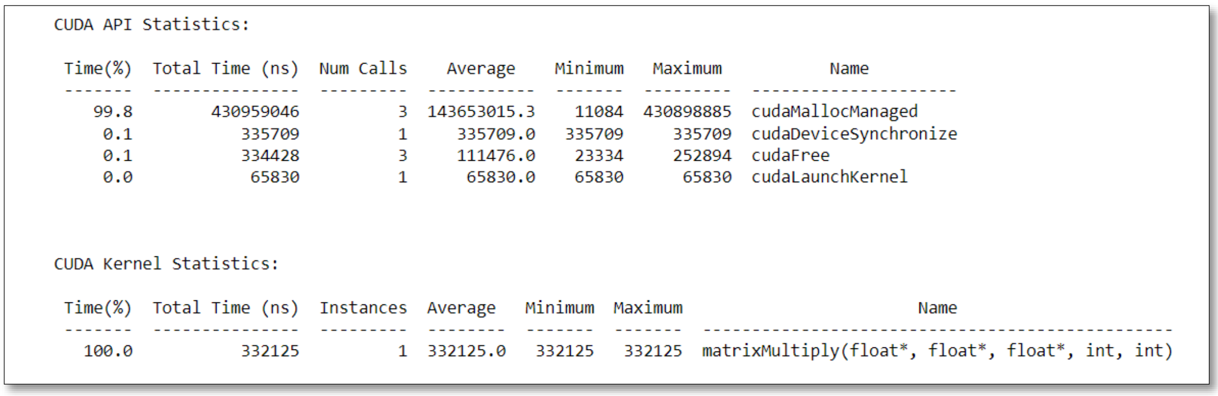
```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average    Minimum   Maximum           Name
 -------  ---------------  ---------  -----------  -------  ---------  ---------------------
    99.9        407981405          3  135993801.7     6173  407950302  cudaMallocManaged
     0.1           337501          1     337501.0   337501     337501  cudaDeviceSynchronize
     0.0           114739          3      38246.3    11446      72049  cudaFree
     0.0            30932          1      30932.0    30932      30932  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                        Name
 -------  ---------------  ---------  --------  -------  -------  ------------------------------------------------
   100.0           334204          1  334204.0   334204   334204  matrixMultiply(float*, float*, float*, int, int)
```

Number of blocks: 32, Thread per blocks: 512, Execution Time: 374204



```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  ---------------------
    99.8        242384591          3  80794863.7     5882  242348869  cudaMallocManaged
     0.2           377506          1    377506.0   377506     377506  cudaDeviceSynchronize
     0.0           114198          3     38066.0    11038      68457  cudaFree
     0.0            36897          1     36897.0    36897      36897  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                        Name
 -------  ---------------  ---------  --------  -------  -------  ------------------------------------------------
   100.0           374204          1  374204.0   374204   374204  matrixMultiply(float*, float*, float*, int, int)
```

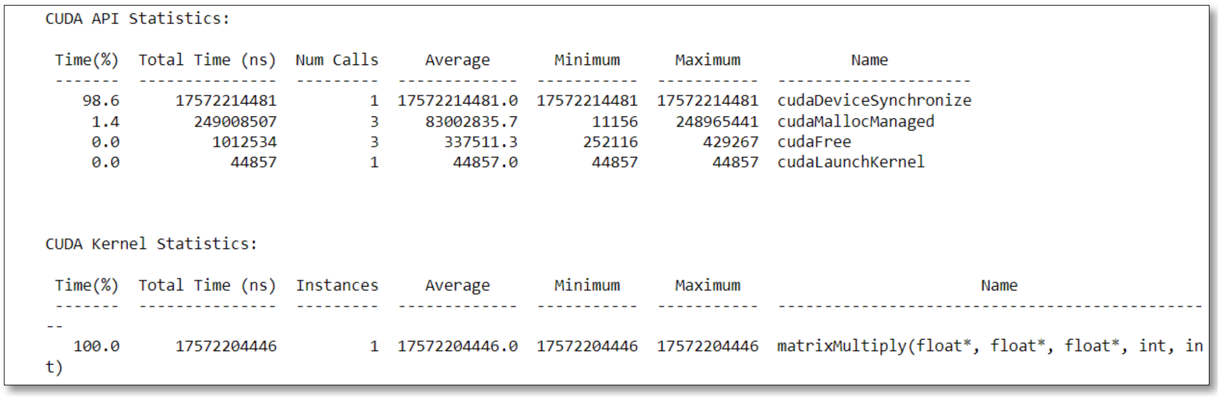
Number of blocks: 32, Thread per blocks: 1024, Execution Time: 332125



```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average    Minimum   Maximum           Name
 -------  ---------------  ---------  -----------  -------  ---------  ---------------------
    99.8        430959046          3  143653015.3    11084  430898885  cudaMallocManaged
     0.1           335709          1     335709.0   335709     335709  cudaDeviceSynchronize
     0.1           334428          3     111476.0    23334     252894  cudaFree
     0.0            65830          1      65830.0    65830      65830  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                        Name
 -------  ---------------  ---------  --------  -------  -------  ------------------------------------------------
   100.0           332125          1  332125.0   332125   332125  matrixMultiply(float*, float*, float*, int, int)
```

1024*1024 Matrix:

Serial Execution Time: 17572204446ns

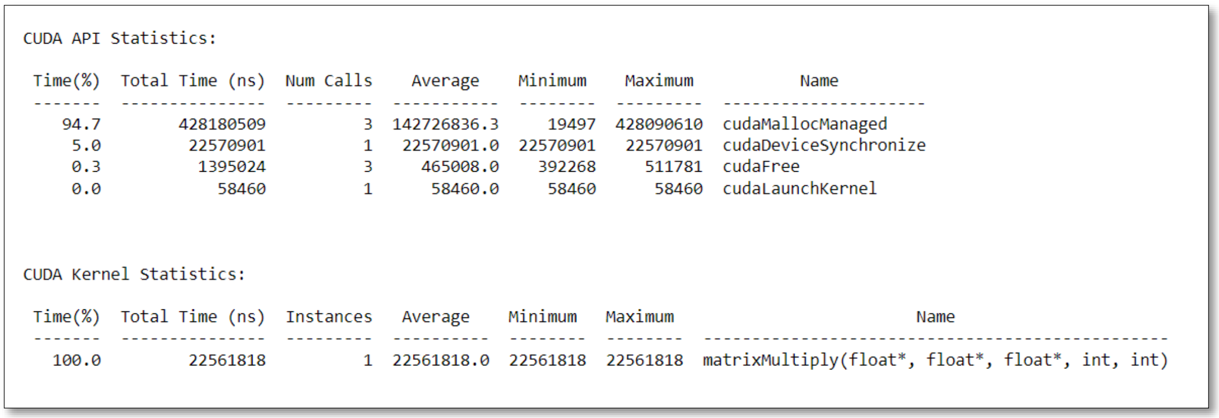


```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average       Minimum      Maximum            Name
 -------  ---------------  ---------  ------------  -----------  -----------  ---------------------
    98.6      17572214481          1  17572214481.0  17572214481  17572214481  cudaDeviceSynchronize
     1.4        249008507          3    83002835.7        11156    248965441  cudaMallocManaged
     0.0          1012534          3      337511.3       252116       429267  cudaFree
     0.0            44857          1       44857.0        44857        44857  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances    Average       Minimum      Maximum                        Name
 -------  ---------------  ---------  ------------  -----------  -----------  ---------------------------------------------
--
   100.0      17572204446          1  17572204446.0  17572204446  17572204446  matrixMultiply(float*, float*, float*, int, in
t)
```

Parallel Execution Time:

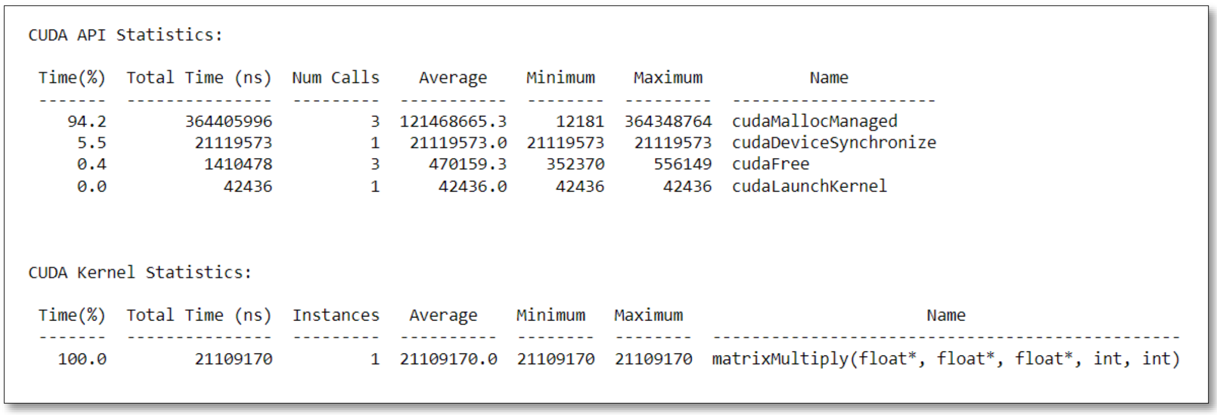
Number of blocks	Thread per blocks	Time (in ns)	Speedup
16	512	22561818	778.84
16	1024	21109170	832.44
32	512	15056306	1167.09
32	1024	14094717	1246.72

Number of blocks: 16, Thread per blocks: 512, Execution Time: 22561818ns



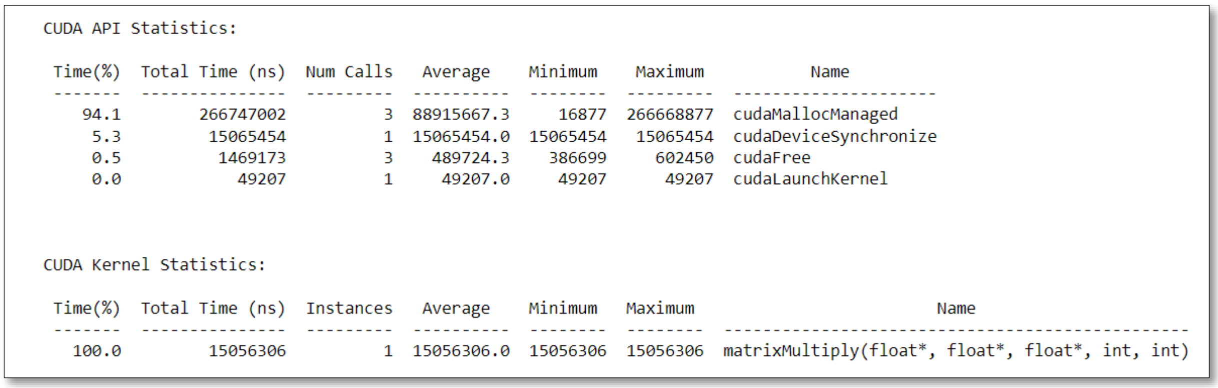
```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  --------  ---------  ---------------------
    94.7        428180509          3  142726836.3     19497  428090610  cudaMallocManaged
     5.0         22570901          1   22570901.0  22570901   22570901  cudaDeviceSynchronize
     0.3          1395024          3     465008.0    392268     511781  cudaFree
     0.0            58460          1      58460.0     58460      58460  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average     Minimum   Maximum                       Name
 -------  ---------------  ---------  ----------  --------  --------  ------------------------------------------------
   100.0         22561818          1  22561818.0  22561818  22561818  matrixMultiply(float*, float*, float*, int, int)
```

