LAB 5

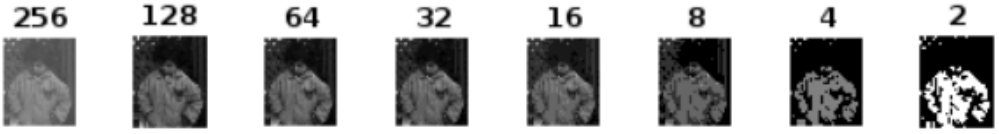
AIM: Implement the following algorithms

1. Take 'pout.tif' image and perform Gray-level Slicing on it. Diplay images with 2, 16, 64, 128 and 256 graylevels.

lab51.m

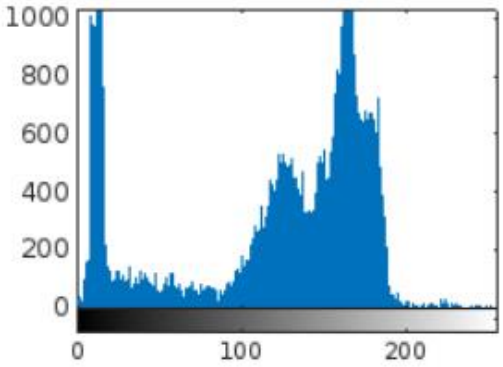
```
1   img=imread('pout.tif');
2   subplot(1,8,1);
3   imshow(img);
4   title('256');
5   [m,n]=size(img);
6   k=1;
7   img1=img;
8   while k<=7
9    img1=double(img);
10   img1=floor(img1/(2^k));
11   k=k+1;
12   img1=mat2gray(img1);
13   subplot(1,8,k);
14   imshow(img1);
15   title([num2str(256/(2^(k-1)))]);
16  end
```

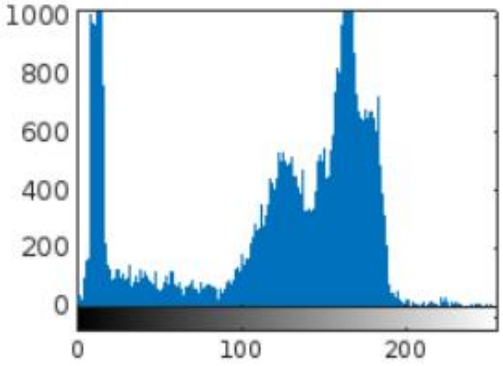
Figure 1



2. Consider an image of 128x128 (Hint: You can resize 'cameraman.tif' to 0.5) and Implement Nearest-Neighbour Interpolation Algorithm and covert into 256x256. Don't use in-buit functions like linspace, meshgrid and interp2. Compare your result with the result obtained using the function interp2.

```
%2. Consider an image of 128x128 (Hint: You can resize 'cameraman.tif' to 0.5)
img=imread('cameraman.tif');
resize_img=imresize(img, 0.5);
%Implement Nearest-Neighbour Interpolation Algorithm and covert into
%256x256. Don't use in-buit functions like linspace, meshgrid and interp2.
[m,n]=size(resize_img);
nearest_neighbour_img=zeros(256,256);
j=1;
for i=1:1:n
    a1(:,j)=resize_img(:,i);
    a1(:,j+1)=resize_img(:,i);
    j=j+2;
end
j=1;
for i=1:1:m
    nearest_neighbour_img(j,:)=a1(i,:);
    nearest_neighbour_img(j+1,:)=a1(i,:);
    j=j+2;
end
subplot(2,2,1);
imshow(nearest_neighbour_img,[]);
subplot(2,2,2);
imhist(uint8(nearest_neighbour_img));
%Compare your result with the result obtained using the function interp2.
a=double(resize_img);
x=linspace(1,128,128);
y=linspace(1,128,128);
xi=linspace(1,128,256);
yi=linspace(1,128,256);
[xx, yy]=meshgrid(xi,yi);
zz=interp2(x,y,a,xx,yy,'nearest');
subplot(2,2,3);
imshow(mat2gray(zz));
subplot(2,2,4);
imhist(uint8(zz));
```


1000
800
600
400
200
0
0
100
200


1000
800
600
400
200
0
0
100
200

3. Take 'Cameraman.tif' image and implement Shear Transformation.

a. Apply shear transformation in X-direction with value 2

b. Apply shear transformation in Y-Direction with value 3

c. Compare your result with the output generated by in-built function imtransform.

```
img=imread('cameraman.tif');
%a. Apply shear transformation in X-direction with value 2
[m,n]=size(img);

for i=1:m
    for j=1:n
        x=i+(2*j);
        y=j;
        shear_x(x,y)=img(i,j);
    end
end
subplot(2,2,1);
imshow(shear_x);
title("shearX");

%b. Apply shear transformation in Y-Direction with value 3
for i=1:m
    for j=1:n
        x=i;
        y=j+(3*i);
        shear_y(x,y)=img(i,j);
    end
end
subplot(2,2,2);
imshow(shear_y);
title("shearY");
%c. Compare your result with the output generated by in-built function
 imtransform.
tform = maketform('affine',[1 2 0; 0 1 0; 0 0 1]);
ix = imtransform(img,tform);
subplot(2,2,3);
imshow(ix);
title("shearXusefunction");
tform = maketform('affine',[1 0 0; 3 1 0; 0 0 1]);
iy = imtransform(img,tform);
subplot(2,2,4);
imshow(iy);
title("shearYusefunction");
```

shearX

shearY

shearXusefunction

shearYusefunction