Objective

The goal of this assignment is threefold:

a) For the image given below (lena.jpg) add random Gaussian noise (mean = 0.0, sigma = 20.0) and generate array of 10 noisy images.

b) Compute average of the 10 images by adding them pixel by pixel and dividing the result by 10. This should be able to clean up the image by suppressing the random noise.

c) Display at least 6 noisy images and finally the clean image

Code

Objective:

The goal of this assignment is three fold:

a) For the image given below (lena.jpg) add random Gaussian noise (mean = 0.0, sigma = 20.0) and generate array of 10 noisy images.

b) Compute average of the 10 images by adding them pixel by pixel and dividing the result by 10. This should be able to clean up the image by suppressing the random noise.

c) Display at least 6 noisy images and finally the clean image

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
def addnoise(x,mu,sig):
    m = len(x)
    n = len(x[0])
    ns = np.random.normal(mu,sig,(m,n))
    ns = np.int16(ns)
    x = np.int16(x)
    x = x+ns
    x[x < 0] = 0
    x[x > 255] = 255
    x = np.uint8(x)
    return x
```

a) For the image given below (lena.jpg) add random Gaussian noise (mean = 0.0, sigma = 20.0) and generate array of 10 noisy images.

b) Compute average of the 10 images by adding them pixel by pixel and dividing the result by 10. This should be able to clean up the image by suppressing the random noise and

c) Display at least 6 noisy images and finally the clean image

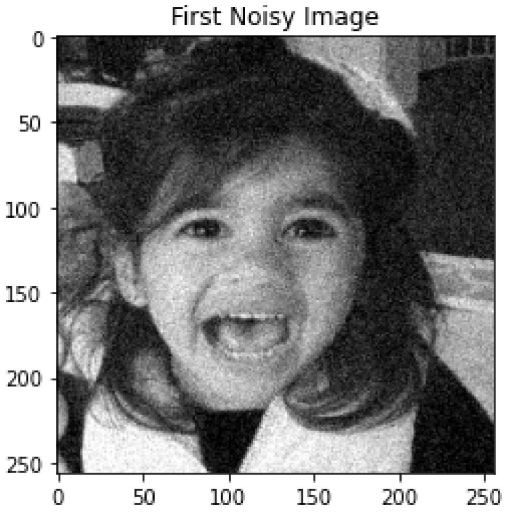
In [3]:
```
def main():
    x = plt.imread('arya.jpg')
    x1 = x[:,:,0]
    m = len(x1)
    n = len(x1[0])
    x2 = np.zeros([m,n], dtype = 'float')

    y = np.ndarray([10,m,n], dtype = 'uint8')

    for i in range (0,10):
        y[i,:,:] = addnoise(x1,0.0,20.0)
    for i in range (0,10):
        x2 += y[i,:,:] / 10
```

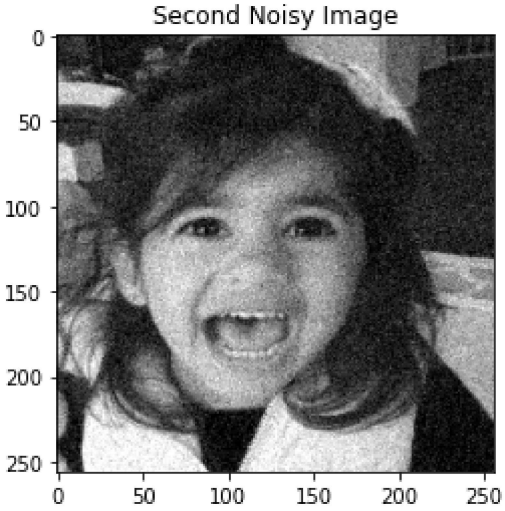
```
    x2 = x2.astype('uint8')

    #Produce six noisy images and one clean image
    plt.imshow(y[0], cmap = 'gray' )
    plt.title('First Noisy Image')
    plt.show()
    plt.imsave('First Noisy Image.jpg', y[0], cmap = 'gray')

    plt.imshow(y[1], cmap = 'gray' )
    plt.title('Second Noisy Image')
    plt.show()
    plt.imsave('Second Noisy Image.jpg', y[1], cmap = 'gray')


    plt.imshow(y[2], cmap = 'gray' )
    plt.title('Third Noisy Image')
    plt.show()
    plt.imsave('Third Noisy Image.jpg', y[2], cmap = 'gray')


    plt.imshow(y[3], cmap = 'gray' )
    plt.title('Fourth Noisy Image')
    plt.show()
    plt.imsave('Forth Noisy Image.jpg', y[3], cmap = 'gray')


    plt.imshow(y[4], cmap = 'gray' )
    plt.title('Fifth Noisy Image')
    plt.show()
    plt.imsave('Fifth Noisy Image.jpg', y[4], cmap = 'gray')


    plt.imshow(y[5], cmap = 'gray')
    plt.title('Sixth Noisy Image')
    plt.show()
    plt.imsave('Sixth Noisy Image.jpg', y[5], cmap = 'gray')