Number of blocks: 16, Thread per blocks: 1024, Execution Time: 21109170ns



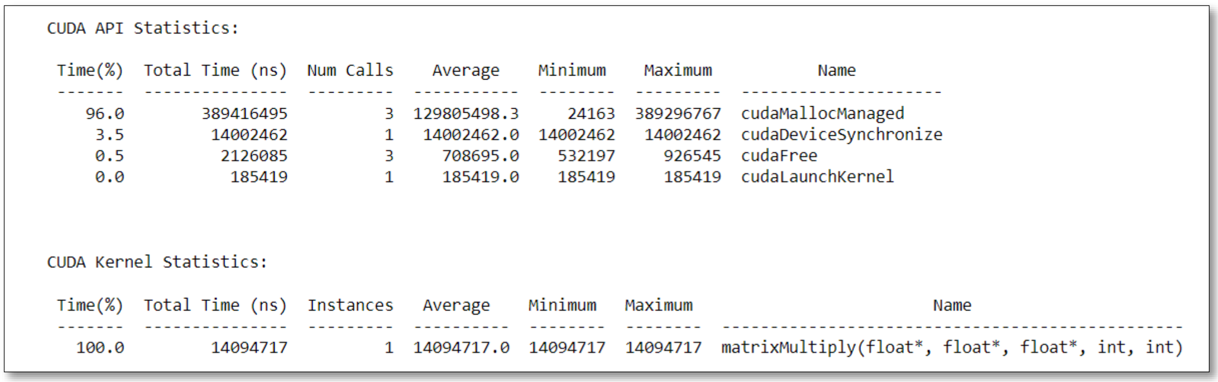
```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average    Minimum   Maximum           Name
 -------  ---------------  ---------  -----------  --------  ---------  ---------------------
    94.2        364405996          3  121468665.3     12181  364348764  cudaMallocManaged
     5.5         21119573          1   21119573.0  21119573   21119573  cudaDeviceSynchronize
     0.4          1410478          3     470159.3    352370     556149  cudaFree
     0.0            42436          1      42436.0     42436      42436  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum                         Name
 -------  ---------------  ---------  ----------  --------  --------  --------------------------------------------------
   100.0         21109170          1  21109170.0  21109170  21109170  matrixMultiply(float*, float*, float*, int, int)
```

Number of blocks: 32, Thread per blocks: 512, Execution Time: 15056306ns



```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  ---------------------
    94.1        266747002          3  88915667.3     16877  266668877  cudaMallocManaged
     5.3         15065454          1  15065454.0  15065454   15065454  cudaDeviceSynchronize
     0.5          1469173          3    489724.3    386699     602450  cudaFree
     0.0            49207          1     49207.0     49207      49207  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum                         Name
 -------  ---------------  ---------  ----------  --------  --------  --------------------------------------------------
   100.0         15056306          1  15056306.0  15056306  15056306  matrixMultiply(float*, float*, float*, int, int)
```

Number of blocks: 32, Thread per blocks: 1024, Execution Time: 14094717ns



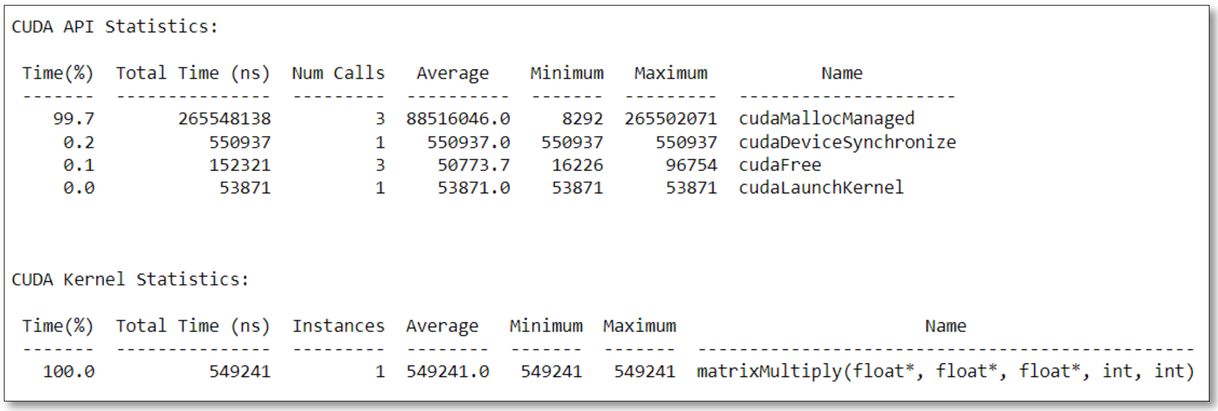
```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average    Minimum   Maximum           Name
 -------  ---------------  ---------  -----------  --------  ---------  ---------------------
    96.0        389416495          3  129805498.3     24163  389296767  cudaMallocManaged
     3.5         14002462          1   14002462.0  14002462   14002462  cudaDeviceSynchronize
     0.5          2126085          3     708695.0    532197     926545  cudaFree
     0.0           185419          1     185419.0    185419     185419  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum                         Name
 -------  ---------------  ---------  ----------  --------  --------  --------------------------------------------------
   100.0         14094717          1  14094717.0  14094717  14094717  matrixMultiply(float*, float*, float*, int, int)
```

2. Implement Matrix-Matrix Multiplication using shared memory in CUDA C. Analyze and tune the program for getting maximum speed up. Do Profiling and state what part of the code takes the huge amount of time to execute.

```
#include <stdio.h>

void initWith(float num, float *a, int SIZE)

{

  for(int i = 0; i < SIZE; ++i)

  {

    a[i] = num;

  }

}

__global__

void matrixMultiply(float *result, float *a, float *b, int N, int SIZE)

{

  __shared__ int stride;

  if (threadIdx.x == 0)

    stride = gridDim.x * blockDim.x;

  __syncthreads();

  int start = blockIdx.x * blockDim.x + threadIdx.x;

  for(int i = start; i < SIZE; i += stride)

  {

    int row = i / N;
    float sum = 0;
    for (int j = 0; j < N; j++)
    {
      sum += a[row * N + j] * b[N * j + row];
    }
    result[i] = sum;
  }
}
void checkElementsAre(float target, float *array, int SIZE)
{
  for(int i = 0; i < SIZE; i++)
  {
    if(array[i] != target)
    {
      printf("FAIL: array[%d] - %0.0f does not equal %0.0f\n", i, array[i], target);
      exit(1);
    }
  }
  printf("SUCCESS! All values multiplied correctly.\n");
}
int main()
{
  const int N = 1024;

  const int SIZE = N * N; // sqaure matrix

  size_t size = SIZE * sizeof(float);



  float *a;

  float *b;

  float *c;



  cudaMallocManaged(&a, size);

  cudaMallocManaged(&b, size);

  cudaMallocManaged(&c, size);



  initWith(3, a, SIZE);

  initWith(4, b, SIZE);

  initWith(0, c, SIZE);

  matrixMultiply<<<1, 1>>>(c, a, b, N, SIZE);

  cudaDeviceSynchronize();

  checkElementsAre(12288, c, SIZE);

  cudaFree(a);

  cudaFree(b);

  cudaFree(c);

}
```

8*8 Matrix:

Serial Execution Time: 549241ns

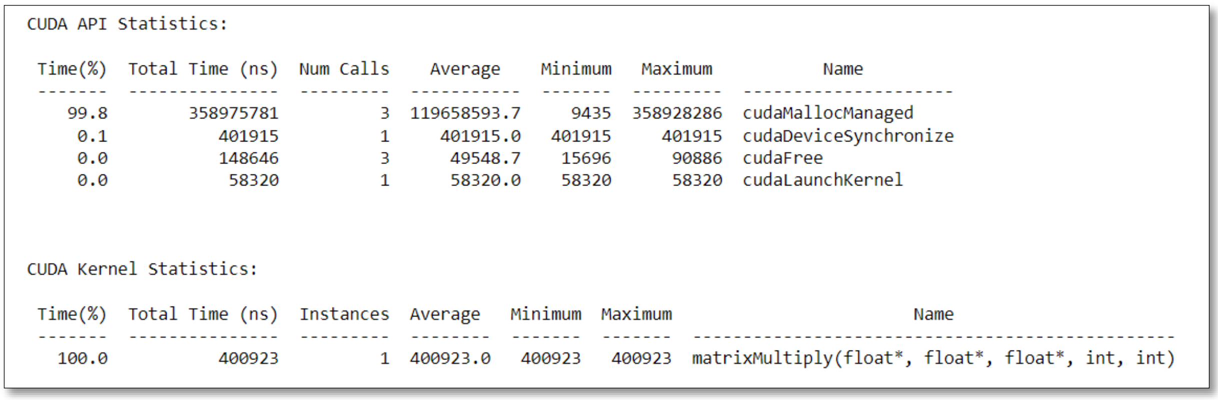


```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  ---------------------
    99.7        265548138          3  88516046.0     8292  265502071  cudaMallocManaged
     0.2           550937          1    550937.0   550937     550937  cudaDeviceSynchronize
     0.1           152321          3     50773.7    16226      96754  cudaFree
     0.0            53871          1     53871.0    53871      53871  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                      Name
 -------  ---------------  ---------  --------  -------  -------  ---------------------------------------------------
   100.0           549241          1  549241.0   549241   549241  matrixMultiply(float*, float*, float*, int, int)
```

Parallel Execution Time:

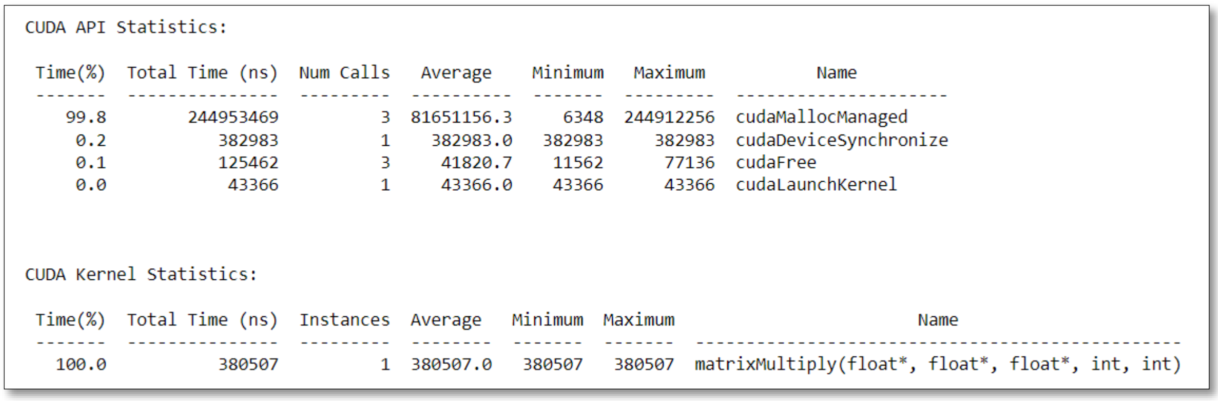
Number of blocks	Thread per blocks	Time (in ns)	Speedup
16	512	400923	1.3699
16	1024	380507	1.4434
32	512	481114	1.1416
32	1024	388795	1.4126

