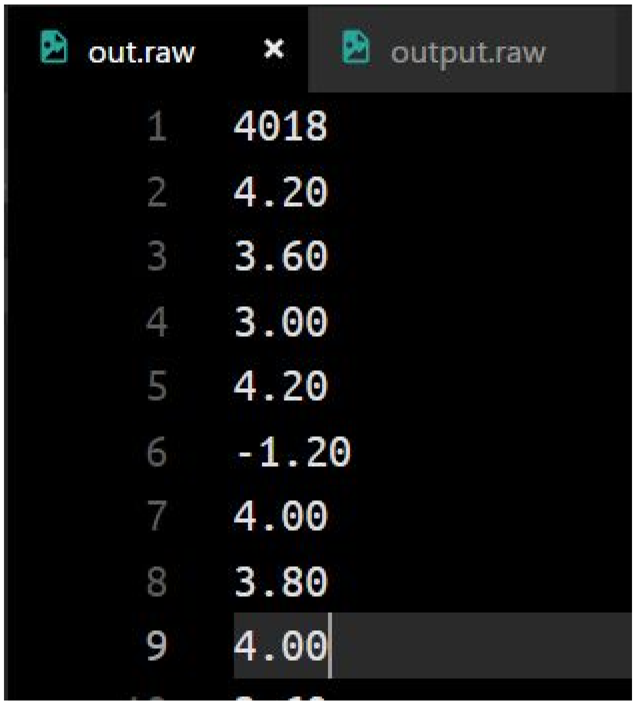
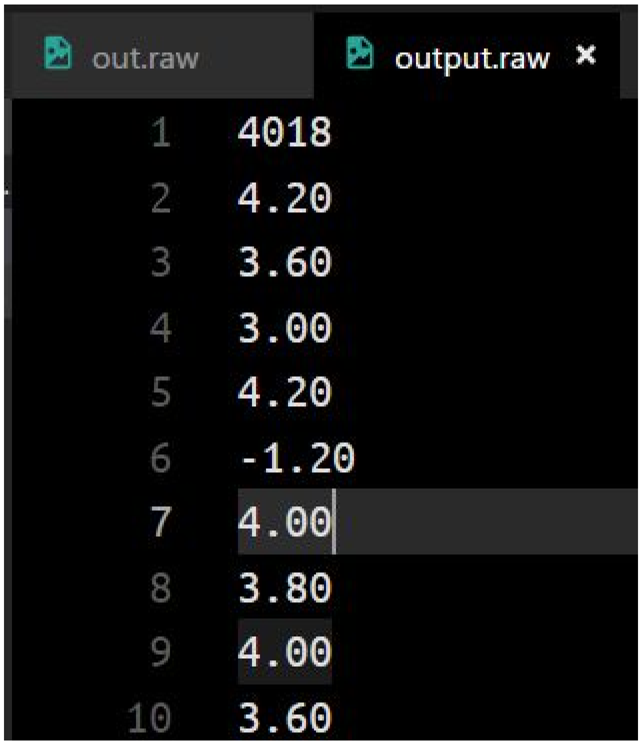
A2 Report

Dibyadarshan Hota 16CO154
Omkar Prabhu 16CO233

Q1)



```
4018
4.20
3.60
3.00
4.20
-1.20
4.00
3.80
4.00
```



```
4018
4.20
3.60
3.00
4.20
-1.20
4.00
3.80
4.00
3.60
```

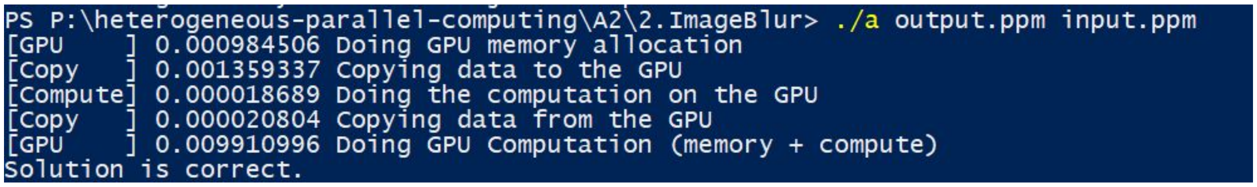
1. How many floating operations are being performed in your vector add kernel?
A1) The number of floating operations in kernel is equal to size of the host arrays generated (4018).

2. How many global memory reads are being performed by your kernel?
A2) 2 * 4018 = 8036 global memory reads are being performed.

3. How many global memory writes are being performed by your kernel?
A3) 4018 global memory writes are being performed.

4. In what ways did Thrust make developing a functional vector addition code easier or harder?
A4) Using Thrust we could use direct function call for vector addition and we didn't have to write kernels explicitly, this made our work easier. Plus, when cudamalloc() is used we have to free the memory explicitly but allocating memory dynamically using vectors doesn't require free() function.

Q2)



```
PS P:\heterogeneous-parallel-computing\A2\2.ImageBlur> ./a output.ppm input.ppm
[GPU    ] 0.000984506 Doing GPU memory allocation
[Copy   ] 0.001359337 Copying data to the GPU
[Compute] 0.000018689 Doing the computation on the GPU
[Copy   ] 0.000020804 Copying data from the GPU
[GPU    ] 0.009910996 Doing GPU Computation (memory + compute)
Solution is correct.
```

1. How many floating operations are being performed in your color conversion kernel?
A1) The number of floating operations is equal to number valid surrounding pixels for each pixel in the image + the number of pixels in the image (for finding the average value).

=> Non-border pixels*9 + corner pixels*4 + border pixels along height + border pixels along width + number of pixels (for averaging)
=> 9*((height-2)*(width-2))+4*(4)+2*(height-2)*6+2*(width-2)*6+(height*width)

2. How many global memory reads are being performed by your kernel?
A2) For each pixel all of the surrounding valid pixels are being read.

=> Non-border pixels*9 + corner pixels*4 + border pixels along height + border pixels along width
=> 9*((height-2)*(width-2))+4*(4)+2*(height-2)*6+2*(width-2)*6

3. How many global memory writes are being performed by your kernel?
A3) Global memory write is equal to number of pixels in the output image.

4. Describe what possible optimizations can be implemented to your kernel to achieve a performance speedup.
A4) One way of speeding up the kernel would be to use the shared memory per block for a segment of the image. Reading and writing into global memory is relatively slower and reducing global reads in this way will make the program run faster.