

MATLAB – LA

1. Given are a matrix A, a matrix B, and a 9-element vector v.

$$A = \begin{bmatrix} 2 & 5 & 8 & 11 & 14 & 17 \\ 3 & 6 & 9 & 12 & 15 & 18 \\ 4 & 7 & 10 & 13 & 16 & 19 \\ 5 & 8 & 11 & 14 & 17 & 20 \\ 6 & 9 & 12 & 15 & 18 & 21 \end{bmatrix}$$

$$B = \begin{bmatrix} 5 & 10 & 15 & 20 & 25 & 30 \\ 30 & 35 & 40 & 45 & 50 & 55 \\ 55 & 60 & 65 & 70 & 75 & 80 \end{bmatrix}$$

$$v = [99 \ 98 \ 97 \ 96 \ 95 \ 94 \ 93 \ 92 \ 91]$$

Create the three arrays in the Command Window, and then, by writing one command, replace the last four columns of the first and third rows of A with the first four columns of the first two rows of B, the last four columns of the fourth row of A with the elements 5 through 8 of v, and the last four columns of the fifth row of A with columns 3 through 5 of the third row of B

- **PROGRAM:**

```
A = [2, 5, 8, 11, 14, 17;
     3, 6, 9, 12, 15, 18;
     4, 7, 10, 13, 16, 19;
     5, 8, 11, 14, 17, 20;
     6, 9, 12, 15, 18, 21];
```

```
B = [5, 10, 15, 20, 25, 30;
     30, 35, 40, 45, 50, 55;
     55, 60, 65, 70, 75, 80];
```

```
v = [99, 98, 97, 96, 95, 94, 93, 92, 91];
```

```
% modifications
```

```
A(1, 3:6) = B(1, 1:4); % Replace last 4 columns of the 1st row of A
A(3, 3:6) = B(2, 1:4); % Replace last 4 columns of the 3rd row of A
A(4, 3:6) = v(5:8); % Replace last 4 columns of the 4th row of A
A(5, 3:5) = B(3, 3:5); % Replace columns 3-5 of the 5th row of A
```

```
% result
disp('Modified matrix A:');
disp(A);
```

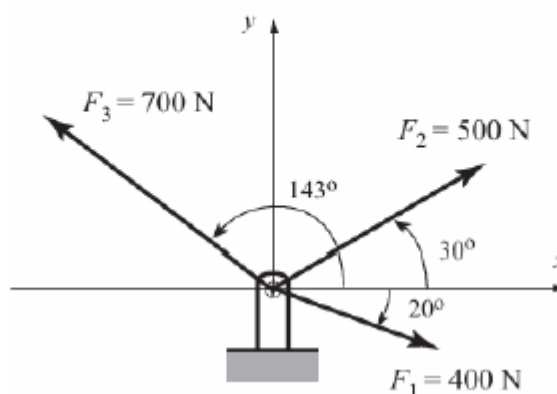
- **OUTPUT:**

```
AKANKSHA > COLLEGE > 4th yr > 7th sem > MatLab > MATLAB > LA_matlab
Editor - D:\AKANKSHA\COLLEGE\4th yr\7th sem\MatLab\MATLAB\LA_matlab\LA_q1.m
LA_q1.m
1 % Define the matrices and vector
2 A = [2, 5, 8, 11, 14, 17;
3       3, 6, 9, 12, 15, 18;
4       4, 7, 10, 13, 16, 19;
5       5, 8, 11, 14, 17, 20;
6       6, 9, 12, 15, 18, 21];
7
8 B = [5, 10, 15, 20, 25, 30;
9       30, 35, 40, 45, 50, 55;
10      55, 60, 65, 70, 75, 80];
11
12 v = [99, 98, 97, 96, 95, 94, 93, 92, 91];
13
14 % Perform the modifications
15 A(1, 3:6) = B(1, 1:4); % Replace last four columns of the first row of A
16 A(3, 3:6) = B(2, 1:4); % Replace last four columns of the third row of A

Command Window
>> LA_q1
Modified matrix A:
     2     5     5    10    15    20
     3     6     9    12    15    18
     4     7    30    35    40    45
     5     8    95    94    93    92
     6     9    65    70    75    21

fx >>
```