Number of blocks: 16, Thread per blocks: 512, Execution Time: 400923ns

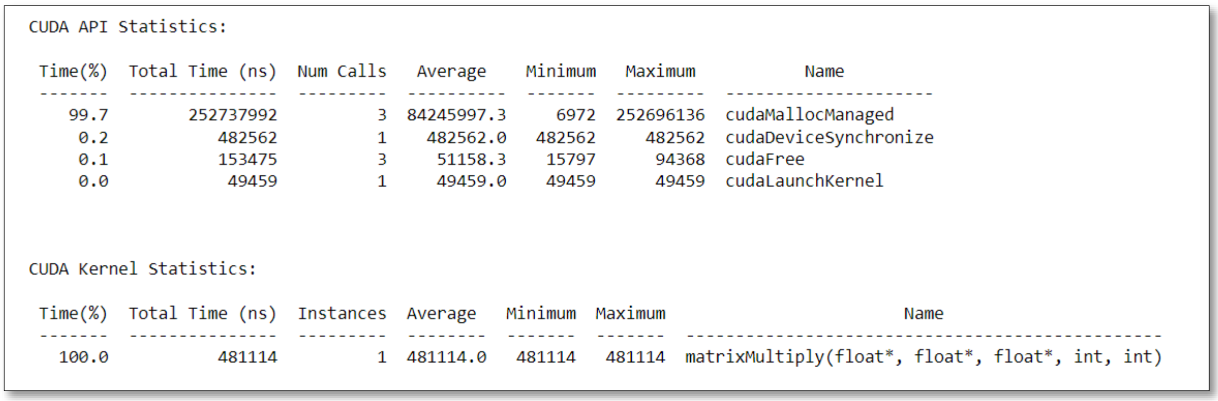


```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average    Minimum   Maximum           Name
 -------  ---------------  ---------  -----------  -------  ---------  ---------------------
    99.8        358975781          3  119658593.7     9435  358928286  cudaMallocManaged
     0.1           401915          1     401915.0   401915     401915  cudaDeviceSynchronize
     0.0           148646          3      49548.7    15696      90886  cudaFree
     0.0            58320          1      58320.0    58320      58320  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                      Name
 -------  ---------------  ---------  --------  -------  -------  ---------------------------------------------------
   100.0           400923          1  400923.0   400923   400923  matrixMultiply(float*, float*, float*, int, int)
```

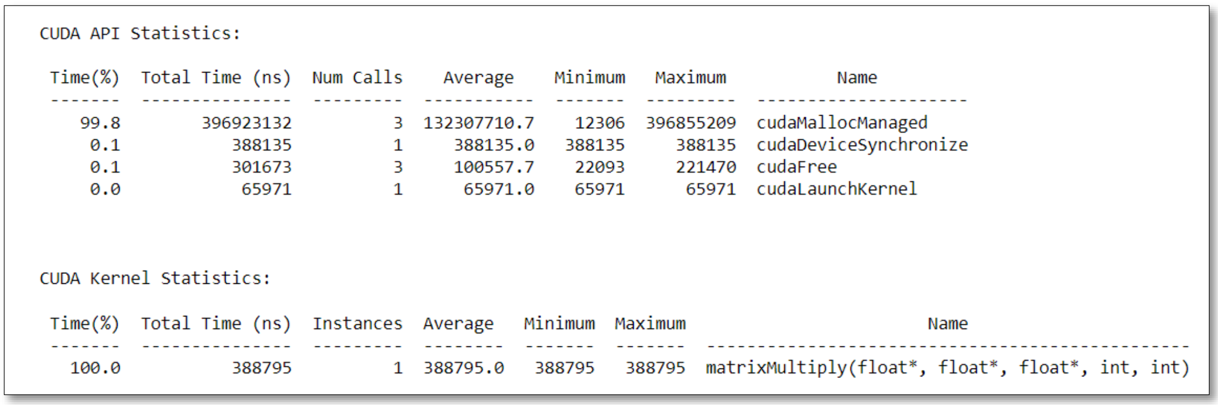
Number of blocks: 16, Thread per blocks: 1024, Execution Time: 380507ns


```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  ---------------------
    99.8        244953469          3  81651156.3     6348  244912256  cudaMallocManaged
     0.2           382983          1    382983.0   382983     382983  cudaDeviceSynchronize
     0.1           125462          3     41820.7    11562      77136  cudaFree
     0.0            43366          1     43366.0    43366      43366  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                         Name
 -------  ---------------  ---------  --------  -------  -------  --------------------------------------------------
   100.0           380507          1  380507.0   380507   380507  matrixMultiply(float*, float*, float*, int, int)
```

Number of blocks: 32, Thread per blocks: 512, Execution Time: 481114ns

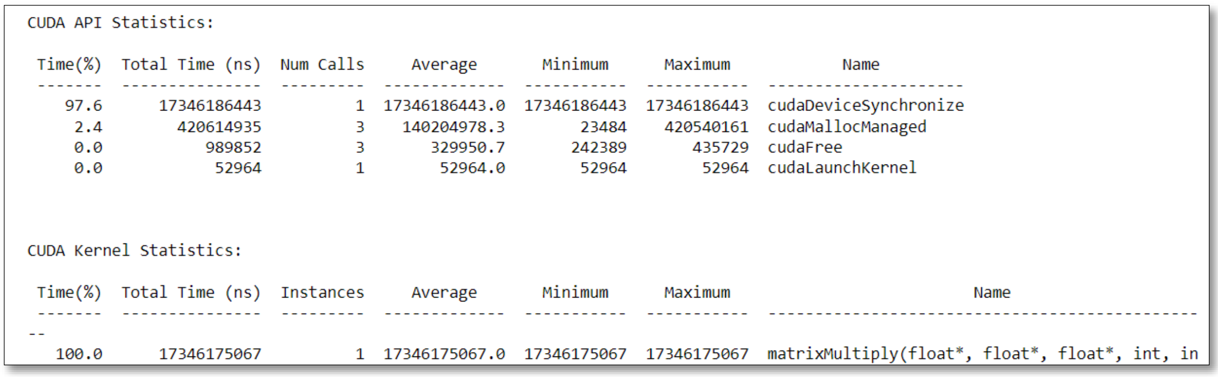

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  ---------------------
    99.7        252737992          3  84245997.3     6972  252696136  cudaMallocManaged
     0.2           482562          1    482562.0   482562     482562  cudaDeviceSynchronize
     0.1           153475          3     51158.3    15797      94368  cudaFree
     0.0            49459          1     49459.0    49459      49459  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                         Name
 -------  ---------------  ---------  --------  -------  -------  --------------------------------------------------
   100.0           481114          1  481114.0   481114   481114  matrixMultiply(float*, float*, float*, int, int)
```

Number of blocks: 32, Thread per blocks: 1024, Execution Time: 388795ns


```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average    Minimum   Maximum           Name
 -------  ---------------  ---------  -----------  -------  ---------  ---------------------
    99.8        396923132          3  132307710.7    12306  396855209  cudaMallocManaged
     0.1           388135          1     388135.0   388135     388135  cudaDeviceSynchronize
     0.1           301673          3     100557.7    22093     221470  cudaFree
     0.0            65971          1      65971.0    65971      65971  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                         Name
 -------  ---------------  ---------  --------  -------  -------  --------------------------------------------------
   100.0           388795          1  388795.0   388795   388795  matrixMultiply(float*, float*, float*, int, int)
```

1024*1024 Matrix:

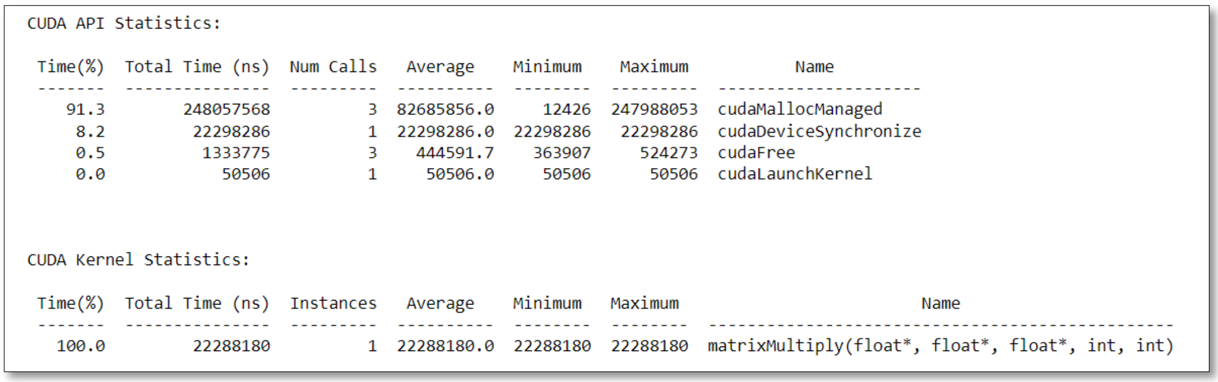
Serial Execution Time: 17346175067ns


