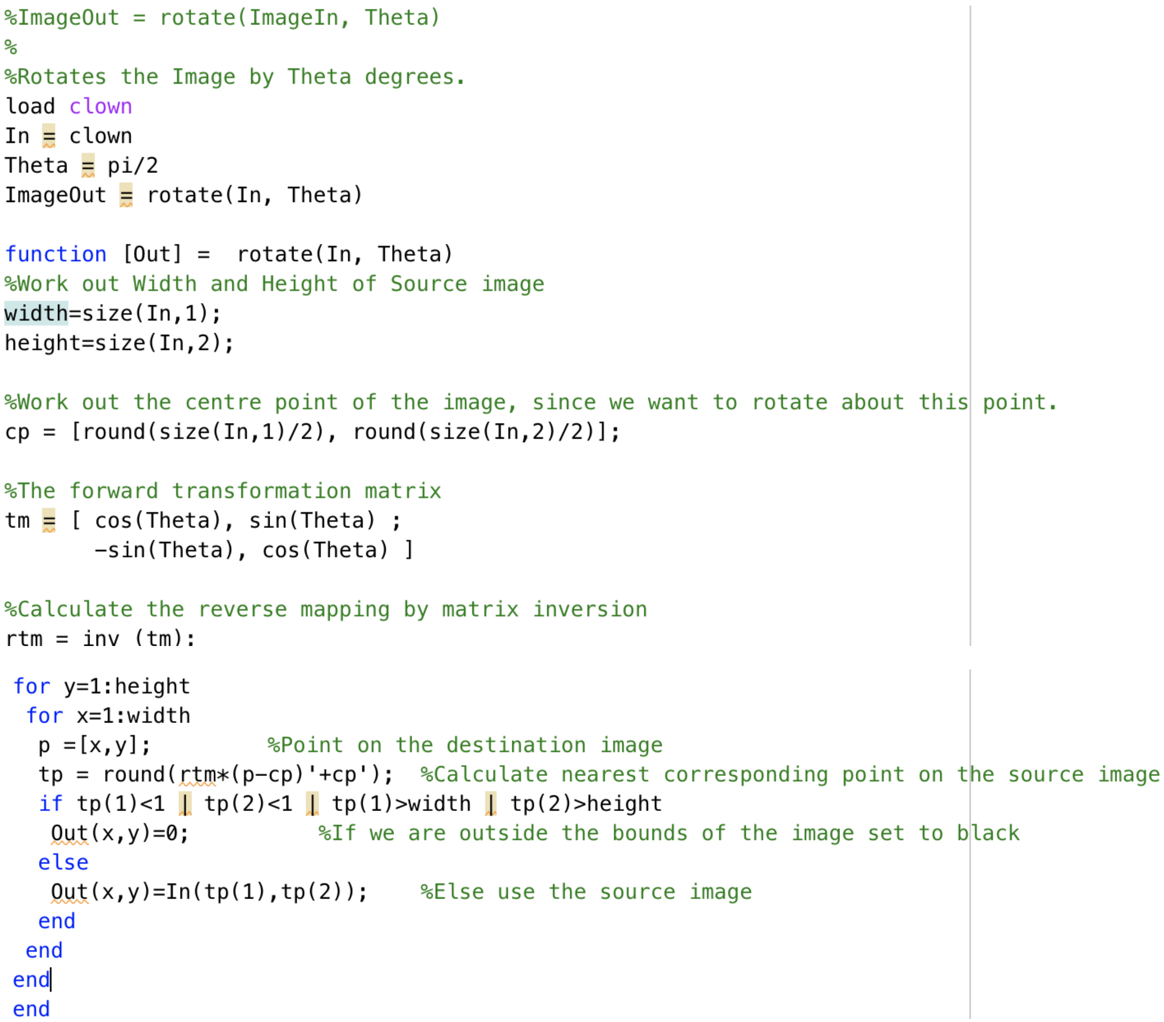
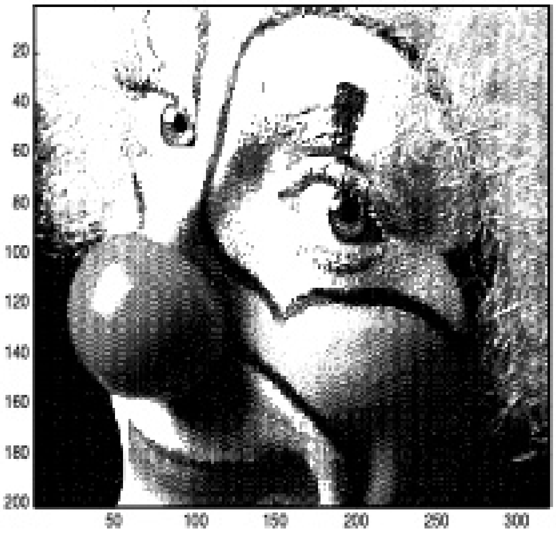
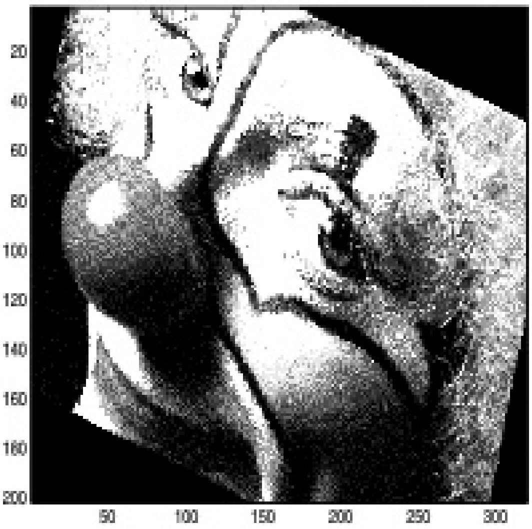