2. Three forces are applied to a bracket as shown. Determine the total (equivalent) force applied to the bracket:



- **PROGRAM:**

% Given forces and angles

F1 = 400; theta1 = 20; % Force in N, angle in degrees

F2 = 500; theta2 = 30;

F3 = 700; theta3 = 143;

% Convert angles to radians

theta1 = deg2rad(theta1);

theta2 = deg2rad(theta2);

theta3 = deg2rad(theta3);

% Resolve forces into components

F1x = F1 * cos(theta1);

F1y = F1 * sin(theta1);

F2x = F2 * sin(theta2);

F2y = F2 * cos(theta2);

F3x = F3 * cos(theta3);

F3y = F3 * sin(theta3);

% Calculate total x and y components

Fx_total = F1x + F2x + F3x;

Fy_total = F1y + F2y + F3y;

% Calculate resultant force and direction

F_total = sqrt(Fx_total^2 + Fy_total^2);

theta_total = atan2(Fy_total, Fx_total); % Angle in radians

% Convert angle to degrees

theta_total_deg = rad2deg(theta_total);

% Display results

disp('Total Force Components:');

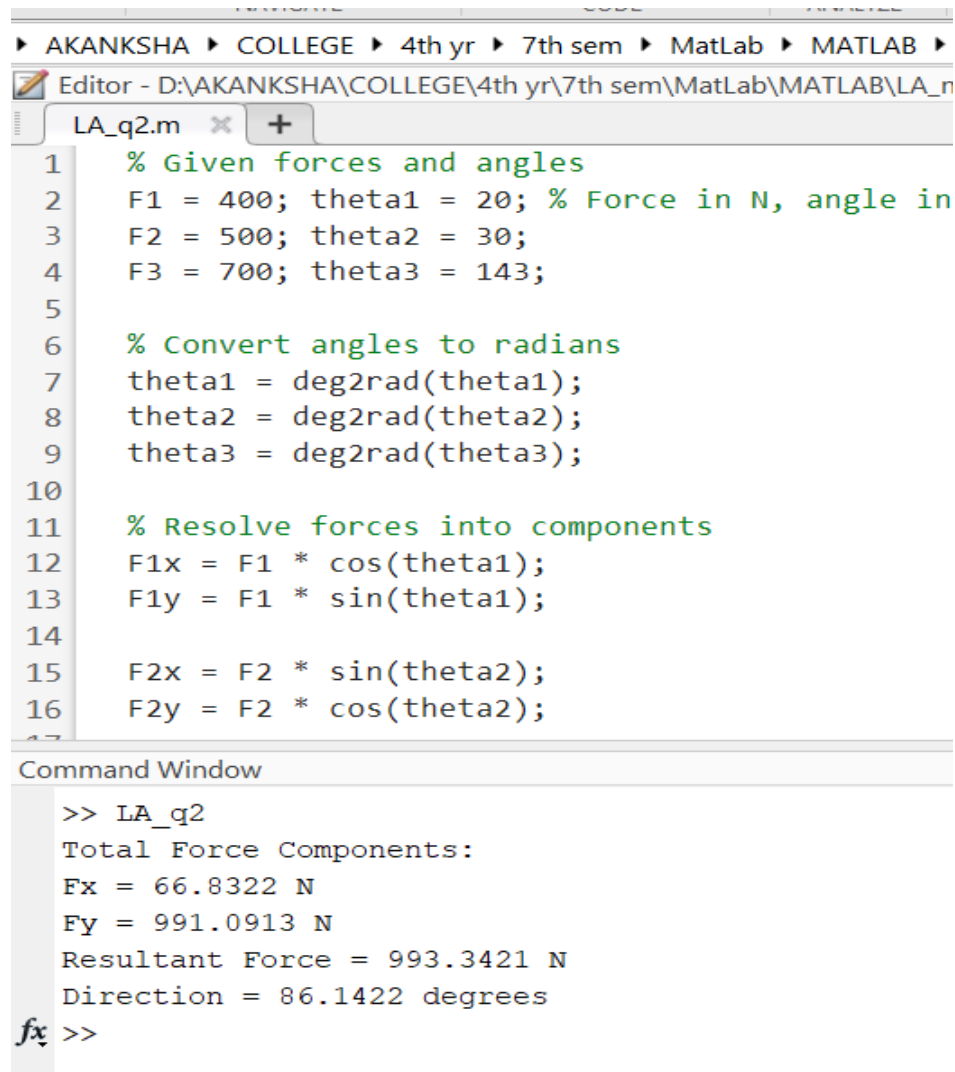
disp(['Fx = ', num2str(Fx_total), ' N']);

