```
%% MathTools Lab 1 (HW0)
%Date: Sep 5th 2024
```

```
%% Question 1
%a)
%Testing 1: Project [3, 4] onto [0.6, 0.8]
v = [3, 4];
u = [0.6, 0.8];

length_projected_vector = project(v, u);
disp(length_projected_vector);
```

```
    2.7785
```

```

%Testing 1: Project [7, 8, 9] onto [0, 0, 1]
v = [7, 8, 9];
u = [0, 0, 1];

length_projected_vector = project(v, u);
disp(length_projected_vector);
```

```
     9
```

```


%% Question 1
%b)
%Testing Gram-Schmidt process
v = [3, 4];
u = [0.6, 0.8];
[coordinates_v, matrix_uw] = changeOfCoords(v, u);
disp('Coordinates of v in the new coordinate system:');
```

```
Coordinates of v in the new coordinate system:
```

```
disp(coordinates_v);
```

```
    5.0000
   -0.0000
```

```

disp('Matrix [u, w]:');
```

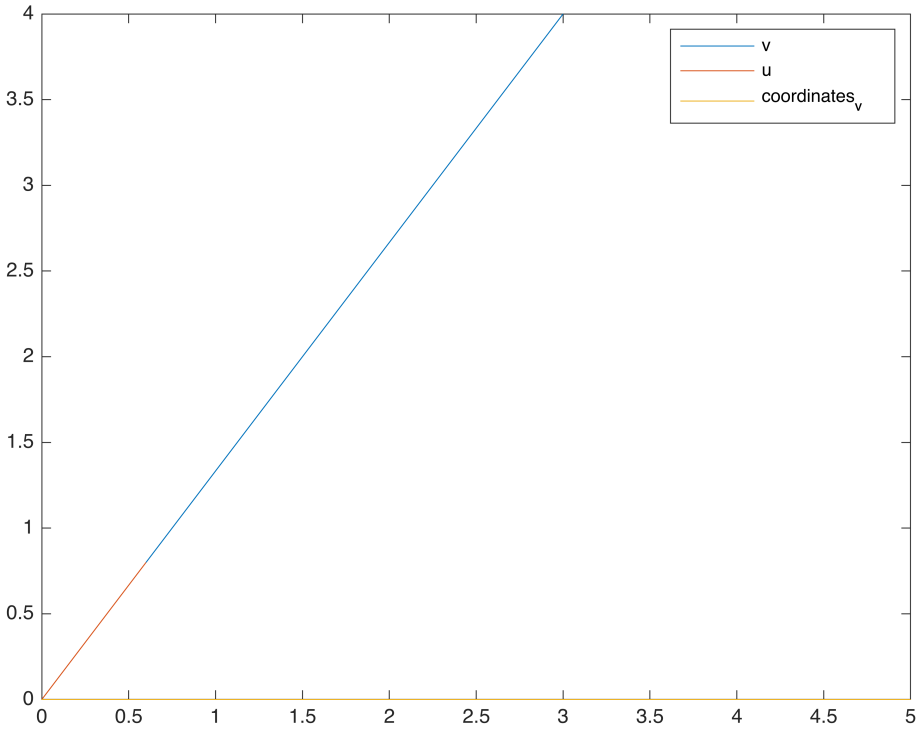
```
Matrix [u, w]:
```

```
disp(matrix_uw);
```

```
    0.6000   -0.8000
    0.8000    0.6000
```

```

plot([0,v(1)],[0,v(2)]);hold on;
```

```
plot([0,u(1)],[0,u(2)]);hold on;
plot([0,coordinates_v(1)], [0,coordinates_v(2)]); hold on;
legend('v','u','coordinates_v');
```




```
testing = (matrix_uw * coordinates_v)';
disp(testing);
```

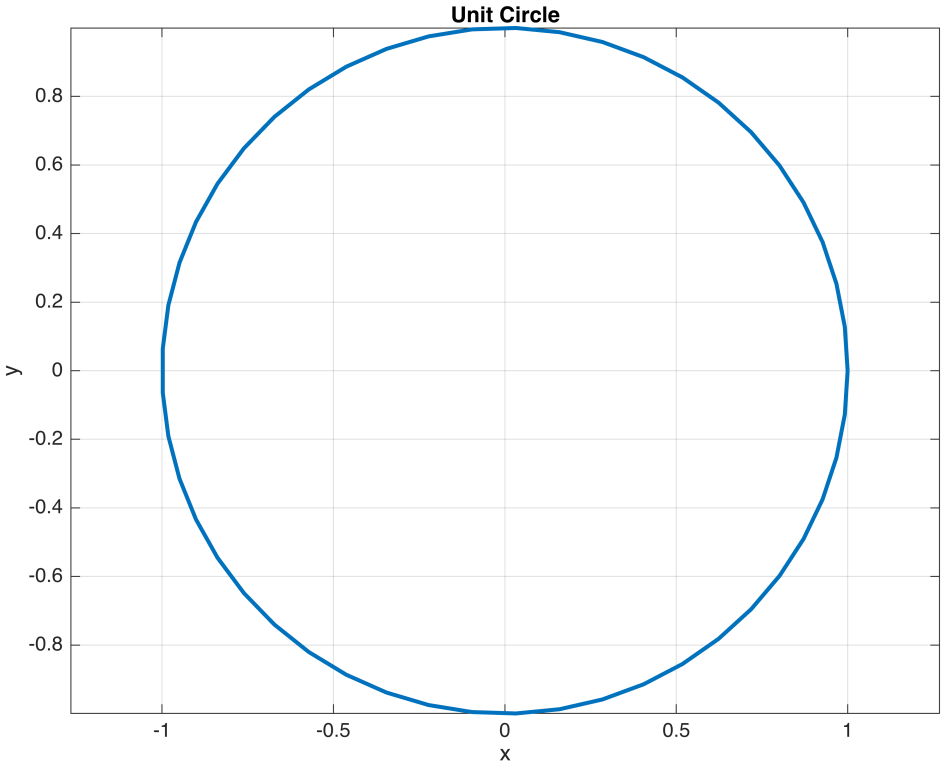
```
    3.0000    4.0000
```

```
%% Functions

%1a)
```

```
function length_projected_vector = project(v, u)

%ensure that u is a unit factor
if norm(u) ~= 1
    error('change u to a unit vector')
end

%projecting v onto u
%proj_v_on_u = dot(v, u) * u;
proj_v_on_u = (v.*u).*u;
%length_projected_vector = norm(proj_v_on_u);
```

```
length_projected_vector = sqrt(sum(proj_v_on_u.^2));

end

%1b)
function [coordinates_v, matrix_uw] = changeOfCoords(v, u)
% v: created orthogonal vector in a new ”coordinate system”
% u: one of the new axes (will be normalized)

%making u an unit vector
%u = u / norm(u);
