**Ayesha Khalid**

**BIT21011 Morning**

Euclidean Distance

Cx = 50; % Center x-coordinate

Cy = 50; % Center y-coordinate

Radius = 20; % Radius for the Euclidean distance threshold

A = zeros(100, 100); % Initialize the matrix to all zeros

for i = 1:100

for j = 1:100

% Calculate Euclidean distance and check if it is within the radius

if ((i - Cx)^2 + (j - Cy)^2) <= Radius^2

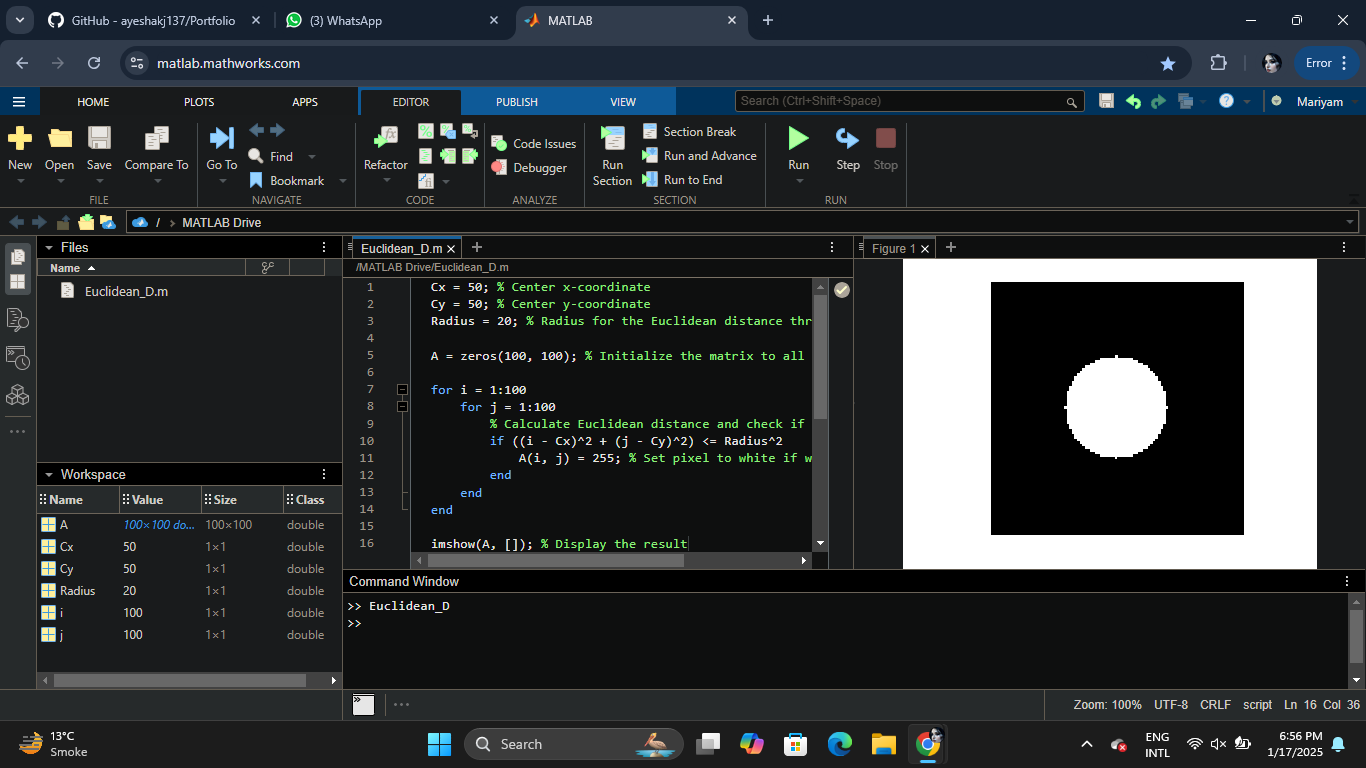
A(i, j) = 255; % Set pixel to white if within the radius