disp(['Fy = ', num2str(Fy_total), ' N']);

disp(['Resultant Force = ', num2str(F_total), ' N']);

disp(['Direction = ', num2str(theta_total_deg), ' degrees']);

- **OUTPUT:**



The screenshot shows the MATLAB Editor with a script named 'LA_q2.m'. The script calculates the components of three forces and their resultant. The Command Window shows the execution results.

```

1  % Given forces and angles
2  F1 = 400; theta1 = 20; % Force in N, angle in
3  F2 = 500; theta2 = 30;
4  F3 = 700; theta3 = 143;
5
6  % Convert angles to radians
7  theta1 = deg2rad(theta1);
8  theta2 = deg2rad(theta2);
9  theta3 = deg2rad(theta3);
10
11 % Resolve forces into components
12 F1x = F1 * cos(theta1);
13 F1y = F1 * sin(theta1);
14
15 F2x = F2 * sin(theta2);
16 F2y = F2 * cos(theta2);
17
Command Window
>> LA_q2
Total Force Components:
Fx = 66.8322 N
Fy = 991.0913 N
Resultant Force = 993.3421 N
Direction = 86.1422 degrees
fx >>

```

3. For the function $y = x^3 - 2x^2 + x$, calculate the value of y for the following values of x using element-by-element operations: $-2, -1, 0, 1, 2, 3, 4$.

- **PROGRAM:**

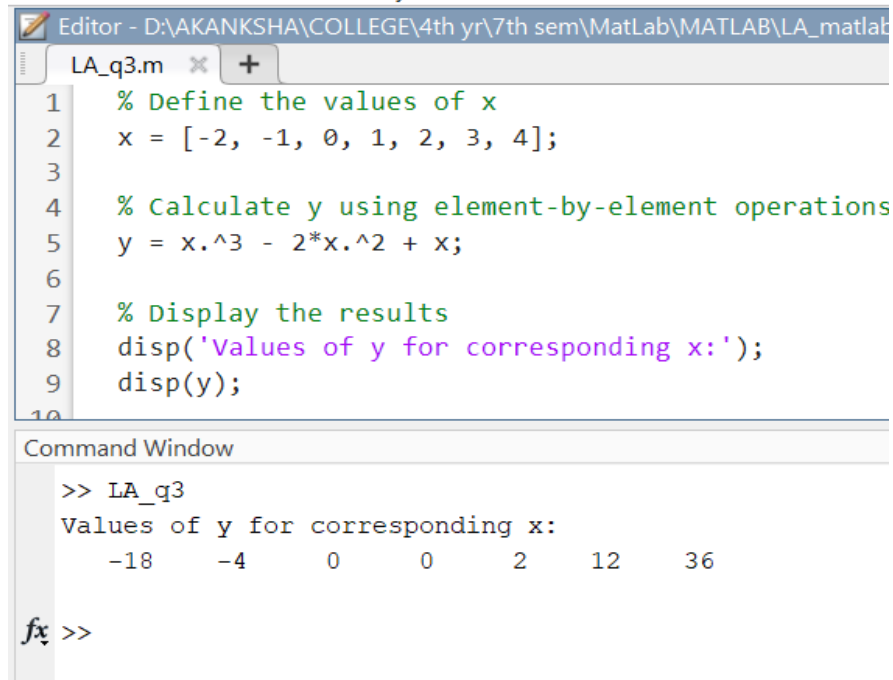
```
% Define the values of x
x = [-2, -1, 0, 1, 2, 3, 4];
```

```
% Calculate y using element-by-element operations
y = x.^3 - 2*x.^2 + x;
```

```
% Display the results
```

```
disp('Values of y for corresponding x:');  
disp(y);
```