```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls     Average       Minimum      Maximum            Name
 -------  ---------------  ---------  -------------  -----------  -----------  ---------------------
    97.6      17346186443          1  17346186443.0  17346186443  17346186443  cudaDeviceSynchronize
     2.4        420614935          3    140204978.3        23484    420540161  cudaMallocManaged
     0.0           989852          3       329950.7       242389       435729  cudaFree
     0.0            52964          1        52964.0        52964        52964  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances     Average       Minimum      Maximum                        Name
 -------  ---------------  ---------  -------------  -----------  -----------  ----------------------------------------------
--
   100.0      17346175067          1  17346175067.0  17346175067  17346175067  matrixMultiply(float*, float*, float*, int, in
```

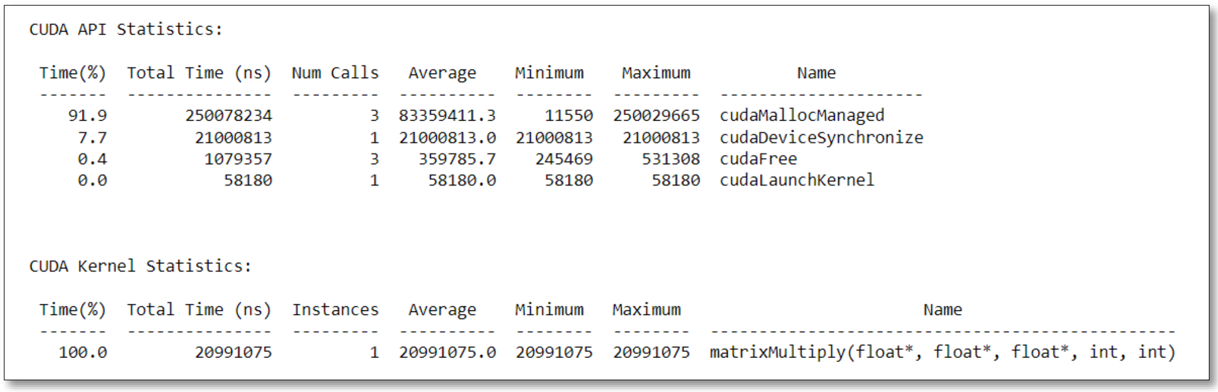
Parallel Execution Time:

Number of blocks	**Thread per blocks**	**Time (in ns)**	**Speedup**
16	512	22288180	778.26
16	1024	20991075	826.35
32	512	15596802	1049.72
32	1024	14524471	1112.16

Number of blocks: 16, Thread per blocks: 512, Execution Time: 22288180ns

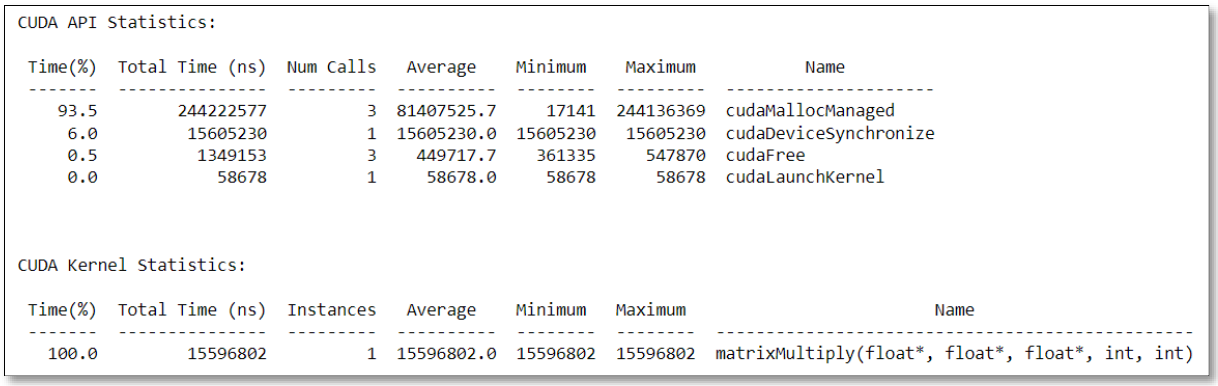

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  ---------------------
    91.3        248057568          3  82685856.0     12426  247988053  cudaMallocManaged
     8.2         22298286          1  22298286.0  22298286   22298286  cudaDeviceSynchronize
     0.5          1333775          3    444591.7    363907     524273  cudaFree
     0.0            50506          1     50506.0     50506      50506  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances    Average     Minimum   Maximum                        Name
 -------  ---------------  ---------  ----------  --------  --------  ----------------------------------------------
   100.0         22288180          1  22288180.0  22288180  22288180  matrixMultiply(float*, float*, float*, int, int)
```

Number of blocks: 16, Thread per blocks: 1024, Execution Time: 20991075ns



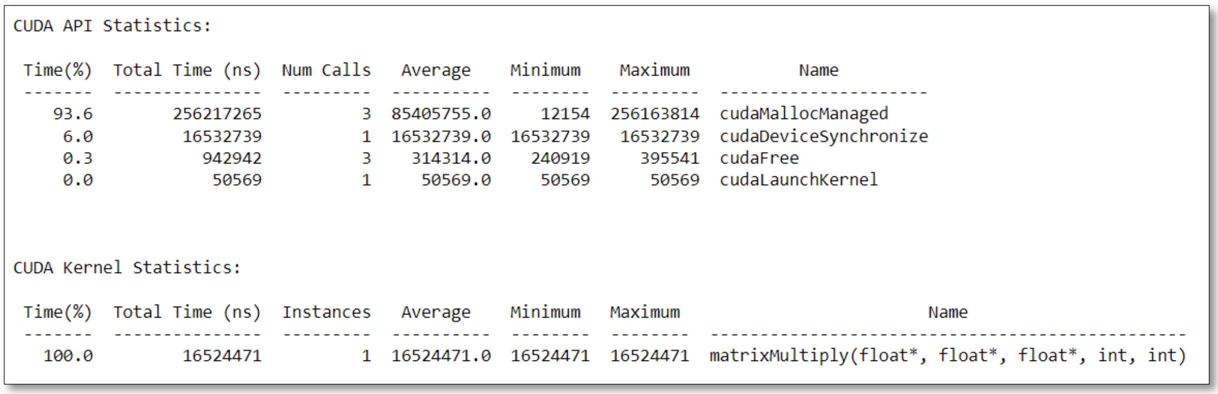
```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum    Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  ---------------------
    91.9        250078234          3  83359411.3     11550  250029665  cudaMallocManaged
     7.7         21000813          1  21000813.0  21000813   21000813  cudaDeviceSynchronize
     0.4          1079357          3    359785.7    245469     531308  cudaFree
     0.0            58180          1     58180.0     58180      58180  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum                         Name
 -------  ---------------  ---------  ----------  --------  --------  ------------------------------------------------
   100.0         20991075          1  20991075.0  20991075  20991075  matrixMultiply(float*, float*, float*, int, int)
```

Number of blocks: 32, Thread per blocks: 512, Execution Time: 15596802ns



```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum    Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  ---------------------
    93.5        244222577          3  81407525.7     17141  244136369  cudaMallocManaged
     6.0         15605230          1  15605230.0  15605230   15605230  cudaDeviceSynchronize
     0.5          1349153          3    449717.7    361335     547870  cudaFree
     0.0            58678          1     58678.0     58678      58678  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum                         Name
 -------  ---------------  ---------  ----------  --------  --------  ------------------------------------------------
   100.0         15596802          1  15596802.0  15596802  15596802  matrixMultiply(float*, float*, float*, int, int)
```

Number of blocks: 32, Thread per blocks: 1024, Execution Time: 16524471ns



```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum    Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  ---------------------
    93.6        256217265          3  85405755.0     12154  256163814  cudaMallocManaged
     6.0         16532739          1  16532739.0  16532739   16532739  cudaDeviceSynchronize
     0.3           942942          3    314314.0    240919     395541  cudaFree
     0.0            50569          1     50569.0     50569      50569  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum                         Name
 -------  ---------------  ---------  ----------  --------  --------  ------------------------------------------------
   100.0         16524471          1  16524471.0  16524471  16524471  matrixMultiply(float*, float*, float*, int, int)
```

3. Implement Prefix sum using CUDA C. Analyze and tune the program for getting maximum speed up. Do Profiling and state what part of the code takes the huge amount of time to execute.

```
#include <stdio.h>

void initWith(float val, float *arr, int N)

{

  for (int i = 0; i < N; i++)

  {

    arr[i] = val;

  }

}

__global__

void prefixSum(float *arr, float *res, float *ptemp, float* ttemp, int N)

{

  int threadId = blockIdx.x * blockDim.x + threadIdx.x;

  int totalThreads = gridDim.x * blockDim.x;

  int elementsPerThread = ceil(1.0 * N / totalThreads);

  int start = threadId * elementsPerThread;

  int count = 0;

  float *sums = new float[elementsPerThread];

  float sum = 0;

  for (int i = start; i < N && count < elementsPerThread; i++, count++) {
```

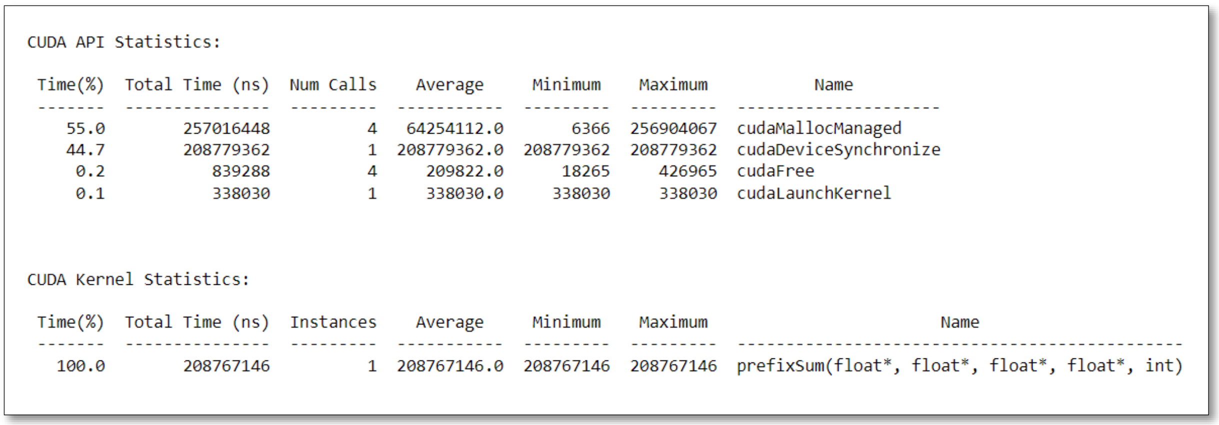
```
  sum += arr[i];
  sums[count] = sum;
}
float localSum;
if (count)
  localSum = sums[count - 1];
else
  localSum = 0;
ptemp[threadId] = localSum;
ttemp[threadId] = localSum;
__syncthreads();
if (totalThreads == 1) {
  for (int i = 0; i < N; i++)
    res[i] = sums[i];
} else {
  int d = 0; // log2(totalThreads)
  int x = totalThreads;
  while (x > 1) {
    d++;
    x = x >> 1;
  }
```

```
    x = 1;
    for (int i = 0; i < 2*d; i++) {
      int tsum = ttemp[threadId];
      __syncthreads();
      int newId = threadId / x;
      if (newId % 2 == 0) {
        int nextId = threadId + x;
        ptemp[nextId] += tsum;
        ttemp[nextId] += tsum;
      } else {
        int nextId = threadId - x;
        ttemp[nextId] += tsum;
      }
      x = x << 1;
    }
    __syncthreads();
    float diff = ptemp[threadId] - localSum;
    for (int i = start, j = 0; i < N && j < count; i++, j++) {
      res[i] = sums[j] + diff;
    }
  }
}
```

```
void checkRes(float *arr, float *res, int N, float *ptemp, float* ttemp)
{
  float sum = 0;
  for (int i = 0; i < N; i++)
  {
    sum += arr[i];
    if (sum != res[i])
    {
      printf("FAIL: res[%d] - %0.0f does not equal %0.0f\n", i, res[i], sum);
      exit(1);
    }
  }
  printf("SUCCESS! All prefix sums added correctly.\n");
}
int main()
{
  const int N = 1000000;
  size_t size = N * sizeof(float);
  float *arr;
  float *res;
  cudaMallocManaged(&arr, size);
  cudaMallocManaged(&res, size);
  initWith(2, arr, N);

  initWith(0, res, N);

  int blocks = 1;

  int threadsPerBlock = 1;

  int totalThreads = blocks * threadsPerBlock;

  float *ptemp;

  float *ttemp;

  cudaMallocManaged(&ptemp, totalThreads * sizeof(float));

  cudaMallocManaged(&ttemp, totalThreads * sizeof(float));


  prefixSum<<<blocks, threadsPerBlock>>>(arr, res, ptemp, ttemp, N);

  cudaDeviceSynchronize();

  checkRes(arr, res, N, ptemp, ttemp);


  cudaFree(arr);

  cudaFree(res);

  cudaFree(ttemp);

  cudaFree(ptemp);

}
```

