Von Kaukeano
915596703
Practicum 1

1.

a.

```
%% (a)

load('pic.mat');

[height,width]=size(xx);

whos

figure(1)
imshow(xx, [0 255])
```

```
  Name           Size                Bytes  Class     Attributes

  height         1x1                     8  double
  width          1x1                     8  double
  xx           311x639              198729  uint8
```

Part a. begins by uploading the image into Matlab and then showing the image with the imshow command. The height and width is found by setting the variables to the size of the image and then presenting them to the output window.

b.

```
%% (b)

zz= [];
for i=1:height
    zz(i,:)=conv([1 -0.9], xx(i,:));
end
zz=zz(:,1:width);

figure(2)

imshow(zz, [-255 255])
```

The image is then differentially encoded to reduce numerical range of the image amplitude after image compression. This impulse response would be considered an FIR high pass filter that sharpens the image. The equation would be considered y[n] = 1-9z^{-1} CONV x[n]. As you can see, zz is the output which equals to the impulse response convoluted with the original image xx. The impulse response is a kernel which multiplies the values of xx by -.9 and subtracts by 1 for every value of n in the array and is causal.

c.

Inverting the effect of the filter begins with our equation y[n] = 1-9z^{-1} CONV x[n]. Instead we want the opposite x[n] = x[n]/ 1-9z^{-1}. After doing the long division it can be seen that the inverse z transform is the equation given in the practicum yy[i,n] =$\sum_{k=0}^{N-1}. 9^k * zz[i, n-k]$. This impulse response creates an IIR low pass filter and is reverse of the original impulse signal and noncausal. It will continuously apply for the entire array for every N samples. This will work to restore the image but is computationally expensive.

d.

```
%% (d)
N = 25;
yy= [];
k =[0:N];
for i = 1:height
yy(i,:)=conv(.9.^k, zz(i,:));
end
yy= yy(:,1:width);

figure(3);
imshow(yy, [0 255])
```

After applying the filter onto the image to restore it at N = 25 the image is not completely restored. On the left side of the image there seems to be a bit of distortion left and does not match the original image.

e.

After trying several different values of N, I could see that the lower the value the more distortion that would be left on the output image. When increasing N the time that it would take to run the program would take significantly larger yet would remove more and more distortion.

f.

```
%% (f)
for i=1:height
    for n = 1:width
        bb(i,n) = .9*bb(i,n-1) + zz(i,n);
    end
end

bb=bb(:,1:width);

figure(4)

imshow(bb, [0 255])
```

The decoder above is an attempt to mathematically decode the image that was filtered. By using the Linear Constant-Coefficient Difference Equation.

$$\underset{\text{present output}}{y[n]} = -\underbrace{\sum_{k=1}^{N} a_k y[n-k]}_{\text{past outputs}} + \underbrace{\sum_{m=0}^{M} b_m x[n-m]}_{\text{Present \& past inputs}}$$

I wanted to reverse the 1-.9[n-1] encoder by multiplying the past outputs by positive .9 and adding the present and past inputs. It seems to have an array index issue. If successful this would be computationally less expensive and more accurate because the IIR filter does not have a linear phase which causes some extra distortion. IIR filters have an infinite number of ripples in their impulses. By reversing the math we are much more precise on restoring the image completely and uses a lot less computation.

https://dsp.stackexchange.com/questions/35597/what-is-the-significance-of-butterworth-filter-in-image-processing

http://www.songho.ca/dsp/convolution/convolution.html#impulse_response

```
%% ========== Matlab code sample for Practicum 1 ===========
% Simulation: Image Deconvolution
% Von Kaukeano
% 915596703
clear
close all

%% (a)

load('pic.mat');

[height,width]=size(xx);

whos

figure(1)
imshow(xx, [0 255])

%% (b)

zz= [];
for i=1:height
    zz(i,:)=conv([1 -0.9], xx(i,:));
end
zz=zz(:,1:width);

figure(2)

imshow(zz, [-255 255])
%% (d)
N = 25;
yy= [];
k =[0:N];
for i = 1:height
    yy(i,:)=conv(.9.^k, zz(i,:));
end
yy= yy(:,1:width);

figure(3);
imshow(yy, [0 255])

%% (f)
bb=[];
for i=1:height
    for n = 1:width
        bb(i,n) = .9*bb(i,n-1) + zz(i,n);
    end
end

bb=bb(:,1:width);

figure(4)

imshow(bb, [0 255])
```