- **OUTPUT:**



The screenshot shows the MATLAB Editor window with a script named 'LA_q3.m'. The script contains the following code:

```
1 % Define the values of x  
2 x = [-2, -1, 0, 1, 2, 3, 4];  
3  
4 % Calculate y using element-by-element operations  
5 y = x.^3 - 2*x.^2 + x;  
6  
7 % Display the results  
8 disp('Values of y for corresponding x:');  
9 disp(y);  
10
```

The Command Window shows the output of the script:

```
>> LA_q3  
Values of y for corresponding x:  
-18    -4     0     0     2    12    36  
  
fx >>
```

4. For the function $y = \frac{x^2 - 2}{x + 4}$, calculate the value of y for the following values of x using element-by-element operations: -3, -2, -1, 0, 1, 2, 3 .

- **PROGRAM:**

```
% Define the values of x  
x = [-3, -2, -1, 0, 1, 2, 3];
```

```
% Calculate y using element-by-element operations  
y = (x.^2 - 2) ./ (x + 4);
```

```
% Display the results  
disp('Values of y for corresponding x:');  
disp(y);
```

- **OUTPUT:**

```
LA_q4.m x +
% Define the values of x
x = [-3, -2, -1, 0, 1, 2, 3];

% Calculate y using element-by-element operations
y = (x.^2 - 2) ./ (x + 4);

% Display the results
disp('Values of y for corresponding x:');
disp(y);
```

```
Command Window
>> LA_q4
Values of y for corresponding x:
    7.0000    1.0000   -0.3333   -0.5000   -0.2000    0.3333    1.0000
```

5. The following two vectors are defined in MATLAB:

$$v = [3, -2, 4] \quad u = [5, 3, -1]$$

By hand (pencil and paper) write what will be displayed if the following commands are executed by MATLAB. Check your answers by executing the commands with MATLAB.

- (a) $v .* u$ (b) $v * u'$ (c) $v' * u$

- **PROGRAM:**

```
v = [3, -2, 4];
u = [5, 3, -1];

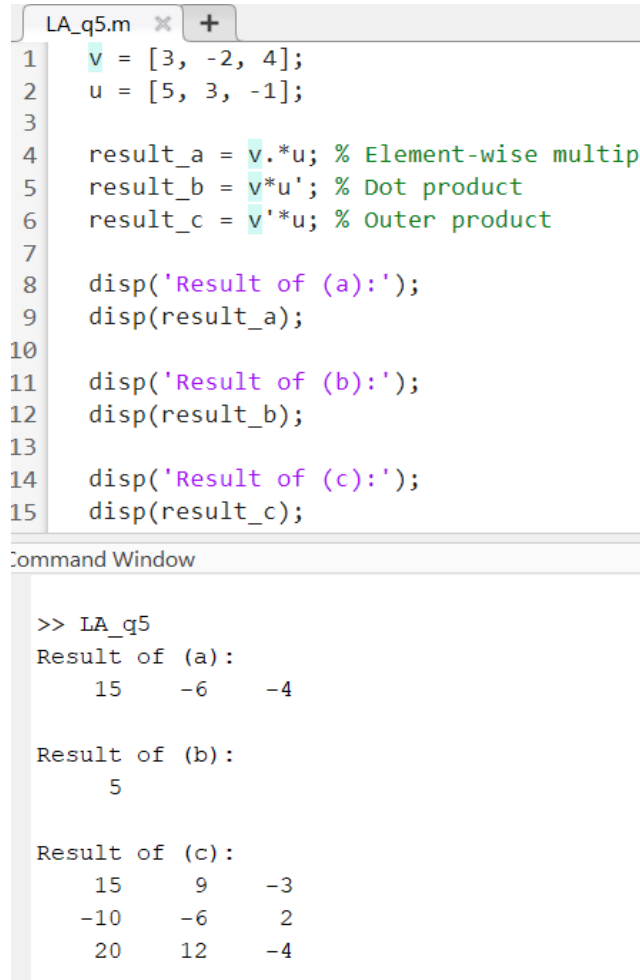
result_a = v.*u; % Element-wise multiplication
result_b = v*u'; % Dot product
result_c = v'*u; % Outer product

disp('Result of (a):');
disp(result_a);

disp('Result of (b):');
disp(result_b);
```

```
disp('Result of (c):');  
disp(result_c);
```