Element 10^6

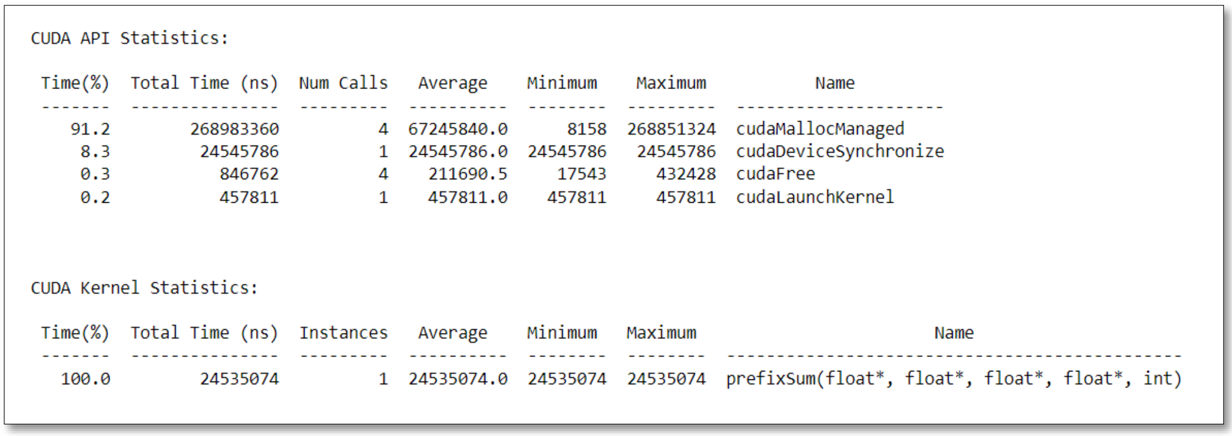
Serial Execution Time: 208767146ns


```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  ---------  ---------  ---------------------
    55.0        257016448          4   64254112.0       6366  256904067  cudaMallocManaged
    44.7        208779362          1  208779362.0  208779362  208779362  cudaDeviceSynchronize
     0.2           839288          4     209822.0      18265     426965  cudaFree
     0.1           338030          1     338030.0     338030     338030  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances    Average     Minimum    Maximum                       Name
 -------  ---------------  ---------  -----------  ---------  ---------  ---------------------------------------------
   100.0        208767146          1  208767146.0  208767146  208767146  prefixSum(float*, float*, float*, float*, int)
```

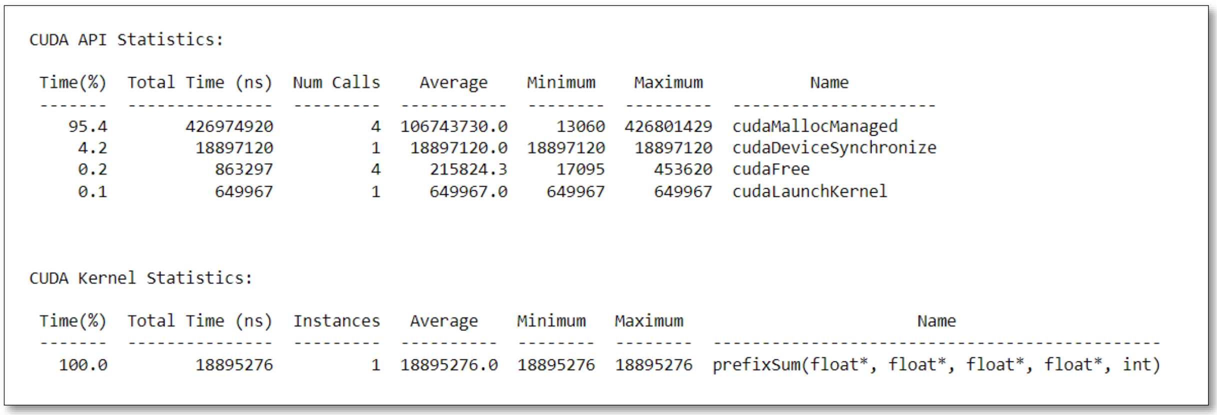
Parallel Execution Time:

Number of Blocks	Threads per Block	Time(ns)	Speedup
1	32	24535074	8.5089
1	64	18895276	11.0486
1	128	17276600	12.0838

Number of Blocks: 1, Thread per Blocks: 32, Execution Time: 24535074ns

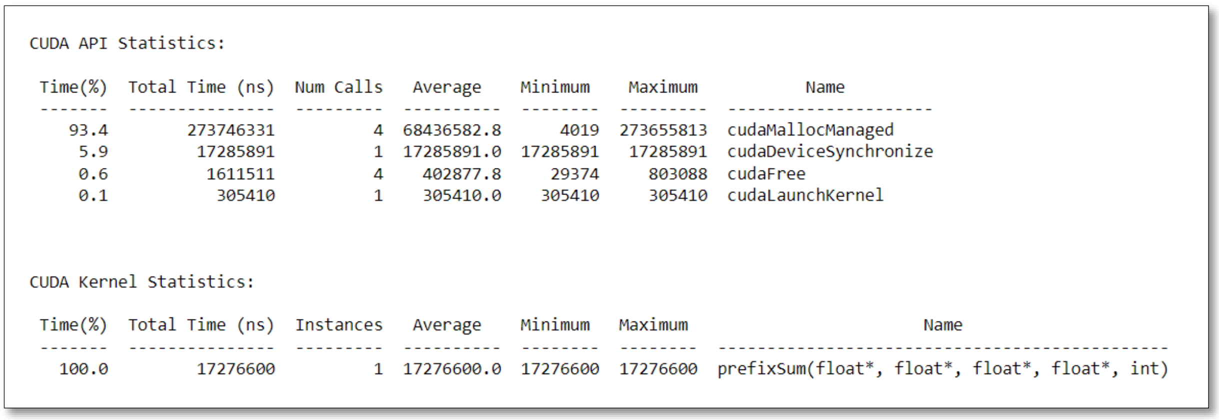

```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  ---------------------
    91.2        268983360          4  67245840.0      8158  268851324  cudaMallocManaged
     8.3         24545786          1  24545786.0  24545786   24545786  cudaDeviceSynchronize
     0.3           846762          4    211690.5     17543     432428  cudaFree
     0.2           457811          1    457811.0    457811     457811  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances    Average    Minimum   Maximum                       Name
 -------  ---------------  ---------  ----------  --------  --------  ---------------------------------------------
   100.0         24535074          1  24535074.0  24535074  24535074  prefixSum(float*, float*, float*, float*, int)
```

Number of Blocks: 1, Thread per Blocks: 64, Execution Time: 18895276ns



```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum    Maximum           Name
 -------  ---------------  ---------  -----------  --------  ---------  ---------------------
    95.4        426974920          4  106743730.0     13060  426801429  cudaMallocManaged
     4.2         18897120          1   18897120.0  18897120   18897120  cudaDeviceSynchronize
     0.2           863297          4     215824.3     17095     453620  cudaFree
     0.1           649967          1     649967.0    649967     649967  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum                      Name
 -------  ---------------  ---------  ----------  --------  --------  ----------------------------------------------
   100.0         18895276          1  18895276.0  18895276  18895276  prefixSum(float*, float*, float*, float*, int)
```

Number of Blocks: 1, Thread per Blocks: 128, Execution Time: 17276600ns

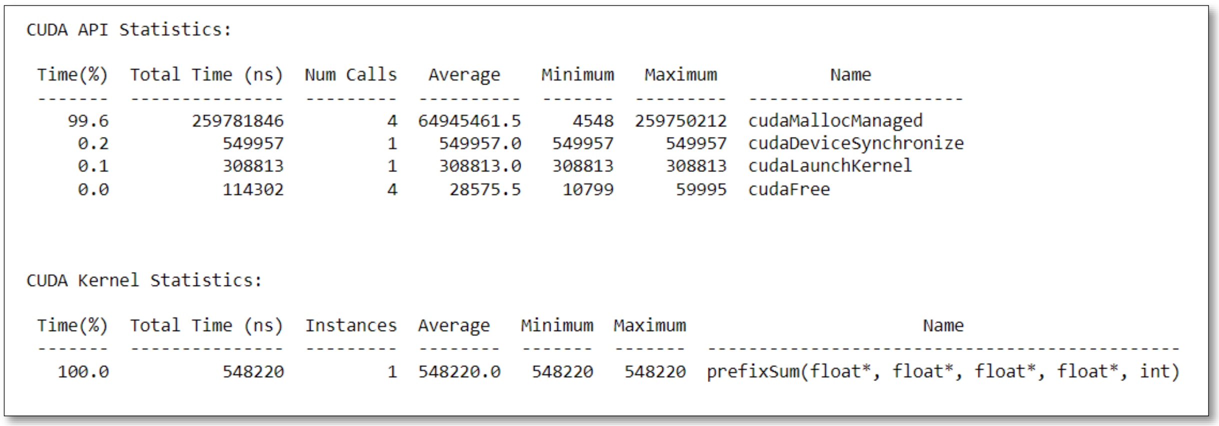


```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average     Minimum    Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  ---------------------
    93.4        273746331          4  68436582.8      4019  273655813  cudaMallocManaged
     5.9         17285891          1  17285891.0  17285891   17285891  cudaDeviceSynchronize
     0.6          1611511          4    402877.8     29374     803088  cudaFree
     0.1           305410          1    305410.0    305410     305410  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum                      Name
 -------  ---------------  ---------  ----------  --------  --------  ----------------------------------------------
   100.0         17276600          1  17276600.0  17276600  17276600  prefixSum(float*, float*, float*, float*, int)
```

Parallel Execution Time:

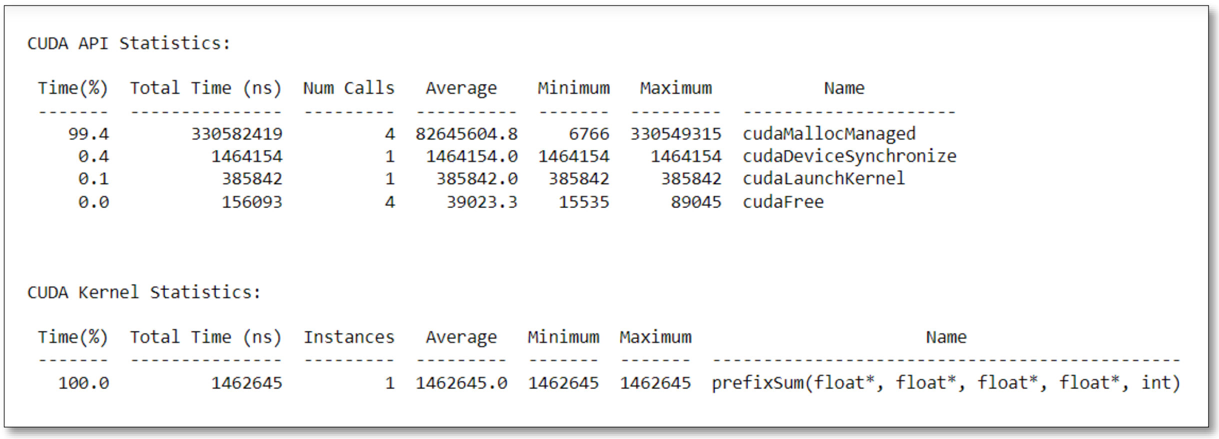
Elements	Number of Blocks	Threads per Block	Time(ns)
10^3	1	64	548220
10^4	1	64	1462645
10^5	1	64	2765066
10^6	1	64	19337795