    plt.imshow(x2, cmap = 'gray')
    plt.title('Filtered Image')
    plt.imsave('Filtered Image.jpg', x2, cmap = 'gray')
```

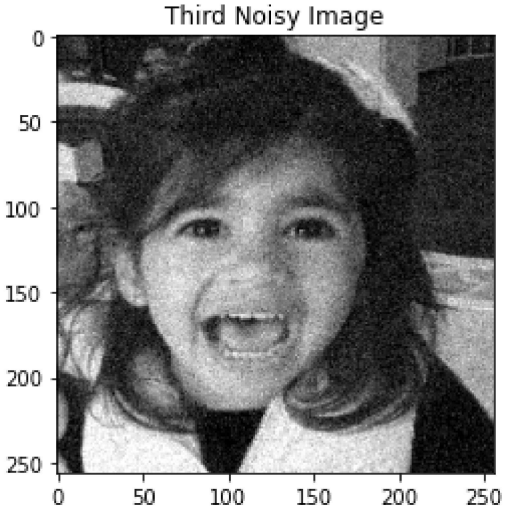
Results

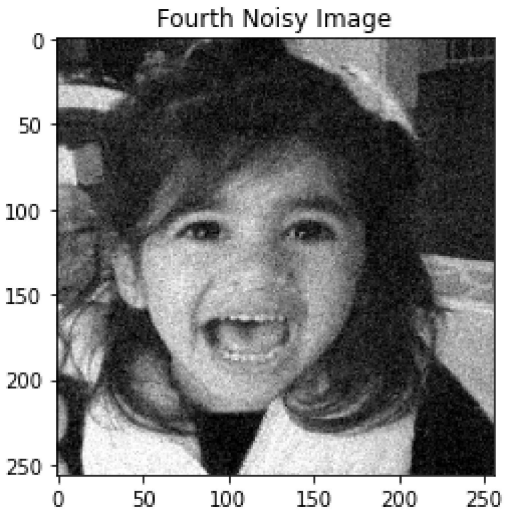
In [4]:

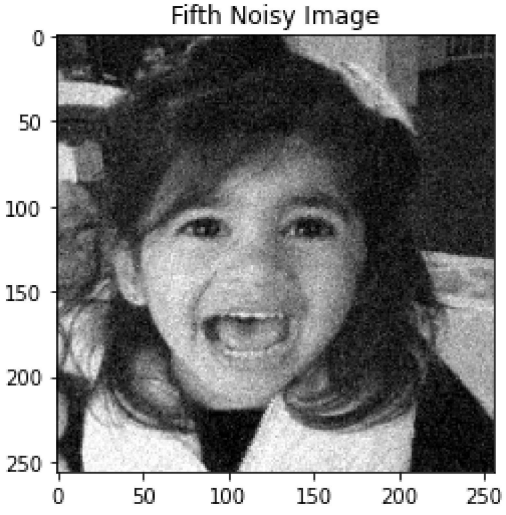
```
if __name__=='__main__':
    main()
```

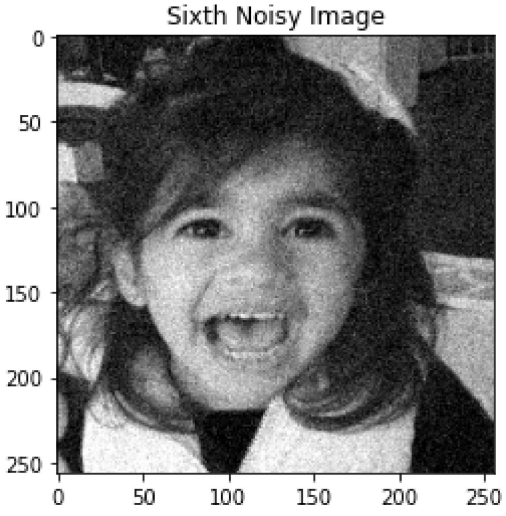

First Noisy Image
0
50
100
150
200
250
0
50
100
150
200
250

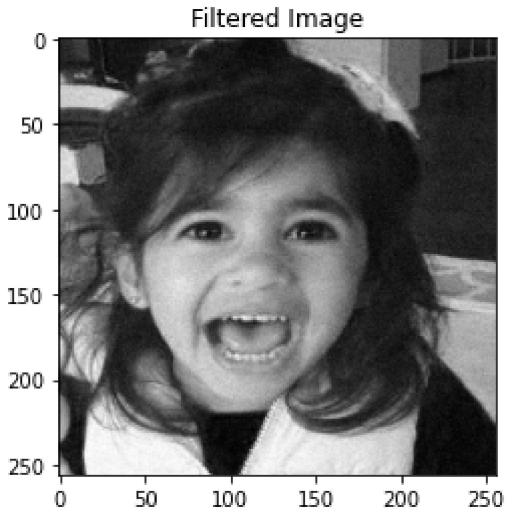

Second Noisy Image
0
50
100
150
200
250
0
50
100
150
200
250


Third Noisy Image
0
50
100
150
200
250
0
50
100
150
200
250


Fourth Noisy Image
0
50
100
150
200
250
0
50
100
150
200
250


Fifth Noisy Image
0
50
100
150
200
250
0
50
100
150
200
250


Sixth Noisy Image
0
50
100
150
200
250
0
50
100
150
200
250



In [ ]:

Conclusion

Based on the images above we can see that after implementing Gaussian noise with a mean value of 0 and a sigma value of 20 we can see that the images are noisy, however, after computing the average of 10 images by adding them pixel by pixel and dividing the result by 10 gives us a cleaned image which is done by suppressing the random noise.