Part1-a:

Displaying image 1:

Computing the difference between values in neighboring pixels(`computing difference in x-direction and y-direction`):

First, print the values of pixels from 0,0 to 15,7 to see whether the computation is right:

```
[ 91 108 118] [ 91 108 118] [ 91 108 118] [ 91 108 118] [ 91 108 118] [ 91 108 118] [ 91 108 118]
[ 91 108 118] [ 91 108 118] [ 91 108 118] [ 91 108 118] [ 91 108 118] [ 91 108 118] [ 91 108 118]
[ 92 109 119] [ 92 109 119] [ 92 109 119] [ 92 109 119] [ 92 109 119] [ 92 109 119] [ 92 109 119]
[ 92 109 119] [ 92 109 119] [ 92 109 119] [ 92 109 119] [ 92 109 119] [ 92 109 119] [ 92 109 119]
[ 93 110 120] [ 93 110 120] [ 93 110 120] [ 93 110 120] [ 93 110 120] [ 93 110 120] [ 93 110 120]
[ 93 110 120] [ 93 110 120] [ 93 110 120] [ 93 110 120] [ 93 110 120] [ 93 110 120] [ 93 110 120]
[ 94 111 121] [ 94 111 121] [ 94 111 121] [ 94 111 121] [ 94 111 121] [ 94 111 121] [ 94 111 121]
[ 94 111 121] [ 94 111 121] [ 94 111 121] [ 94 111 121] [ 94 111 121] [ 94 111 121] [ 94 111 121]
[ 95 112 122] [ 95 112 122] [ 95 112 122] [ 96 113 123] [ 96 113 123] [ 97 114 124] [ 97 114 124]
[ 95 112 122] [ 95 112 122] [ 95 112 122] [ 96 113 123] [ 96 113 123] [ 97 114 124] [ 97 114 124]
[ 95 112 122] [ 95 112 122] [ 95 112 122] [ 96 113 123] [ 96 113 123] [ 97 114 124] [ 97 114 124]
[ 95 112 122] [ 95 112 122] [ 95 112 122] [ 96 113 123] [ 96 113 123] [ 97 114 124] [ 97 114 124]
[ 95 112 122] [ 95 112 122] [ 95 112 122] [ 96 113 123] [ 96 113 123] [ 97 114 124] [ 97 114 124]
[ 95 112 122] [ 95 112 122] [ 95 112 122] [ 96 113 123] [ 96 113 123] [ 97 114 124] [ 97 114 124]
[ 95 112 122] [ 95 112 122] [ 95 112 122] [ 96 113 123] [ 96 113 123] [ 97 114 124] [ 97 114 124]
```

Now the **difference in x-direction and y-direction:**

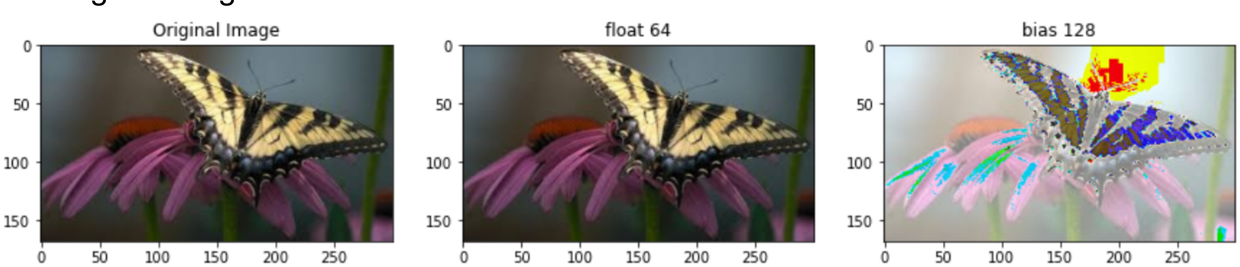
```
Diff in x-direction:
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [1 1 1] [0 0 0] [1 1 1] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [1 1 1] [0 0 0] [1 1 1] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [1 1 1] [0 0 0] [1 1 1] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [1 1 1] [0 0 0] [1 1 1] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [1 1 1] [0 0 0] [1 1 1] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [1 1 1] [0 0 0] [1 1 1] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [1 1 1] [0 0 0] [1 1 1] [0 0 0] [0 0 0]
```

```
Diff in y-direction:
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[1 1 1] [1 1 1] [1 1 1] [1 1 1] [1 1 1] [1 1 1] [1 1 1]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[1 1 1] [1 1 1] [1 1 1] [1 1 1] [1 1 1] [1 1 1] [1 1 1]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[1 1 1] [1 1 1] [1 1 1] [1 1 1] [1 1 1] [1 1 1] [1 1 1]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[1 1 1] [1 1 1] [1 1 1] [2 2 2] [2 2 2] [3 3 3] [3 3 3]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
[0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0] [0 0 0]
```

Number of bytes:`151200 Bytes`
Size: `height: 168, width; 300, depth: 3`

Part1-b:

The original image is unit8.