Task 1 - Rotation

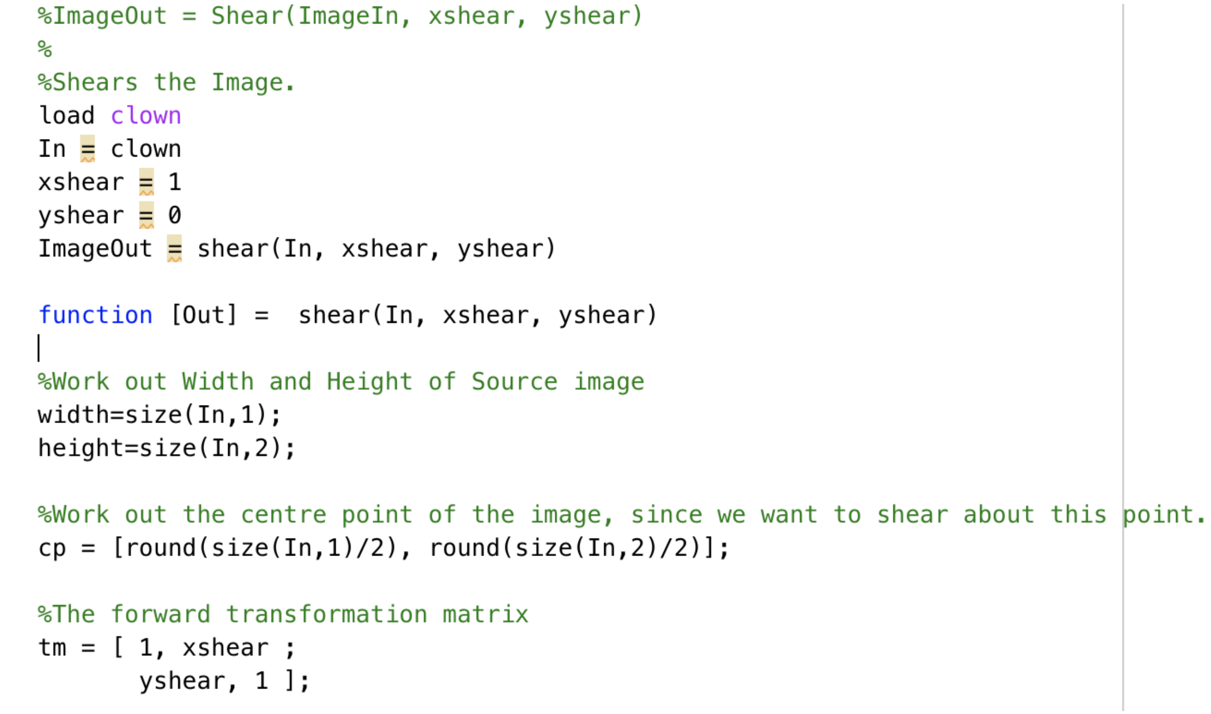


```
%ImageOut = rotate(ImageIn, Theta)
%
%Rotates the Image by Theta degrees.
