Question 1.1

```
% Create the matrix a
a = [1 4 3; 4 2 6; 7 8 9]; % Creates a 3x3 matrix a

% Calculate determinant, inverse, and transpose
determinant_a = det(a); % Calculates determinant and stores in determinant_a
inverse_a = inv(a);    % Calculates inverse and stores in inverse_a
transpose_a = a';     % Calculates transpose and stores in transpose_a

% Find row vectors of minimum and maximum elements, and smallest and largest
elements
min_elements = min(a);         % Finds minimum in each column, stores in
min_elements
max_elements = max(a);         % Finds maximum in each column, stores in
max_elements
smallest_element = min(min(a));  % Finds smallest element overall, stores in
smallest_element
largest_element = max(max(a));  % Finds largest element overall, stores in
largest_element

% Calculate element-wise square, sum of each column, and sum of all elements
squared_a = a.^2;        % Performs element-wise square, stores in squared_a
column_sums = sum(a);     % Calculates sum of each column, stores in
column_sums
total_sum = sum(sum(a));  % Calculates sum of all elements, stores in total_sum

% Get matrix dimensions
[rows, columns] = size(a); % Gets number of rows and columns, stores in rows
and columns

% Create a row vector
b = [4 5 6];

% Display results
disp('Determinant of a:');
```

```
Determinant of a:
```

```
disp(determinant_a);
```

```
   48.0000
```

```
disp('Inverse of a:');
```

```
Inverse of a:
```

```
disp(inverse_a);
```

```
   -0.6250   -0.2500    0.3750
```

```
    0.1250   -0.2500    0.1250
    0.3750    0.4167   -0.2917
```

```
disp('Transpose of a:');
```

```
Transpose of a:
```

```
disp(transpose_a);
```

```
     1     4     7
     4     2     8
     3     6     9
```

```
disp('Minimum elements in each column:');
```

```
Minimum elements in each column:
```

```
disp(min_elements);
```

```
     1     2     3
```

```
disp('Smallest element in the matrix:');
```

```
Smallest element in the matrix:
```

```
disp(smallest_element);
```

```
     1
```

```
disp('Maximum elements in each column:');
```

```
Maximum elements in each column:
```

```
disp(max_elements);
```

```
     7     8     9
```

```
disp('Largest element in the matrix:');
```

```
Largest element in the matrix:
```

```
disp(largest_element);
```

```
     9
```

```
disp('Element-wise square of a:');
```

```
Element-wise square of a:
```

```
disp(squared_a);
```

```
     1    16     9
    16     4    36
    49    64    81
```

```
disp('Sum of each column:');
```

```
Sum of each column:
```

```
disp(column_sums);
```

```
    12    14    18
```

```
disp('Sum of all elements:');
```

```
Sum of all elements:
```

```
disp(total_sum);
```

```
    44
```

```
disp('Number of rows:');
```

```
Number of rows:
```

```
disp(rows);
```

```
     3
```

```
disp('Number of columns:');
```

```
Number of columns:
```

```
disp(columns);
```

```
     3
```

```
% Explain sub-matrix operations
disp('Explanation of sub-matrix operations:');
```

```
Explanation of sub-matrix operations:
```

```
a(1:2, 1:2)
```

```
ans = 2×2
     1     4
     4     2
```

```
disp('- a(1:2, 1:2): Selects elements from rows 1 and 2, and columns 1 and 2.');
```

```
- a(1:2, 1:2): Selects elements from rows 1 and 2, and columns 1 and 2.
```

```
a(1:2)
```

```
ans = 1×2
     1     4
```

```
disp('- a(1:2): Selects rows the range of element starting from 1st row 1st column 1st element to 2nd element');
```

```
- a(1:2): Selects rows the range of element starting from 1st row 1st column 1st element to
2nd element
```

```
a(3) % This line remains the same
```

```
ans = 7
```

```
disp('- a(3): Selects the third element (which is the entire third row in this case).');
```

```
- a(3): Selects the third element (which is the entire third row in this case).
```