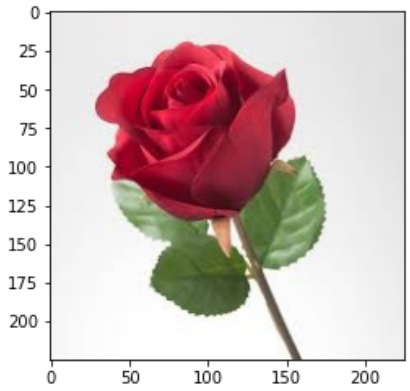
Part1-c:

I used scaling for this part. I mean I reduced the number of bytes by multiplying its row and col in 0.5, 0.4, 0.3, 0.2, and 0.1. The result is:

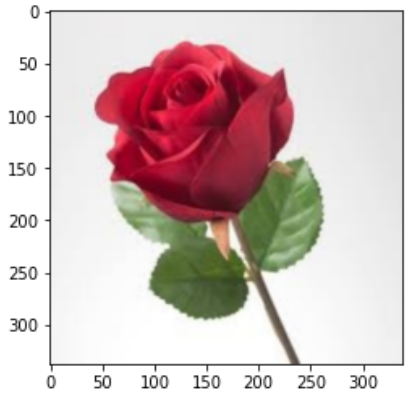
It seems that the 67*120-byte image is good enough as the details are visible. But the flower and the butterfly are still recognizable in the 50*90-byte image.

Part2:

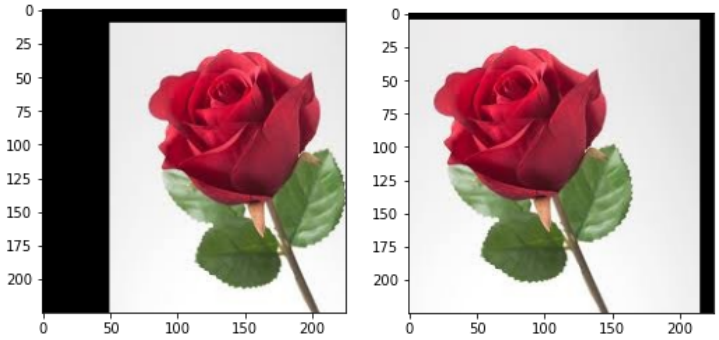
Displaying image 2:



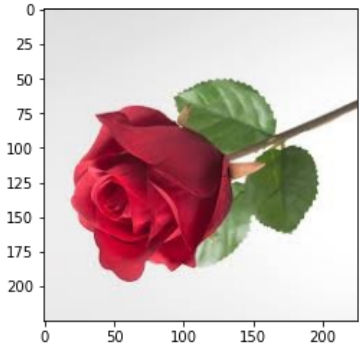
Scaling(1.5*1.5):



Translation:(First:x+=50, y+=10 Second:x+=-10, y+=5)

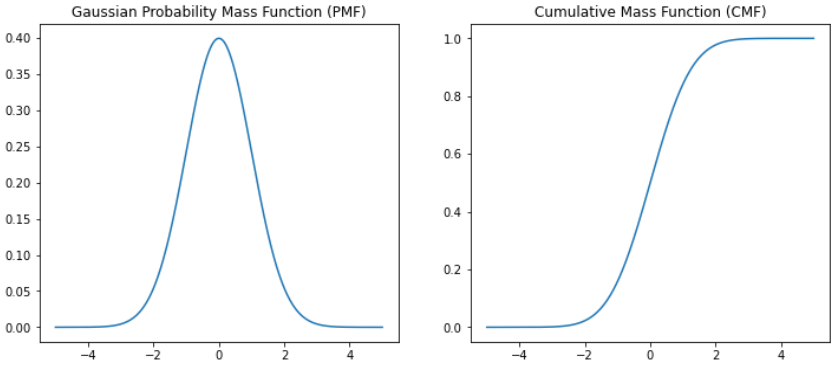


rotation(90°)