u = u / sqrt(sum(u.^2));


%creating an orthogonal unit vector w
w = [-u(2), u(1)];

%List the matrix
matrix_uw = [u(:), w(:)];

%compute the coordinates
coordinates_v = matrix_uw \ v(:);
end
```

```
%% Question 2
%define theta
theta = linspace(0, 2*pi, 50);

%coordinates of the unit circle
x = cos(theta);
y = sin(theta);

% Plot the unit circle
figure;
plot(x, y, 'LineWidth', 2);
axis equal; xlabel('x'); ylabel('y');
title('Unit Circle');
grid on;
```



```
%% Question 3a)
input_vector = rand(8,1);
scalar_threshold = 0.5;

%testing
decision = neuron(input_vector, scalar_threshold);

disp('Input vector:');
```

```
Input vector:
```

```
disp(input_vector);
```

```
    0.9503
    0.4980
    0.8175
    0.7357
    0.1582
    0.7916
    0.0137
    0.7235
```

```
disp('Does the neuron fire? (1 = yes, 0 = no)');
```

```
Does the neuron fire? (1 = yes, 0 = no)
```

```
disp(decision);
```

```
     1
```

```


%% Question 3b)
scalar_threshold = 0.5;
T = 60;

%3c)
time_vector = 1:T;
plot_vector = zeros(1, T);


for xx = time_vector
    plot_vector(xx) = neuronOverTime(scalar_threshold, T);
end
```

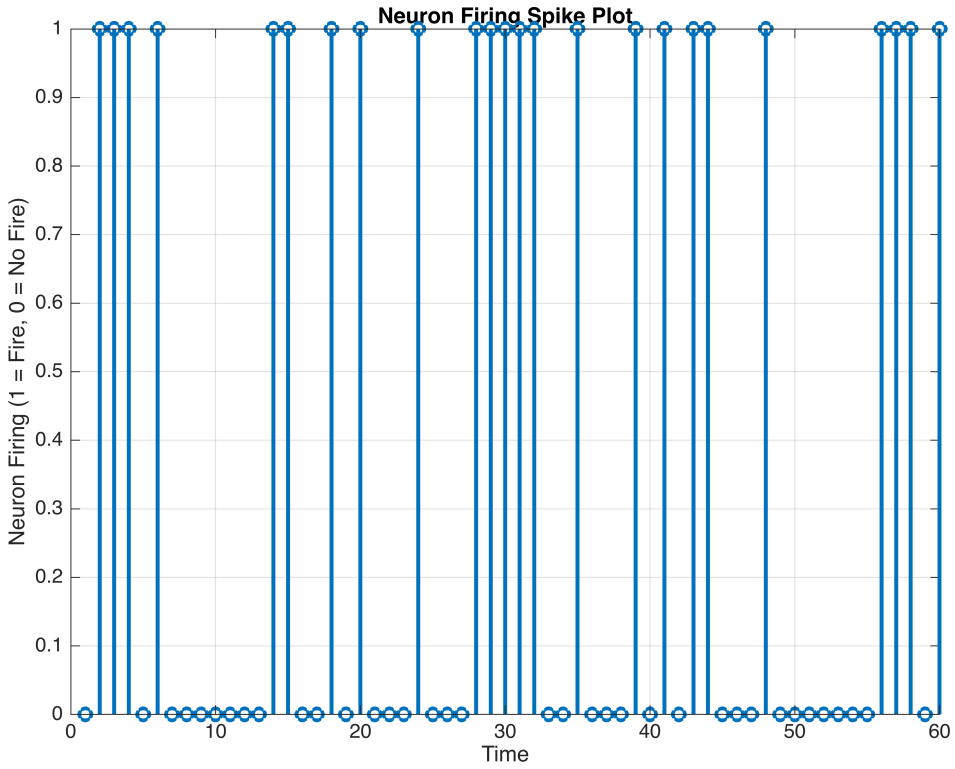
```
Firing activity over time:
     0

Firing activity over time:
     1

Firing activity over time:
     1

Firing activity over time:
     1
```

```
Firing activity over time:
     0

Firing activity over time:
     1

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     1

Firing activity over time:
     1

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     1

Firing activity over time:
     0

Firing activity over time:
     1

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     1

Firing activity over time:
     0

Firing activity over time:
```

```
     0

Firing activity over time:
     0

Firing activity over time:
     1