Question 1.2

```
% Generate a 6x6 matrix A
A = rand(6); % Creates a 6x6 matrix with random elements

% Generate a 6x1 matrix z
z = rand(6, 1); % Creates a 6x1 matrix with random elements

% Solve linear system of equations Ax = z
x = A \ z; % Solves Ax = z and stores the solution in x

% Compute determinant of matrix A
det_A = det(A); % Calculates the determinant of A and stores it in det_A

% Extract a 4x4 matrix from the 6x6 matrix
A_sub = A(1:4, 1:4); % Extracts a 4x4 sub-matrix from the top-left corner

% Extract a 4x1 matrix from the 6x1 matrix
z_sub = z(1:4); % Extracts a 4x1 sub-matrix from the top 4 elements

% Display results
disp('Solution to Ax = z:');
```

```
Solution to Ax = z:
```

```
disp(x);
```

```
   -0.8261
    0.8344
   -0.0316
    0.0396
   -1.6430
    1.9476
```

```
disp('Determinant of A:');
```

```
Determinant of A:
```

```
disp(det_A);
```

```
   -0.0208
```

```
disp('Extracted 4x4 sub-matrix:');
```

```
Extracted 4x4 sub-matrix:
```

```
disp(A_sub);
```

```
    0.5752    0.0430    0.5470    0.3685
    0.0598    0.1690    0.2963    0.6256
    0.2348    0.6491    0.7447    0.7802
    0.3532    0.7317    0.1890    0.0811
```

```
disp('Extracted 4x1 sub-matrix:');
```

```
Extracted 4x1 sub-matrix:
```

```
disp(z_sub);
```

```
    0.3507
    0.9390
    0.8759
    0.5502
```

```
% Explanation of sub-matrix operations
disp('Explanation of sub-matrix operations:');
```

```
Explanation of sub-matrix operations:
```

```
disp('- A(1:4, 1:4): Selects elements from rows 1 to 4 and columns 1 to 4.');
```

```
- A(1:4, 1:4): Selects elements from rows 1 to 4 and columns 1 to 4.
```

```
disp('- z(1:4): Selects elements from rows 1 to 4.');
```

```
- z(1:4): Selects elements from rows 1 to 4.
```

Question 1.3

```
% Question 1.3

% 1) Calculating results
numerator1 = 35.7 * (64 - 7^4);
denominator1 = 45 + 5^3;
result1 = numerator1 / denominator1;

% 2) Calculating results
numerator2 = 3^7 * log(76);
denominator2 = 7^3 + 564;
result2 = (numerator2 / denominator2)+(nthroot(910,3));

% 3) Calculating results
angle = 5*pi/6;
result3 = cos(angle)^2 * sin(7*pi/8)^2 + tan(pi/6 * log(8)) / sqrt(7);

% 4) Creating matrix B with equal spacing in all rows
% Define starting and ending points for each row
start_values = [1, 72, 0];
end_values = [25, 24, 1];
num_elements = 9;

% Create row vectors using linspace
rows = [];
for i = 1:3
    rows = [rows; linspace(start_values(i), end_values(i), num_elements)];
end
% Combine rows into the matrix
B = rows;
% Display results
fprintf('1) %.4f\n', result1);
```

```
1) -490.7700
```

```
fprintf('2) %.4f\n', result2);
```

```
2) 20.1330
```

```
fprintf('3) %.4f\n', result3);
```

```
3) 1.4395
```

```
disp('4) ');
```

```
4)
```

```
disp(B); % Display the matrix
```

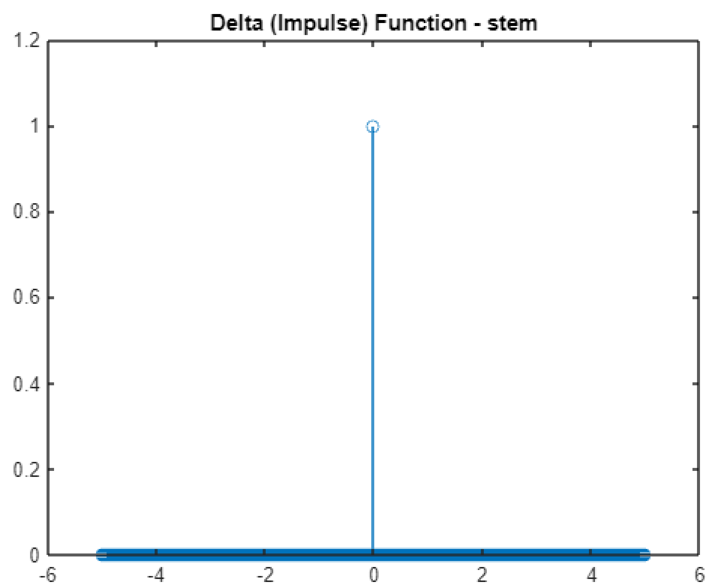
1.0000	4.0000	7.0000	10.0000	13.0000	16.0000	19.0000	22.0000	25.0000
72.0000	66.0000	60.0000	54.0000	48.0000	42.0000	36.0000	30.0000	24.0000
0	0.1250	0.2500	0.3750	0.5000	0.6250	0.7500	0.8750	1.0000