10^3, Execution time: 548220ns



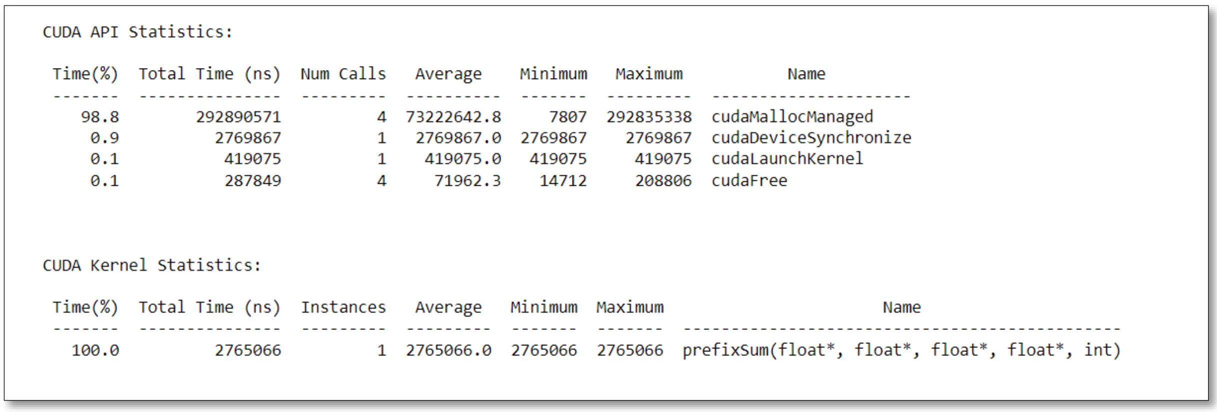
```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum   Maximum           Name
 -------  ---------------  ---------  ----------   -------  ---------  ---------------------
    99.6        259781846          4  64945461.5      4548  259750212  cudaMallocManaged
     0.2           549957          1    549957.0    549957     549957  cudaDeviceSynchronize
     0.1           308813          1    308813.0    308813     308813  cudaLaunchKernel
     0.0           114302          4     28575.5     10799      59995  cudaFree


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                      Name
 -------  ---------------  ---------  --------  -------  -------  ----------------------------------------------
   100.0           548220          1  548220.0   548220   548220  prefixSum(float*, float*, float*, float*, int)
```

10^4, Execution time: 1462645ns



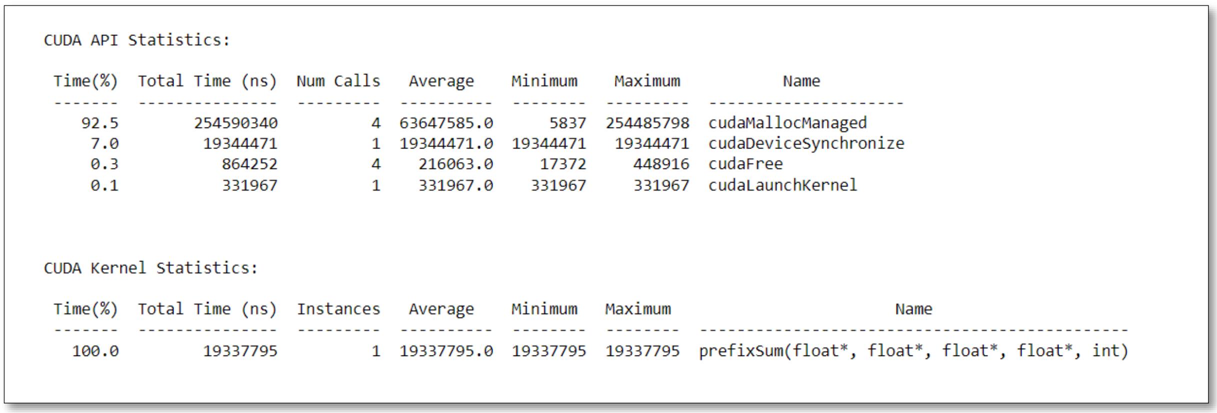
```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum   Maximum           Name
 -------  ---------------  ---------  ----------   -------  ---------  ---------------------
    99.4        330582419          4  82645604.8      6766  330549315  cudaMallocManaged
     0.4          1464154          1   1464154.0   1464154    1464154  cudaDeviceSynchronize
     0.1           385842          1    385842.0    385842     385842  cudaLaunchKernel
     0.0           156093          4     39023.3     15535      89045  cudaFree


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum                      Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------------------------
   100.0          1462645          1  1462645.0  1462645  1462645  prefixSum(float*, float*, float*, float*, int)
```

10^5, Execution time: 2765066ns



```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  ---------------------
    98.8        292890571          4  73222642.8     7807  292835338  cudaMallocManaged
     0.9          2769867          1   2769867.0  2769867    2769867  cudaDeviceSynchronize
     0.1           419075          1    419075.0   419075     419075  cudaLaunchKernel
     0.1           287849          4     71962.3    14712     208806  cudaFree


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average   Minimum  Maximum                      Name
 -------  ---------------  ---------  ---------  -------  -------  ----------------------------------------------
   100.0          2765066          1  2765066.0  2765066  2765066  prefixSum(float*, float*, float*, float*, int)
```

10^6, Execution time: 19337795ns



```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum    Maximum           Name
 -------  ---------------  ---------  ----------  --------  ---------  ---------------------
    92.5        254590340          4  63647585.0      5837  254485798  cudaMallocManaged
     7.0         19344471          1  19344471.0  19344471   19344471  cudaDeviceSynchronize
     0.3           864252          4    216063.0     17372     448916  cudaFree
     0.1           331967          1    331967.0    331967     331967  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances   Average    Minimum   Maximum                    Name
 -------  ---------------  ---------  ----------  --------  --------  ----------------------------------------------
   100.0         19337795          1  19337795.0  19337795  19337795  prefixSum(float*, float*, float*, float*, int)
```

4. Implement 2D Convolution using shared memory using CUDA C. Analyze and tune the program for getting maximum speed up. Do Profiling and state what part of the code takes the huge amount of time to execute.

```
#include <stdio.h>

#define MASK_DIM 7

#define MASK_OFFSET (MASK_DIM / 2

__constant__ int mask[7 * 7];

__global__ void convolution_2d(int *matrix, int *result, int N)

{

    // Calculate the global thread positions

    int row = blockIdx.y * blockDim.y + threadIdx.y;

    int col = blockIdx.x * blockDim.x + threadIdx.x;

    // Starting index for calculation

    int start_r = row - MASK_OFFSET;

    int start_c = col - MASK_OFFSET;
```

```
// Temp value for accumulating the result
int temp = 0;
// Iterate over all the rows
for (int i = 0; i < MASK_DIM; i++)
{
    // Go over each column
    for (int j = 0; j < MASK_DIM; j++)
    {
        // Range check for rows
        if ((start_r + i) >= 0 && (start_r + i) < N)
        {
            // Range check for columns
            if ((start_c + j) >= 0 && (start_c + j) < N)
            {
                // Accumulate result
                temp += matrix[(start_r + i) * N + (start_c + j)] * mask[i * MASK_DIM + j];
            }
        }
    }
}
// Write back the result
result[row * N + col] = temp;
```

```
}
void init_matrix(int *m, int n)
{
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
        {
            m[n * i + j] = rand() % 100;
        }
    }
}
void verify_result(int *m, int *mask, int *result, int N)
{
    int temp;
    int offset_r;
    int offset_c;
    // Go over each row
    for (int i = 0; i < N; i++)
    {
        // Go over each column
        for (int j = 0; j < N; j++)
        {
```

```
// Reset the temp variable
temp = 0;
// Go over each mask row
for (int k = 0; k < MASK_DIM; k++)
{
    // Update offset value for row
    offset_r = i - MASK_OFFSET + k;
    // Go over each mask column
    for (int l = 0; l < MASK_DIM; l++)
    {
        // Update offset value for column
        offset_c = j - MASK_OFFSET + l;
        // Range checks if we are hanging off the matrix
        if (offset_r >= 0 && offset_r < N)
        {
            if (offset_c >= 0 && offset_c < N)
            {
                // Accumulate partial results
                temp += m[offset_r * N + offset_c] * mask[k * MASK_DIM + l];
            }
        }
    }
```

```
            }
            // Fail if the results don't match
            if (result[i * N + j] != temp)
            {
                printf("Check failed");
                return;
            }
        }
    }
}

int main()
{
    int N = 1 << 10; // 2^10

    size_t bytes_n = N * N * sizeof(int);
    size_t bytes_m = MASK_DIM * MASK_DIM * sizeof(int);

    int *matrix;
    int *result;
    int *h_mask;
```

```
cudaMallocManaged(&matrix, bytes_n);

cudaMallocManaged(&result, bytes_n);

cudaMallocManaged(&h_mask, bytes_m);


init_matrix(matrix, N);

init_matrix(mask, MASK_DIM);


cudaMemcpyToSymbol(mask, h_mask, bytes_m);


// Calculate grid dimensions

int THREADS = 1;

int BLOCKS = (N + THREADS - 1) / THREADS;


// Dimension launch arguments

dim3 block_dim(THREADS, THREADS);

dim3 grid_dim(BLOCKS, BLOCKS);


convolution_2d<<<grid_dim, block_dim>>>(matrix, result, N);


verify_result(matrix, h_mask, result, N);


printf("COMPLETED SUCCESSFULLY!");
```

```
    cudaFree(matrix);

    cudaFree(result);

    cudaFree(h_mask);

    return 0;

}
```

Execution Time:

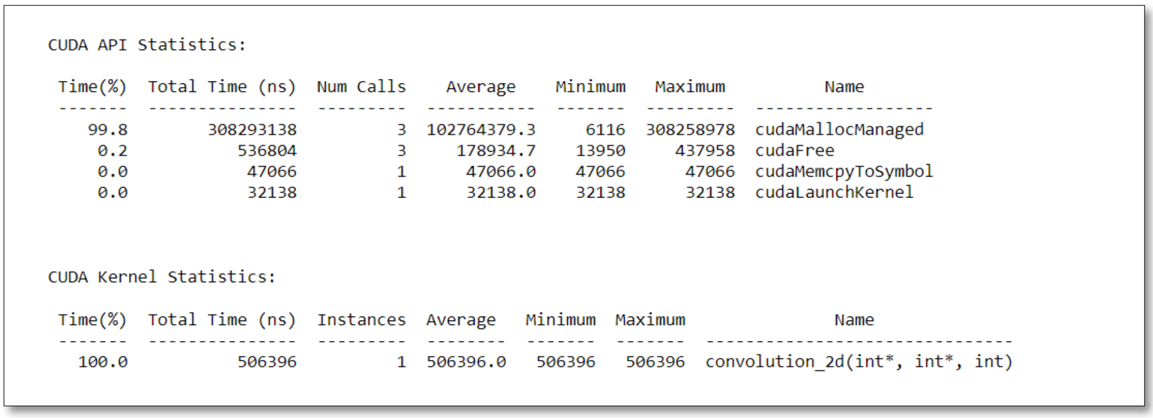
Threads	**2^4*2^4**	**Speedup**
Serial Execution:	506396	-
4	638810	0.7927
8	382717	1.3231
16	531931	0.9519
32	517883	0.9778

Threads	**2^5*2^5**	**Speedup**
Serial Execution:	769561	-
4	564605	1.3630
8	452604	1.7002
16	439085	1.7526
32	400955	1.9193