Firing activity over time:
     1

Firing activity over time:
     1

Firing activity over time:
     1

Firing activity over time:
     1

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     1

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     1

Firing activity over time:
     0

Firing activity over time:
     1

Firing activity over time:
     0

Firing activity over time:
     1

Firing activity over time:
     1

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0
```

```
Firing activity over time:
     1

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     0

Firing activity over time:
     1

Firing activity over time:
     1

Firing activity over time:
     1

Firing activity over time:
     0

Firing activity over time:
     1
```

```
figure;
stem(time_vector, plot_vector, 'LineWidth', 2);hold on;
xlabel('Time');
ylabel('Neuron Firing (1 = Fire, 0 = No Fire)');
title('Neuron Firing Spike Plot');
grid on;
```




```
%3a)
```

```
function decision = neuron(input_vector, scalar_threshold)
dendrite1 = mean(input_vector);
dendrite2 = mean(input_vector(5:8));
dendrite3 = input_vector(8);

%check firing or not
check_firing = 0;
if dendrite1 > scalar_threshold
    check_firing = check_firing +1;
end

if dendrite2 > scalar_threshold
    check_firing = check_firing +1;
end

if dendrite3 > scalar_threshold
    check_firing = check_firing +1;
end

if check_firing >=2
    decision = 1;
else
```

```
    decision = 0;
end

end

%3b)
function decision_over_time = neuronOverTime(scalar_threshold, T)
input_vector = rand(1, T);
padded_input = [zeros(1, 7), input_vector];

for t = 1:T
    real_inputs = padded_input(t:t+7)';
    decision_over_time = neuron(real_inputs, scalar_threshold);
end

    disp('Firing activity over time:');
    disp(decision_over_time);

end
```