Question 1.4

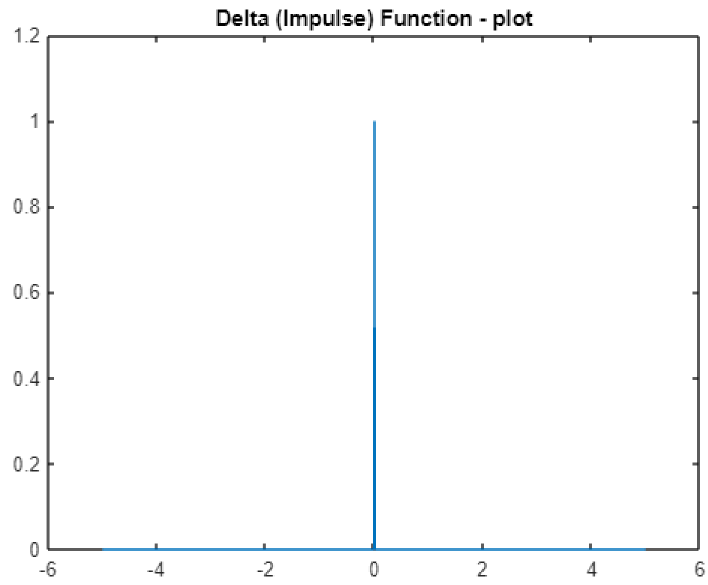
1) Write a MATLAB code that generates Delta (Impulse) Function.

- **Use the plotting function stem to make the graphs**
- **Replace the stem command in the above code with the plot command and run the code again. How does this change the plot? And why?**

```
t = -5:0.01:5;  % Time vector
impulse = t == 0;   % Impulse function (non-zero only at t=0)
% Plot with stem
figure;
stem(t, impulse);
axis([-6 6 0 1.2])
title('Delta (Impulse) Function - stem');
```




```
% Plot with plot
figure;
plot(t, impulse);
axis([-6 6 0 1.2])
title('Delta (Impulse) Function - plot');
```

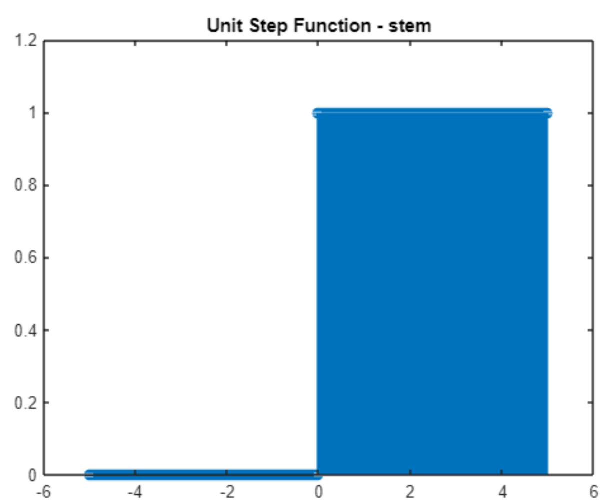


2) Write a MATLAB code that generate Unit Step function:

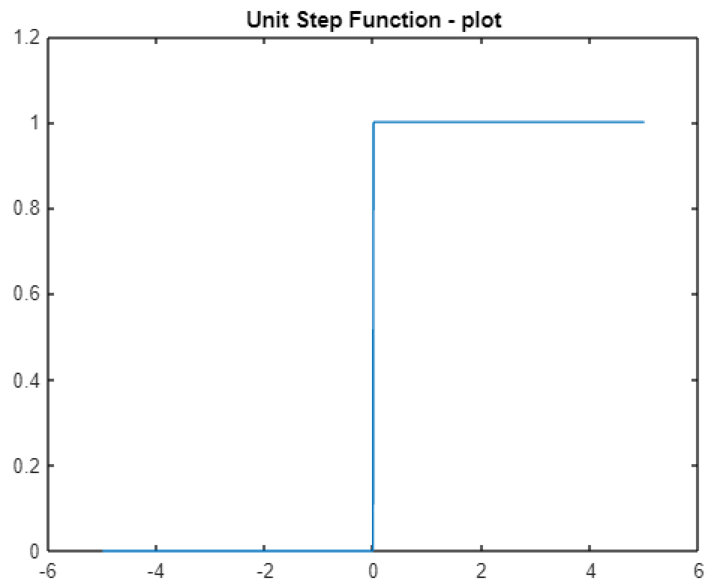
- **Use the plotting function stem to make the graphs**
- **Replace the stem command in the above code with the plot command and run the code again. How does this change the plot?**

```
t = -5:0.01:5;
step = t >= 0;    % Unit step function

% Plot with stem
figure;
stem(t, step);
axis([-6 6 0 1.2])
title('Unit Step Function - stem');
```



```
% Plot with plot
figure;
plot(t, step);
axis([-6 6 0 1.2])
title('Unit Step Function - plot');
```