Threads	2^10*2^10	Speedup
Serial Execution:	14161311	-
4	4514933	3.1365
8	4055855	3.4915
16	460935	30.7230
32	432174	32.7676

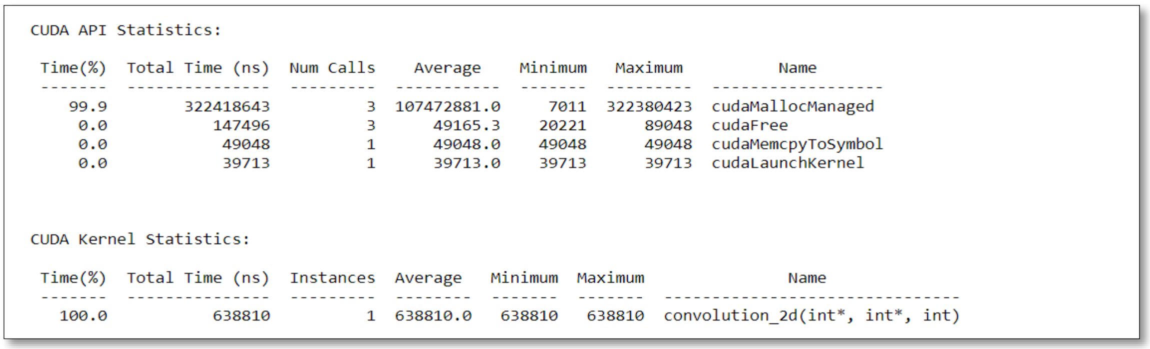
2^4

Thread=1, Execution Time: 506396ns



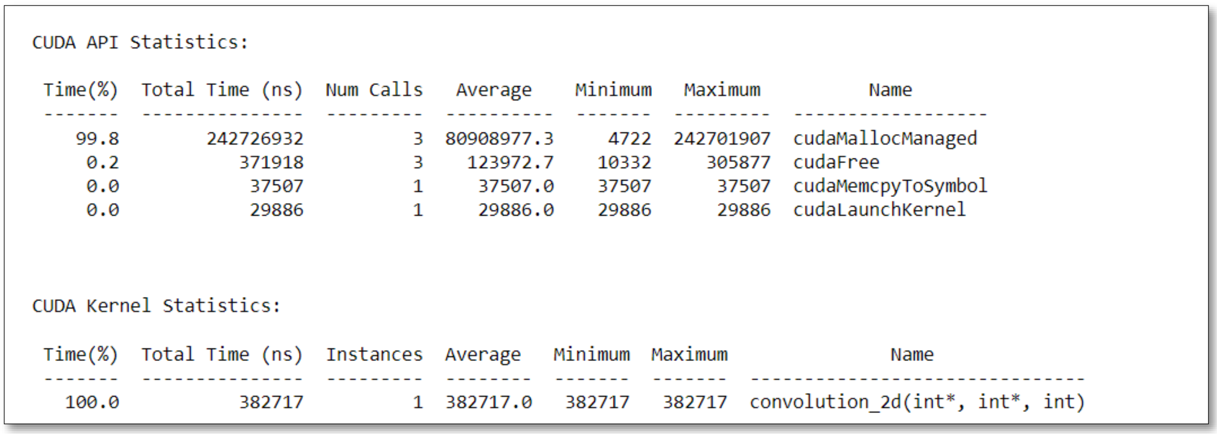
```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum   Maximum          Name
 -------  ---------------  ---------  -----------  -------  ---------  ------------------
    99.8        308293138          3  102764379.3     6116  308258978  cudaMallocManaged
     0.2           536804          3     178934.7    13950     437958  cudaFree
     0.0            47066          1      47066.0    47066      47066  cudaMemcpyToSymbol
     0.0            32138          1      32138.0    32138      32138  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum              Name
 -------  ---------------  ---------  --------  -------  -------  -------------------------------
   100.0           506396          1  506396.0   506396   506396  convolution_2d(int*, int*, int)
```

Thread=4, Execution Time: 638810ns



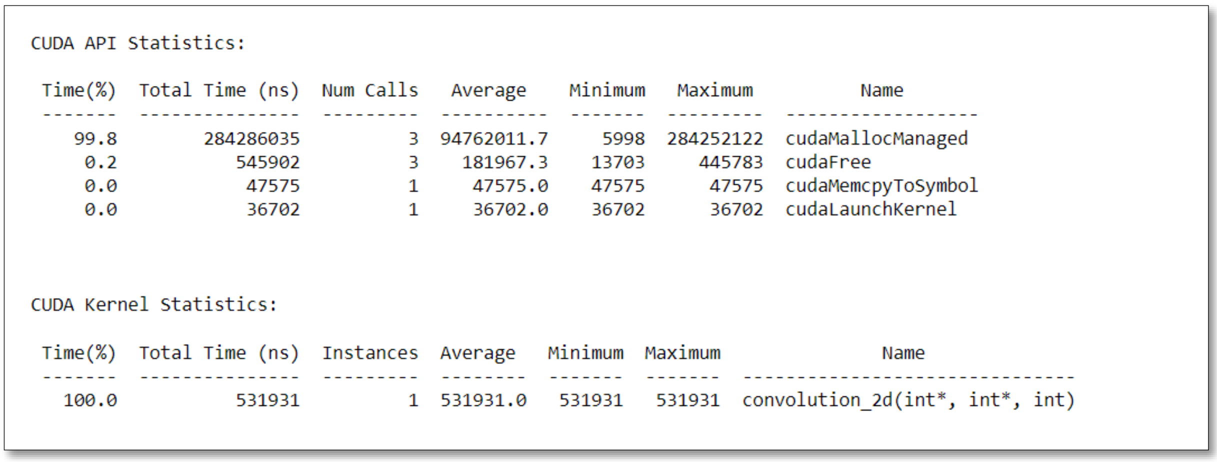
```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average     Minimum   Maximum          Name
 -------  ---------------  ---------  -----------  -------  ---------  ------------------
    99.9        322418643          3  107472881.0     7011  322380423  cudaMallocManaged
     0.0           147496          3      49165.3    20221      89048  cudaFree
     0.0            49048          1      49048.0    49048      49048  cudaMemcpyToSymbol
     0.0            39713          1      39713.0    39713      39713  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum              Name
 -------  ---------------  ---------  --------  -------  -------  -------------------------------
   100.0           638810          1  638810.0   638810   638810  convolution_2d(int*, int*, int)
```

Thread=8, Execution Time: 382717ns



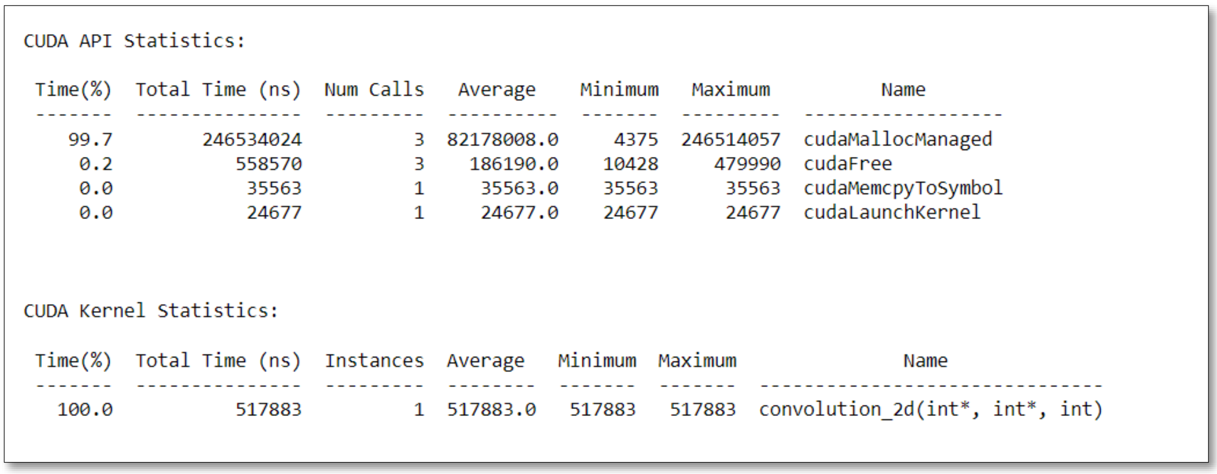
```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum          Name
 -------  ---------------  ---------  ----------  -------  ---------  ------------------
    99.8        242726932          3  80908977.3     4722  242701907  cudaMallocManaged
     0.2           371918          3    123972.7    10332     305877  cudaFree
     0.0            37507          1     37507.0    37507      37507  cudaMemcpyToSymbol
     0.0            29886          1     29886.0    29886      29886  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum               Name
 -------  ---------------  ---------  --------  -------  -------  ------------------------------
   100.0           382717          1  382717.0   382717   382717  convolution_2d(int*, int*, int)
```

Thread=16, Execution Time: 531931ns



```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum          Name
 -------  ---------------  ---------  ----------  -------  ---------  ------------------
    99.8        284286035          3  94762011.7     5998  284252122  cudaMallocManaged
     0.2           545902          3    181967.3    13703     445783  cudaFree
     0.0            47575          1     47575.0    47575      47575  cudaMemcpyToSymbol
     0.0            36702          1     36702.0    36702      36702  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum               Name
 -------  ---------------  ---------  --------  -------  -------  ------------------------------
   100.0           531931          1  531931.0   531931   531931  convolution_2d(int*, int*, int)
```

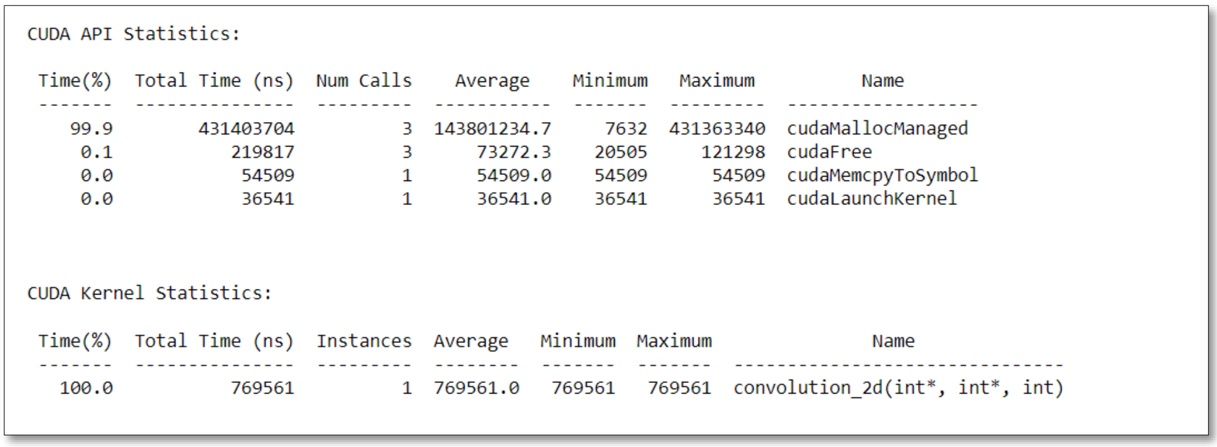
Thread=32, Execution Time: 517883ns



```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum          Name
 -------  ---------------  ---------  ----------  -------  ---------  ------------------
    99.7        246534024          3  82178008.0     4375  246514057  cudaMallocManaged
     0.2           558570          3    186190.0    10428     479990  cudaFree
     0.0            35563          1     35563.0    35563      35563  cudaMemcpyToSymbol
     0.0            24677          1     24677.0    24677      24677  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum               Name
 -------  ---------------  ---------  --------  -------  -------  ------------------------------
   100.0           517883          1  517883.0   517883   517883  convolution_2d(int*, int*, int)
```