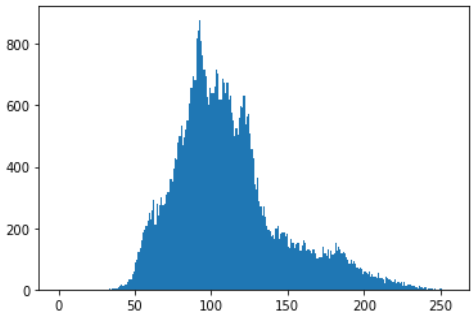
Part2:

Displaying image 2:



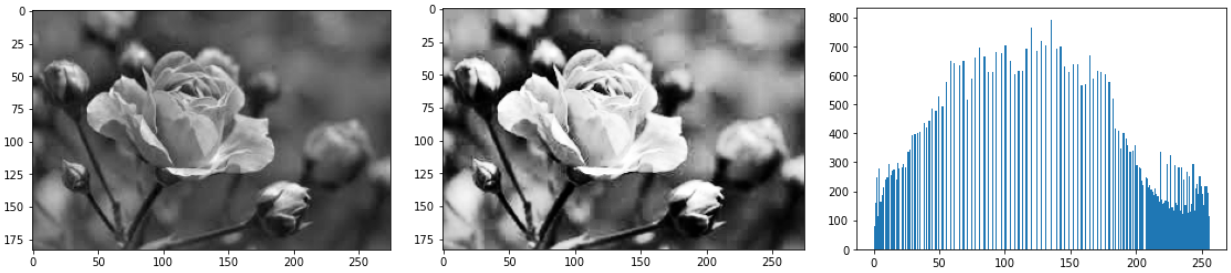
mu = 0, sigma=1

Original image histogram:



For this part first we need to ture the imagr to grayscale

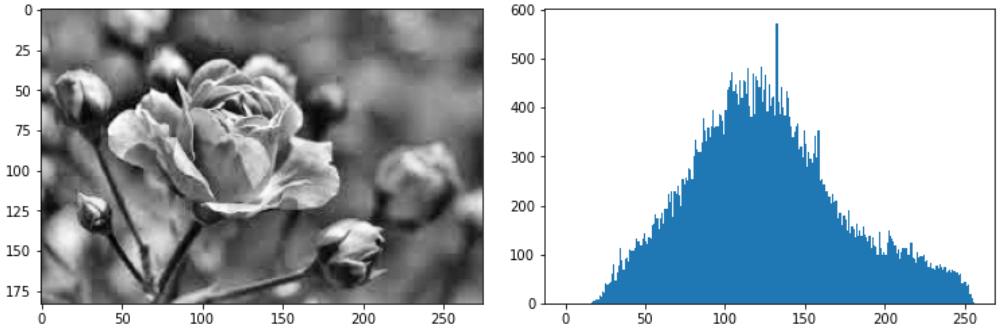
Original Image gray equalized Image Equalized Histogram



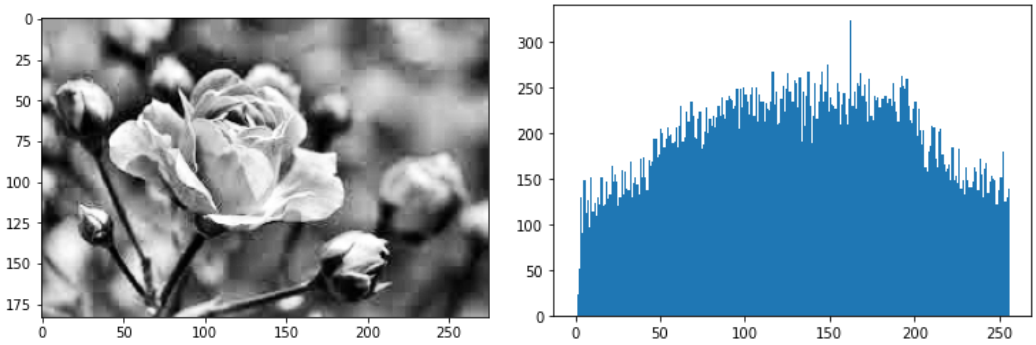
Histogram matching:

Using the *original gray* image and *equalized image* as the reference for matching

Matching original image:



Matching equalized image:



Contrast is more in equalized than the original image(grayscale) and that what image equalization does. And we can see the gussain(normal) distribution for the equalized image.
No difference is seen by eyes between matching image with using the original image and equalized image as the reference. Olthow their histograms both look like normal distribution but, they are different.