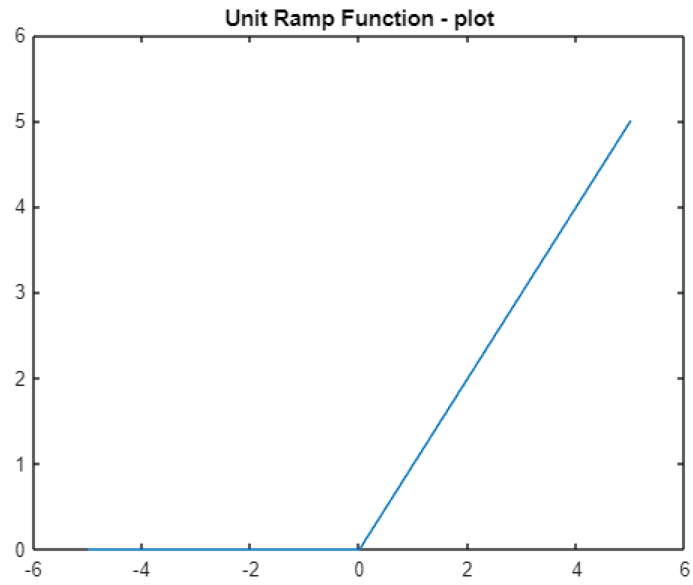


3) Write a MATLAB code that generate Unit Ramp function

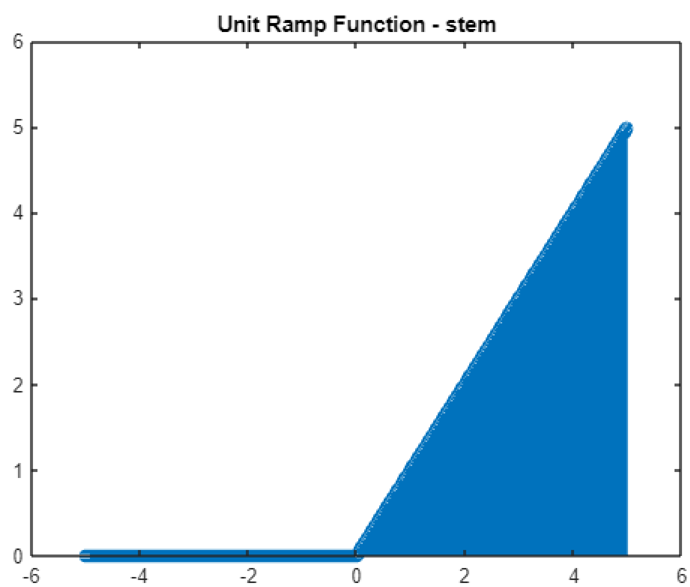
- **Use the plotting function plot to make the graphs**
- **Replace the plot command in the above code with the stem command and run the code again. How does this change the plot?**

```
t = -5:0.01:5;
ramp = t .* (t >= 0);  % Unit ramp function

% Plot with plot
figure;
plot(t, ramp);
axis([-6 6 0 6])
title('Unit Ramp Function - plot');
```




```
% Plot with stem
figure;
stem(t, ramp);
axis([-6 6 0 6])
title('Unit Ramp Function - stem');
```



Conclusion:

This MATLAB lab covered various functionalities. We explored creating, manipulating, and analyzing matrices using built-in functions and vector notation. We learned how to solve linear systems of equations and extract specific sections of matrices. Additionally, the lab covered performing mathematical computations with numerical values, functions, and roots. We also generated matrices with consistent spacing within rows or columns and visualized different functions using both stem and plot commands, observing their distinct graphical representations. Overall, this lab provided a hands-on experience with essential MATLAB functionalities for various applications in engineering, science, and mathematics.