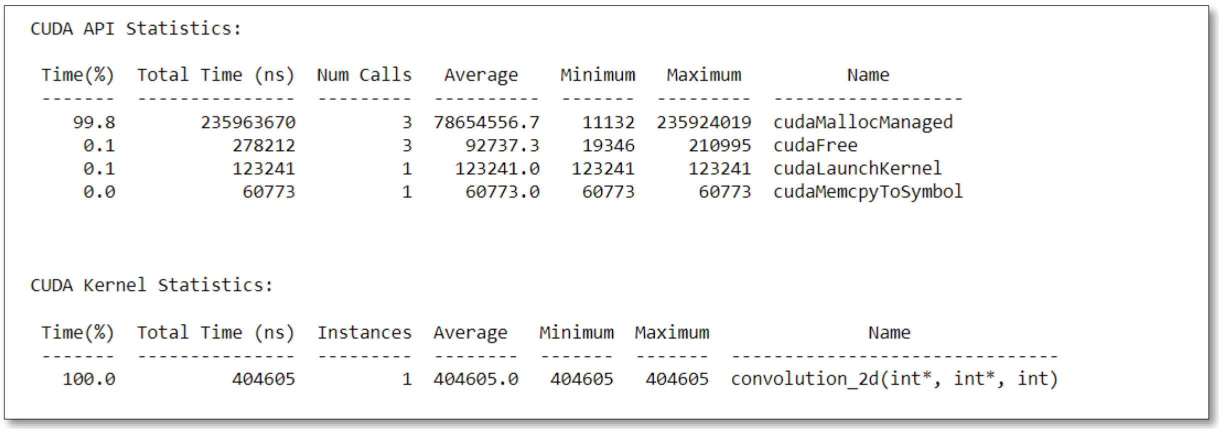
2^5

Thread=1, Execution Time: 769561ns



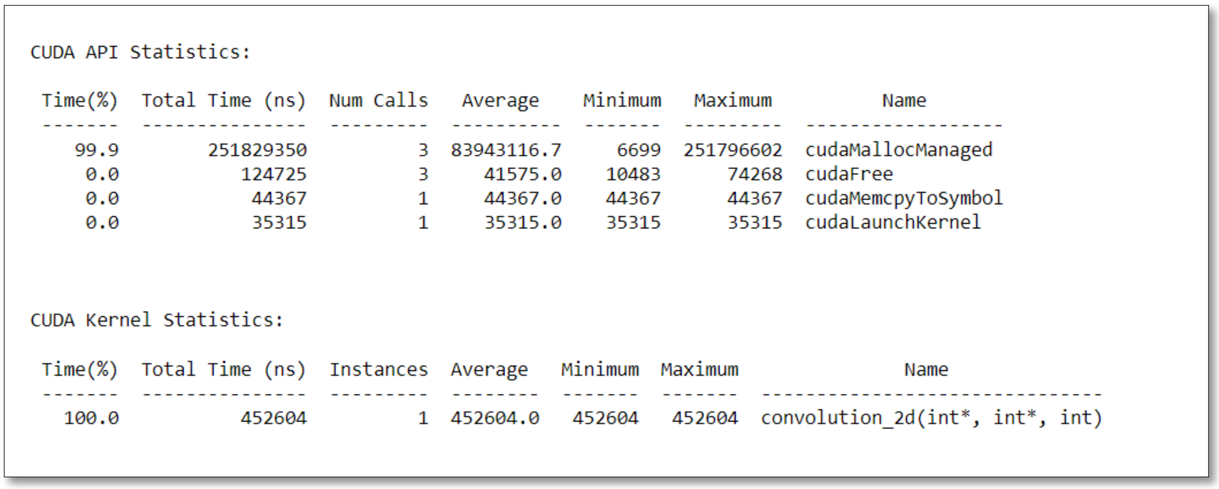
```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls    Average    Minimum   Maximum           Name
 -------  ---------------  ---------  -----------  -------  ---------  ------------------
    99.9        431403704          3  143801234.7     7632  431363340  cudaMallocManaged
     0.1           219817          3      73272.3    20505     121298  cudaFree
     0.0            54509          1      54509.0    54509      54509  cudaMemcpyToSymbol
     0.0            36541          1      36541.0    36541      36541  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                Name
 -------  ---------------  ---------  --------  -------  -------  -------------------------------
   100.0           769561          1  769561.0   769561   769561  convolution_2d(int*, int*, int)
```

Thread=4, Execution Time: 404605ns



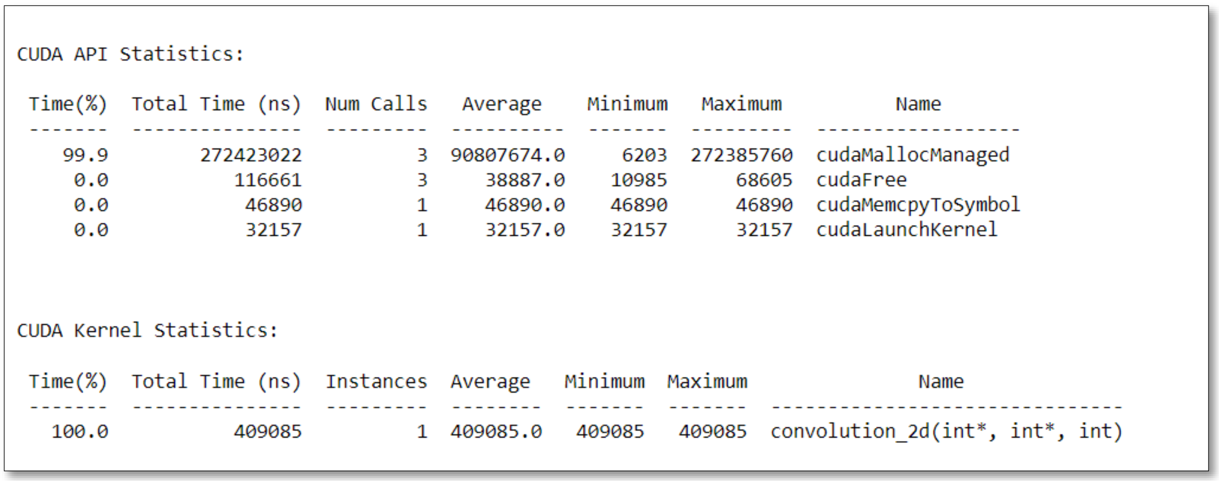
```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum          Name
 -------  ---------------  ---------  ----------  -------  ---------  ------------------
    99.8        235963670          3  78654556.7    11132  235924019  cudaMallocManaged
     0.1           278212          3     92737.3    19346     210995  cudaFree
     0.1           123241          1    123241.0   123241     123241  cudaLaunchKernel
     0.0            60773          1     60773.0    60773      60773  cudaMemcpyToSymbol


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                Name
 -------  ---------------  ---------  --------  -------  -------  -------------------------------
   100.0           404605          1  404605.0   404605   404605  convolution_2d(int*, int*, int)
```

Thread=8, Execution Time: 452604ns



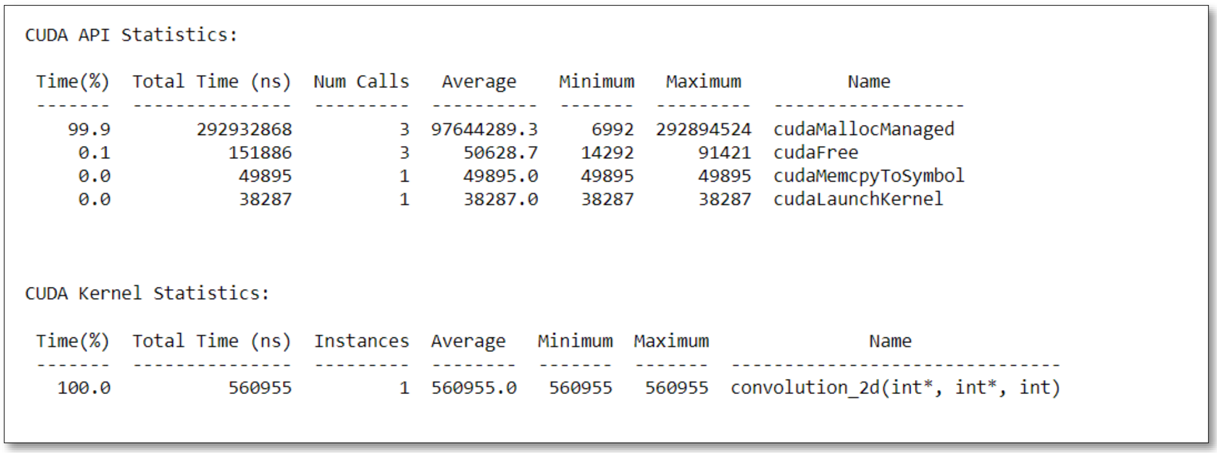
```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum           Name
 -------  ---------------  ---------  ----------  -------  ---------  ------------------
    99.9        251829350          3  83943116.7     6699  251796602  cudaMallocManaged
     0.0           124725          3     41575.0    10483      74268  cudaFree
     0.0            44367          1     44367.0    44367      44367  cudaMemcpyToSymbol
     0.0            35315          1     35315.0    35315      35315  cudaLaunchKernel


CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum                Name
 -------  ---------------  ---------  --------  -------  -------  -------------------------------
   100.0           452604          1  452604.0   452604   452604  convolution_2d(int*, int*, int)
```

Thread=16, Execution Time: 409085ns



```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum          Name
 -------  ---------------  ---------  ----------  -------  ---------  ------------------
    99.9        272423022          3  90807674.0     6203  272385760  cudaMallocManaged
     0.0           116661          3     38887.0    10985      68605  cudaFree
     0.0            46890          1     46890.0    46890      46890  cudaMemcpyToSymbol
     0.0            32157          1     32157.0    32157      32157  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum               Name
 -------  ---------------  ---------  --------  -------  -------  -------------------------------
   100.0           409085          1  409085.0   409085   409085  convolution_2d(int*, int*, int)
```

Thread=32, Execution Time: 560955ns



```
CUDA API Statistics:

 Time(%)  Total Time (ns)  Num Calls   Average    Minimum   Maximum          Name
 -------  ---------------  ---------  ----------  -------  ---------  ------------------
    99.9        292932868          3  97644289.3     6992  292894524  cudaMallocManaged
     0.1           151886          3     50628.7    14292      91421  cudaFree
     0.0            49895          1     49895.0    49895      49895  cudaMemcpyToSymbol
     0.0            38287          1     38287.0    38287      38287  cudaLaunchKernel



CUDA Kernel Statistics:

 Time(%)  Total Time (ns)  Instances  Average   Minimum  Maximum               Name
 -------  ---------------  ---------  --------  -------  -------  -------------------------------
   100.0           560955          1  560955.0   560955   560955  convolution_2d(int*, int*, int)
```