- **OUTPUT:**



The image shows a MATLAB script editor window with a file named 'LA_q5.m'. The script contains 15 lines of code. Lines 1-3 define vectors **v** and **u**. Lines 4-6 calculate **result_a** (element-wise multiplication), **result_b** (dot product), and **result_c** (outer product). Lines 7-15 use `disp` to display the results. Below the script editor is the Command Window, which shows the output of the script. It displays 'Result of (a):' followed by a 1x3 vector [15, -6, -4], 'Result of (b):' followed by the scalar 5, and 'Result of (c):' followed by a 3x3 matrix.

```
LA_q5.m  x  +  
1  v = [3, -2, 4];  
2  u = [5, 3, -1];  
3  
4  result_a = v.*u; % Element-wise multip  
5  result_b = v*u'; % Dot product  
6  result_c = v'*u; % Outer product  
7  
8  disp('Result of (a):');  
9  disp(result_a);  
10  
11 disp('Result of (b):');  
12 disp(result_b);  
13  
14 disp('Result of (c):');  
15 disp(result_c);  
  
Command Window  
  
>> LA_q5  
Result of (a):  
    15    -6    -4  
  
Result of (b):  
         5  
  
Result of (c):  
    15     9    -3  
   -10    -6     2  
    20    12    -4
```

6. Two vectors are given:

$$\mathbf{u} = -3\mathbf{i} + 8\mathbf{j} - 2\mathbf{k} \quad \text{and} \quad \mathbf{v} = 6.5\mathbf{i} - 5\mathbf{j} - 4\mathbf{k}$$

Use MATLAB to calculate the dot product $\mathbf{u} \cdot \mathbf{v}$ of the vectors in three ways:

- Write an expression using element-by-element calculation and the MATLAB built-in function `sum`.
- Define **u** as a row vector and **v** as a column vector, and then use matrix multiplication.
- Use the MATLAB built-in function `dot`.

- **PROGRAM:**

```
% Define the vectors
```

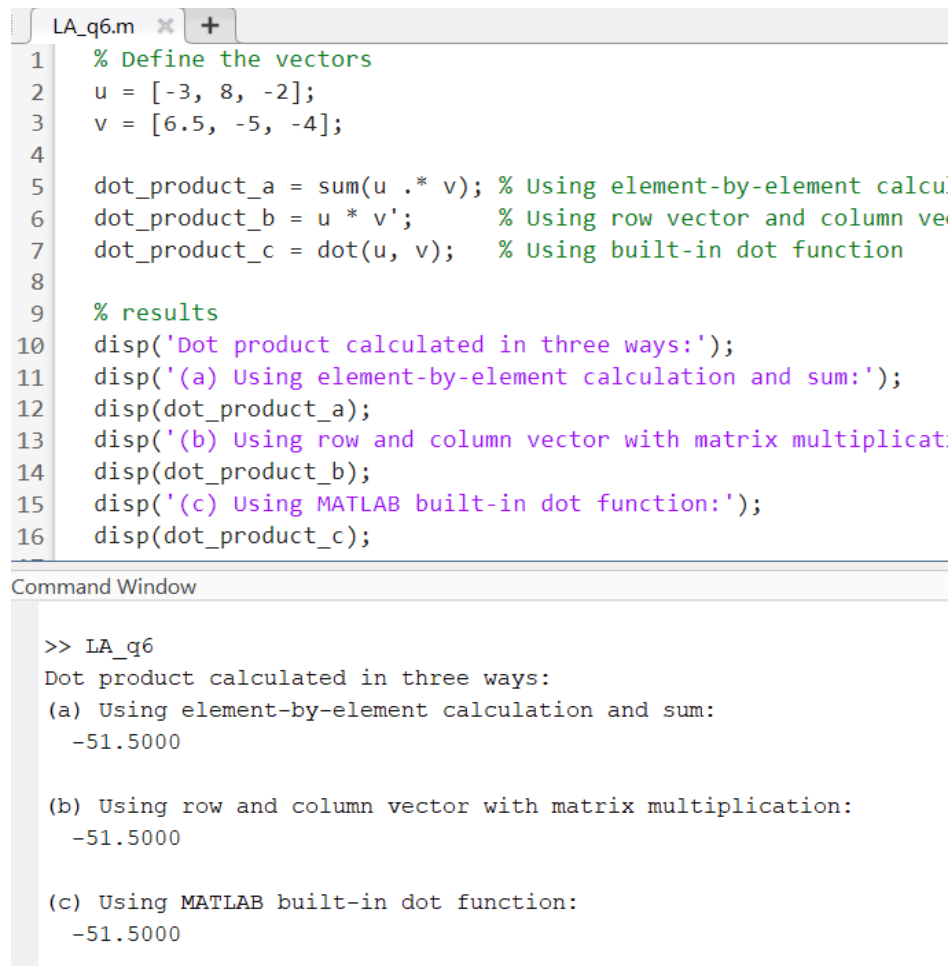
```
u = [-3, 8, -2];  
v = [6.5, -5, -4];
```

```
dot_product_a = sum(u .* v); % Using element-by-element calculation and sum  
dot_product_b = u * v'; % Using row vector and column vector with matrix  
multiplication  
dot_product_c = dot(u, v); % Using built-in dot function
```

```
% results
```

```
disp('Dot product calculated in three ways:');  
disp('(a) Using element-by-element calculation and sum:');  
disp(dot_product_a);  
disp('(b) Using row and column vector with matrix multiplication:');  
disp(dot_product_b);  
disp('(c) Using MATLAB built-in dot function:');  
disp(dot_product_c);
```