end

end

end

imshow(A, []); % Display the result



**City\_Block Distance**

Cx = 50; % Center x-coordinate

Cy = 50; % Center y-coordinate

Radius = 20; % Radius for the City Block distance threshold

A = zeros(100, 100); % Initialize the matrix to all zeros

for i = 1:100

for j = 1:100

% Calculate City Block (Manhattan) distance

if abs(i - Cx) + abs(j - Cy) <= Radius

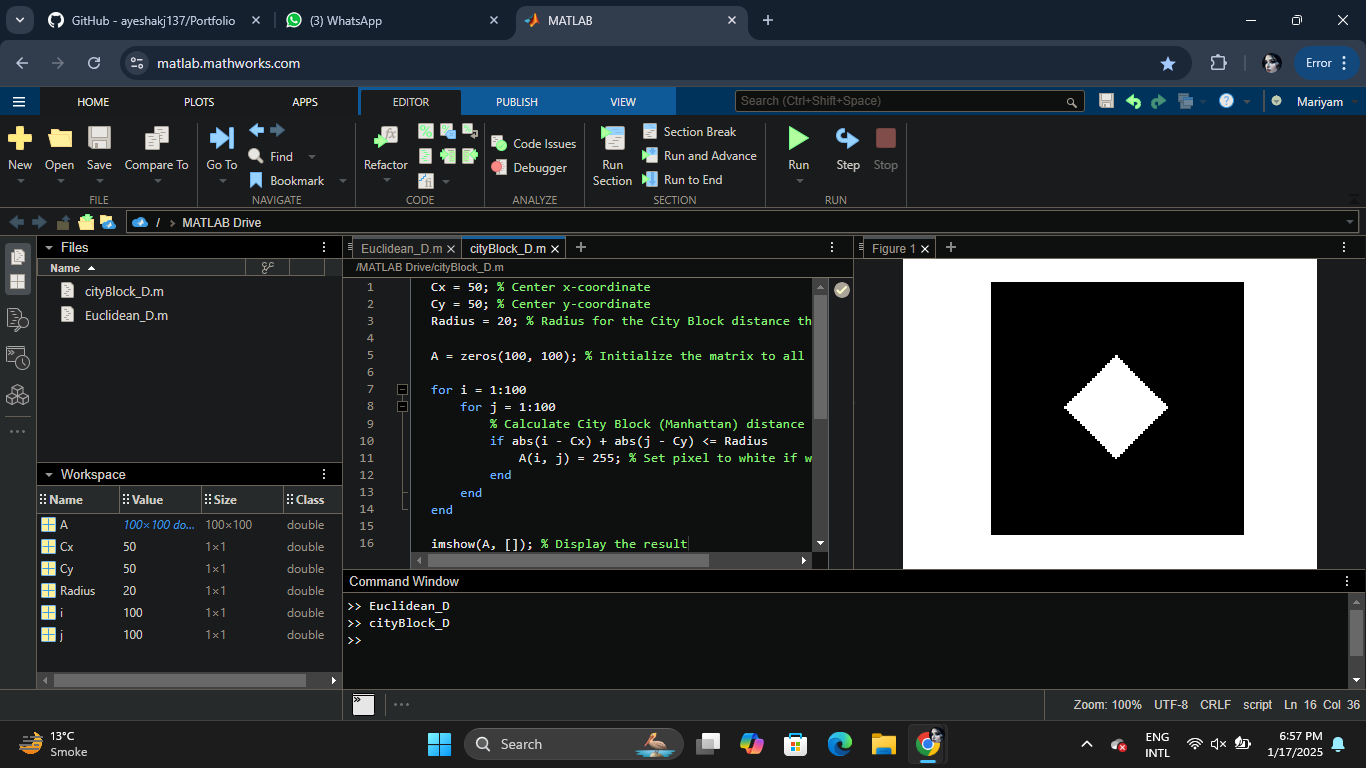
A(i, j) = 255; % Set pixel to white if within the radius

end

end

end

imshow(A, []); % Display the result



**Chess\_Board distance**

Cx = 50; % Center x-coordinate

Cy = 50; % Center y-coordinate

Radius = 20; % "Radius" (threshold for the Chessboard distance)

A = zeros(100, 100); % Initialize the matrix to all zeros

for i = 1:100

for j = 1:100

% Apply the Chessboard (Chebyshev) distance condition

if max(abs(i - Cx), abs(j - Cy)) <= Radius

A(i, j) = 255; % Set pixel to white if within the "radius"

end

end

end

imshow(A, []); % Display the result