load clown
In = clown
Theta = pi/2
ImageOut = rotate(In, Theta)

function [Out] =  rotate(In, Theta)
%Work out Width and Height of Source image
width=size(In,1);
height=size(In,2);

%Work out the centre point of the image, since we want to rotate about this point.
cp = [round(size(In,1)/2), round(size(In,2)/2)];

%The forward transformation matrix
tm = [ cos(Theta), sin(Theta) ;
       -sin(Theta), cos(Theta) ]

%Calculate the reverse mapping by matrix inversion
rtm = inv (tm);

 for y=1:height
  for x=1:width
   p =[x,y];         %Point on the destination image
   tp = round(rtm*(p-cp)'+cp');  %Calculate nearest corresponding point on the source image
   if tp(1)<1 | tp(2)<1 | tp(1)>width | tp(2)>height
    Out(x,y)=0;           %If we are outside the bounds of the image set to black
   else
    Out(x,y)=In(tp(1),tp(2));    %Else use the source image
   end
  end
 end
 end
```

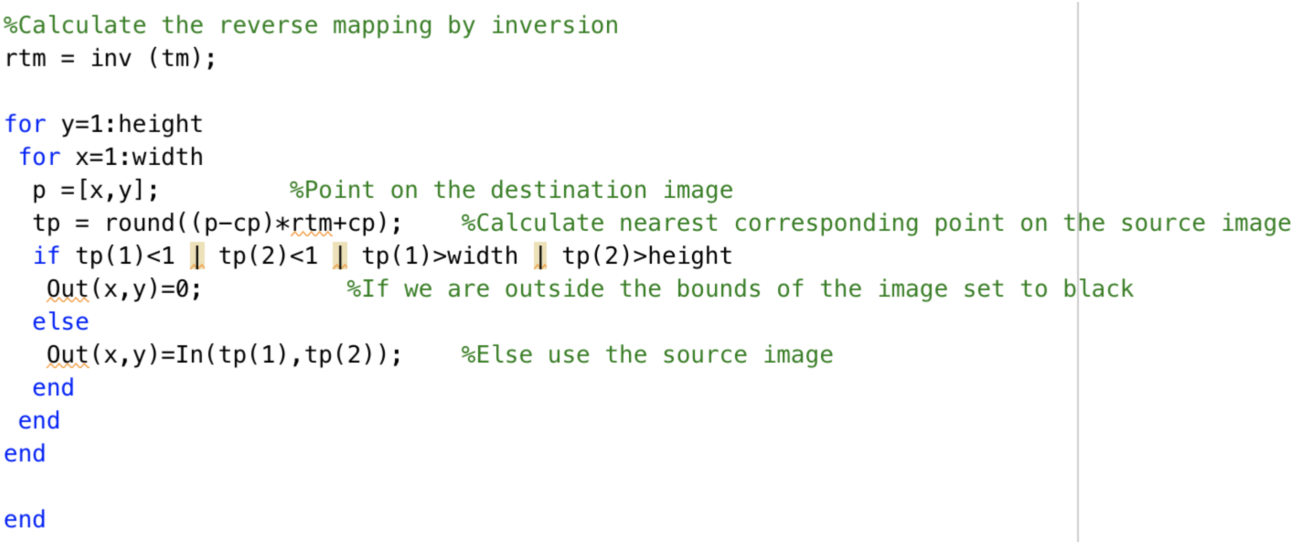




Task 2 - Shearing



```
%ImageOut = Shear(ImageIn, xshear, yshear)
%
%Shears the Image.
load clown
In = clown
xshear = 1
yshear = 0
ImageOut = shear(In, xshear, yshear)

function [Out] =  shear(In, xshear, yshear)

%Work out Width and Height of Source image
width=size(In,1);
height=size(In,2);

%Work out the centre point of the image, since we want to shear about this point.
cp = [round(size(In,1)/2), round(size(In,2)/2)];

%The forward transformation matrix
tm = [ 1, xshear ;
       yshear, 1 ];
```



```
%Calculate the reverse mapping by inversion
rtm = inv (tm);

for y=1:height
 for x=1:width
  p =[x,y];         %Point on the destination image
  tp = round((p-cp)*rtm+cp);    %Calculate nearest corresponding point on the source image
  if tp(1)<1 | tp(2)<1 | tp(1)>width | tp(2)>height
   Out(x,y)=0;           %If we are outside the bounds of the image set to black
  else
   Out(x,y)=In(tp(1),tp(2));    %Else use the source image
  end
 end
end

end
```