- **OUTPUT:**



```
LA_q6.m  x  +  
1  % Define the vectors  
2  u = [-3, 8, -2];  
3  v = [6.5, -5, -4];  
4  
5  dot_product_a = sum(u .* v); % Using element-by-element calcula  
6  dot_product_b = u * v'; % Using row vector and column ve  
7  dot_product_c = dot(u, v); % Using built-in dot function  
8  
9  % results  
10 disp('Dot product calculated in three ways:');  
11 disp('(a) Using element-by-element calculation and sum:');  
12 disp(dot_product_a);  
13 disp('(b) Using row and column vector with matrix multiplicat  
14 disp(dot_product_b);  
15 disp('(c) Using MATLAB built-in dot function:');  
16 disp(dot_product_c);  
  
Command Window  
  
>> LA_q6  
Dot product calculated in three ways:  
(a) Using element-by-element calculation and sum:  
-51.5000  
  
(b) Using row and column vector with matrix multiplication:  
-51.5000  
  
(c) Using MATLAB built-in dot function:  
-51.5000
```


7. Plot the function $y = 3x^3 - 26x + 10$, and its first and second derivatives, for $-2 \leq x \leq 4$, all in the same plot.

- **PROGRAM:**

```
% Define the function and its derivatives as anonymous functions
f = @(x) 3*x.^3 - 26*x + 10; % The function y = 3x^3 - 26x + 10
