

Explanation:

Configuration used for the thread block is 32*32. The number of blocks was calculated using the following formula:

$(\text{Number of elements in columns/rows})/(\text{threadsPerBlock along a particular dimension})+1$

The following formula was used for calculating indexes:

$x/y/z \text{ index} = \text{blockIdx} * \text{blockDim} + \text{threadIdx}$

The convolution was implemented in the kernel in the following manner:

```
for(int p=0;p<=(j-1);p++)
{
    for(int q=0;q<=(k-1);q++)
    {
        if(!((m-p)<0 || (n-q)<0 || (m-p)>=z || (n-q)>=i))
        {
            c[(m*c_cols)+n]+=h[(p*k)+q]*a[((m-p)*i)+(n-q)];
            __syncthreads();
        }
    }
}
```

Observations:

It is observed that the time taken for computation using cudaMemcpy is less than cudaMallocManaged (unified memory).

Timing reported for the given 500*500 input test case is as follows:

For 2dconvolV1:

Time=16us

For 2dconvolV2:

Time = 630us

For 2dconvolV1:

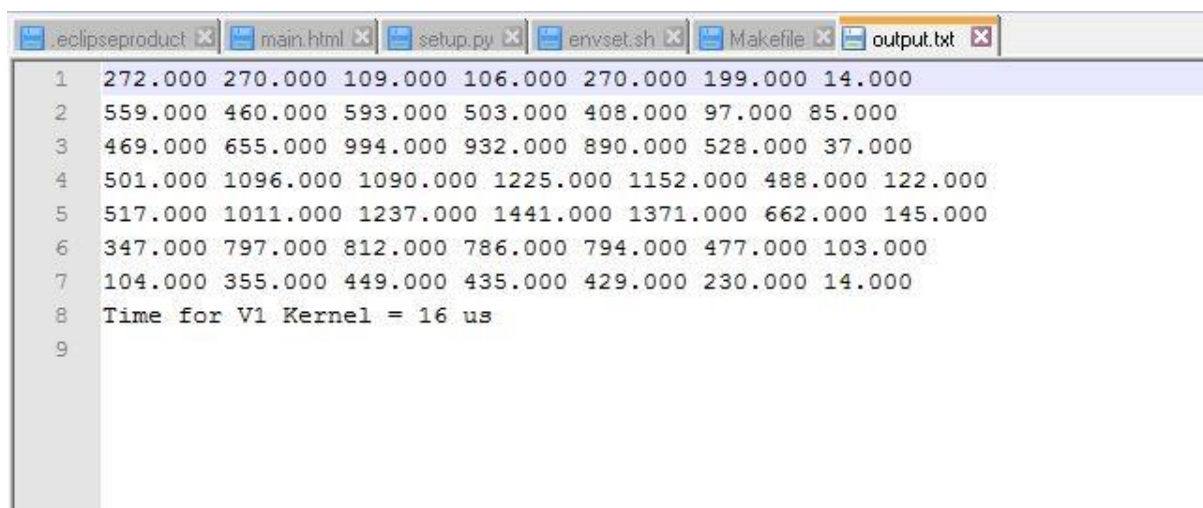
Test case 1:

Given input:



```
.eclipseproduct x main.html x setup.py x envset.sh x Makefile x output.txt x inputformat.txt x
1 17 2 4 3 14
2 19 6 20 9 1
3 3 11 20 19 17
4 19 20 10 16 19
5 13 20 17 20 14
6
7 16 14 1
8 15 4 6
9 8 15 1
```

Obtained output:



```
.eclipseproduct x main.html x setup.py x envset.sh x Makefile x output.txt x
1 272.000 270.000 109.000 106.000 270.000 199.000 14.000
2 559.000 460.000 593.000 503.000 408.000 97.000 85.000
3 469.000 655.000 994.000 932.000 890.000 528.000 37.000
4 501.000 1096.000 1090.000 1225.000 1152.000 488.000 122.000
5 517.000 1011.000 1237.000 1441.000 1371.000 662.000 145.000
6 347.000 797.000 812.000 786.000 794.000 477.000 103.000
7 104.000 355.000 449.000 435.000 429.000 230.000 14.000
8 Time for V1 Kernel = 16 us
9
```

The time obtained for 1st test case is 16us.

