A. IMAG EDGE DETECTION #1

1. Simpan program berikut dengan nama coims.jpg, kemudian jalankan:
   Dalam laporan anda harus menjelaskan bagaimana rumus edge detection dengan metode Sobel dan Canny.

```
I = imread('coins.jpg');
imshow(I);
 % menamplkan  deteksi sobel dan canny
BW1 = edge(I,'sobel');
BW2 = edge(I,'canny');
imshow(BW1)
figure, imshow(BW2)
```

2. Tulis dan jalankan program berikut, dalam laporan, anda harus menjelaskan tiap baris dalam program ini.

```
% -------------------------1--------------
I = imread('coins.jpg');
imshow(I);
%-------------------------2--------------------
BW = im2bw(I);
imshow(BW);
%----------------------3-------------------------
dim = size(BW)
col = round(dim(2)/2)-90;
row = min(find(BW(:,col)))
%---------------------------------------4---------------------
boundary = bwtraceboundary(BW,[row, col],'N');

%---------------------------------------5---------------------
imshow(I)
hold on;
plot(boundary(:,2),boundary(:,1),'g','LineWidth',3);
%-----------------------------------------------------6---------------------
BW_filled = imfill(BW,'holes');
boundaries = bwboundaries(BW_filled);
%-----------------------------------------------end of this program----------------
for k=1:10
  b = boundaries{k};
  plot(b(:,2),b(:,1),'g','LineWidth',3);
end
```

B. IMAGE EDGE DETECTION #2

RUN program berikut, setelah anda RUN hanya gambar hasil yg sudah di-deteksi pinggir yang muncul sebagai output. Tugas anda setelah itu adalah mengedit (menambah perintah) untuk menampilkan gambar (citra) asli. Tampilkan citra asli (gambar 1) dan citra hasil (gambar 2) dalam 1 window.

```
%% Edge Detection on Images
%% Create a New Folder and Copy Relevant Files
% The following code will create a folder in your current
% working folder (pwd). The new folder will only contain the files
% that are relevant for this example. If you do not want to affect the
% current folder (or if you cannot generate files in this folder),
% you should change your working folder.
%% About the 'sobel' Function
% The
<matlab:edit(fullfile(matlabroot,'toolbox','coder','codegendemos','coderdemo_edge
_detection','sobel.m')) sobel.m>
% function takes an image (represented as a double matrix) and a threshold
% value and returns an image with the edges detected (based on the
% threshold value).
type sobel

%% Generate the MEX Function
% Generate a MEX function using the 'codegen' command.
codegen sobel
%%
% Before generating C code, you should first test the MEX function in
% MATLAB to ensure that it is functionally equivalent to the original
% MATLAB code and that no run-time errors occur.
% By default, 'codegen' generates a MEX function named 'sobel_mex'
% in the current folder. This allows you to test the MATLAB code and MEX
% function and compare the results.

%% Read in the Original Image
% Use the standard 'imread' command.
im = imread('hello.jpg');
image(im);

%% Convert Image to a Grayscale Version
% Convert the color image (shown above) to an equivalent grayscale image
% with normalized values (0.0 for black, 1.0 for white).
gray = (0.2989 * double(im(:,:,1)) + 0.5870 * double(im(:,:,2)) + 0.1140 *
double(im(:,:,3)))/255;
```

```
%% Run the MEX Function (The Sobel Filter)
% Pass the normalized image and a threshold value.
edgeIm = sobel_mex(gray, 0.7);

%% Display the Result
im3 = repmat(edgeIm, [1 1 3]);
image(im3);

%% Generate Standalone C Code
codegen -config coder.config('lib') sobel
%%
% Using 'codegen' with the '-config coder.config('lib')' option produces
% a standalone C library.
% By default, the code generated for the library is in the folder
% codegen/lib/sobel/

%% Inspect the Generated Function
type codegen/lib/sobel/sobel.c

%% Cleanup
% Remove files and return to original folder
%% Run Command: Cleanup
cleanup

displayEndOfDemoMessage(mfilename)
```