f_prime = @(x) 9*x.^2 - 26; % First derivative: 9x^2 - 26
f_double_prime = @(x) 18*x; % Second derivative: 18x

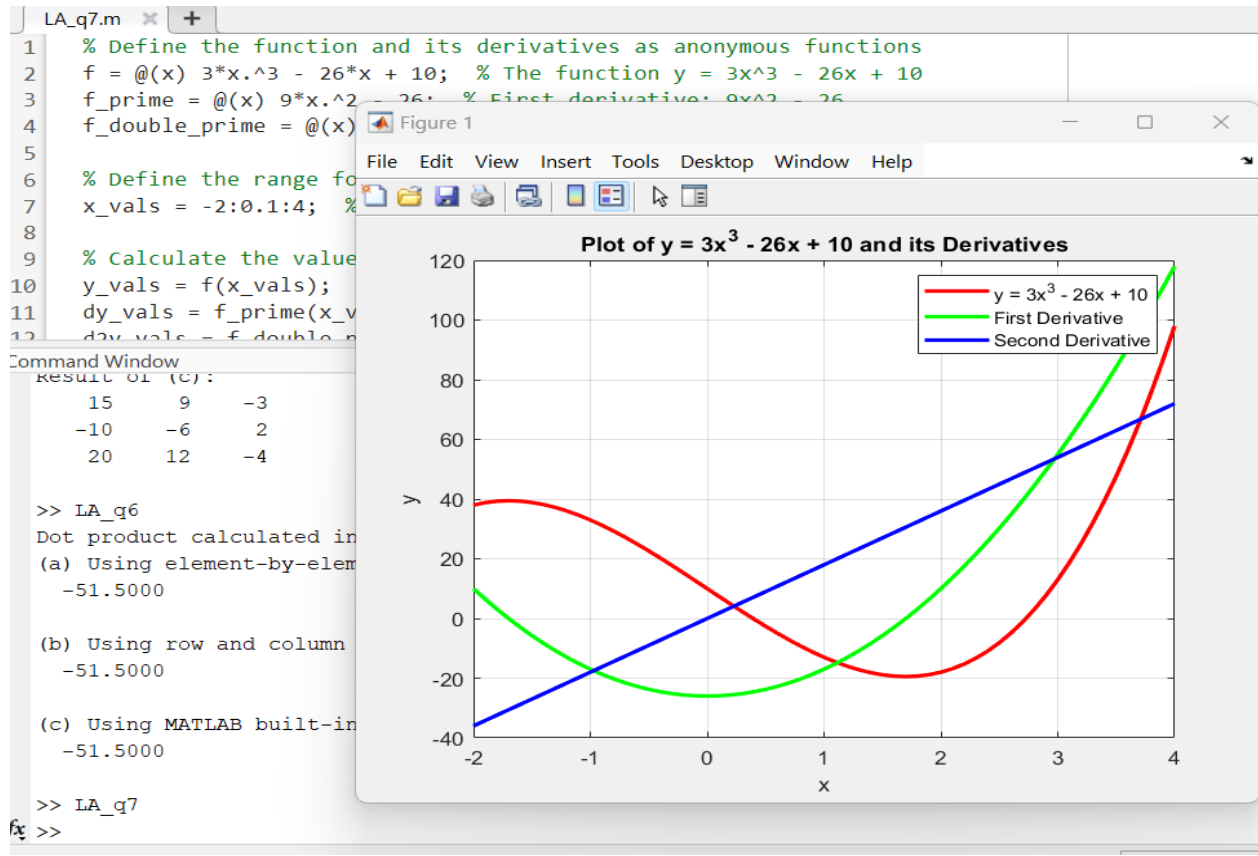
% Define the range for x
x_vals = -2:0.1:4; % From -2 to 4 with step 0.1

% Calculate the values of y, first derivative, and second derivative
y_vals = f(x_vals);
dy_vals = f_prime(x_vals);
d2y_vals = f_double_prime(x_vals);

% Plot the function and its derivatives
figure;
plot(x_vals, y_vals, 'r', 'LineWidth', 2); % Plot y in red
hold on;
plot(x_vals, dy_vals, 'g', 'LineWidth', 2); % Plot first derivative in green
plot(x_vals, d2y_vals, 'b', 'LineWidth', 2); % Plot second derivative in blue
hold off;

% Add labels and legend
xlabel('x');
ylabel('y');
title('Plot of y = 3x^3 - 26x + 10 and its Derivatives');
legend('y = 3x^3 - 26x + 10', 'First Derivative', 'Second Derivative');
grid on;
```

- **OUTPUT:**



8. Use the `fplot` command to plot the function

$$f(x) = \sqrt{|\cos(3x)|} + \sin^2(4x) \quad \text{in the domain } -2 \leq x \leq 2.$$

- **PROGRAM:**

```
f = @(x) sqrt(abs(cos(3*x))) + sin(4*x).^2; % Define the function using an anonymous function
```

```
fplot(f, [-2, 2]); % Use fplot to plot the function in the domain -2 <= x <= 2
```