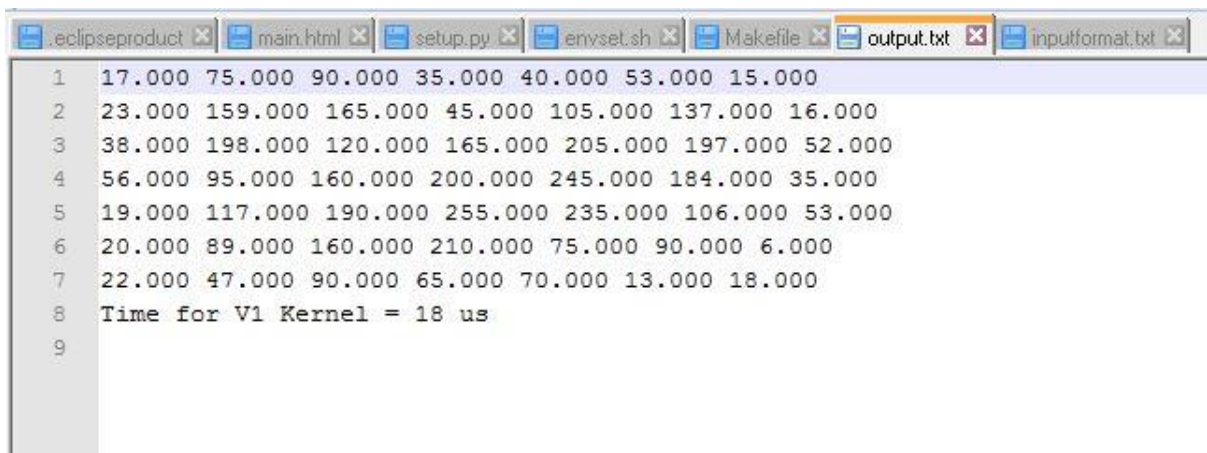
Test case 2:

Given input:



```
.eclipseproduct x main.html x setup.py x envset.sh x Makefile x output.txt x inputformat.txt x
1 17 24 1 8 15
2 23 5 7 14 16
3 4 6 13 20 22
4 10 12 19 21 3
5 11 18 25 2 9
6
7 1 3 1
8 0 5 0
9 2 1 2
```

Obtained output:



```
.eclipseproduct x main.html x setup.py x envset.sh x Makefile x output.txt x inputformat.txt x
1 17.000 75.000 90.000 35.000 40.000 53.000 15.000
2 23.000 159.000 165.000 45.000 105.000 137.000 16.000
3 38.000 198.000 120.000 165.000 205.000 197.000 52.000
4 56.000 95.000 160.000 200.000 245.000 184.000 35.000
5 19.000 117.000 190.000 255.000 235.000 106.000 53.000
6 20.000 89.000 160.000 210.000 75.000 90.000 6.000
7 22.000 47.000 90.000 65.000 70.000 13.000 18.000
8 Time for V1 Kernel = 18 us
9
```

The time obtained for 2nd test case is 18us.

For 2dconvolV2:

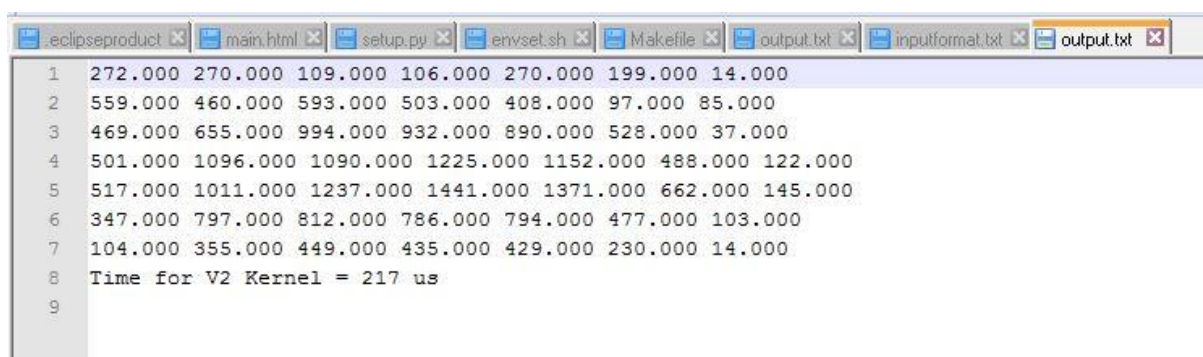
Test case 1:

Given input:



```
.eclipseproduct x main.html x setup.py x envset.sh x Makefile x output.txt x inputformat.txt x
1 17 2 4 3 14
2 19 6 20 9 1
3 3 11 20 19 17
4 19 20 10 16 19
5 13 20 17 20 14
6
7 16 14 1
8 15 4 6
9 8 15 1
```

Obtained output:



```
.eclipseproduct x main.html x setup.py x envset.sh x Makefile x output.txt x inputformat.txt x output.txt x
1 272.000 270.000 109.000 106.000 270.000 199.000 14.000
2 559.000 460.000 593.000 503.000 408.000 97.000 85.000
3 469.000 655.000 994.000 932.000 890.000 528.000 37.000
4 501.000 1096.000 1090.000 1225.000 1152.000 488.000 122.000
5 517.000 1011.000 1237.000 1441.000 1371.000 662.000 145.000
6 347.000 797.000 812.000 786.000 794.000 477.000 103.000
7 104.000 355.000 449.000 435.000 429.000 230.000 14.000
8 Time for V2 Kernel = 217 us
9
```

The time obtained for 1st test case is 217us.

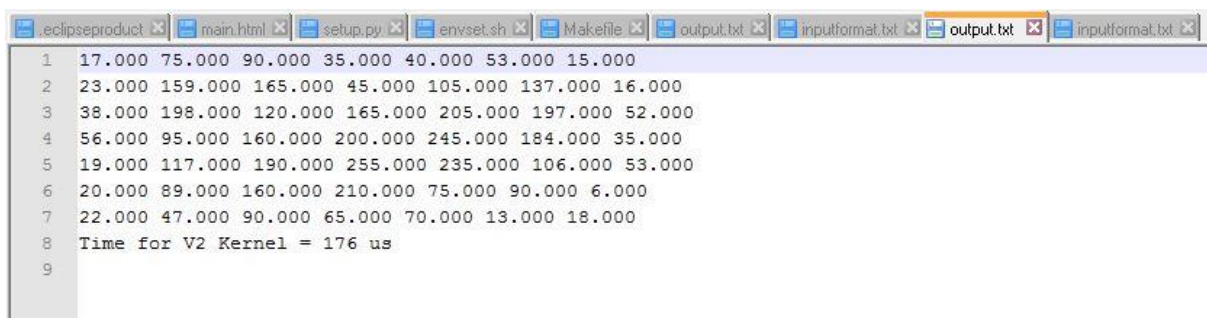
Test case 2:

Given input:



```
.eclipseproduct x main.html x setup.py x envset.sh x Makefile x output.txt x inputformat.txt x
1 17 24 1 8 15
2 23 5 7 14 16
3 4 6 13 20 22
4 10 12 19 21 3
5 11 18 25 2 9
6
7 1 3 1
8 0 5 0
9 2 1 2
```

Obtained output;



```
.eclipseproduct x main.html x setup.py x envset.sh x Makefile x output.txt x inputformat.txt x output.txt x inputformat.txt x
1 17.000 75.000 90.000 35.000 40.000 53.000 15.000
2 23.000 159.000 165.000 45.000 105.000 137.000 16.000
3 38.000 198.000 120.000 165.000 205.000 197.000 52.000
4 56.000 95.000 160.000 200.000 245.000 184.000 35.000
5 19.000 117.000 190.000 255.000 235.000 106.000 53.000
6 20.000 89.000 160.000 210.000 75.000 90.000 6.000
7 22.000 47.000 90.000 65.000 70.000 13.000 18.000
8 Time for V2 Kernel = 176 us
9
```

The time obtained for 2nd test case is 176us.