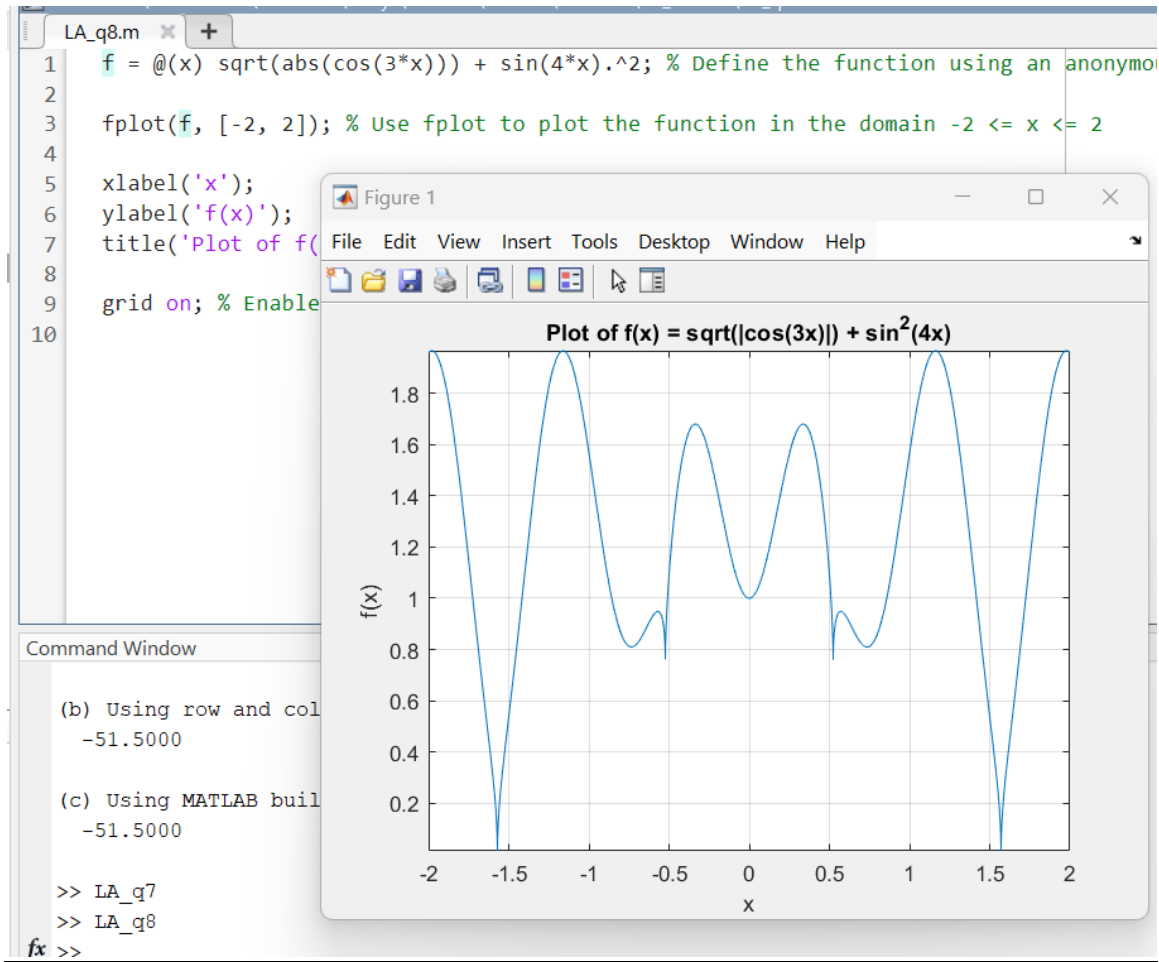
```
xlabel('x');
```

```
ylabel('f(x)');
```

```
title('Plot of f(x) = sqrt(|cos(3x)|) + sin^2(4x)');
```

```
grid on; % Enable grid
```

- **OUTPUT:**



9. A parametric equation is given by

$$x = 1.5 \sin(5t), \quad y = 1.5 \cos(3t)$$

Plot the function for $0 \leq t \leq 2\pi$. Format the plot such that the both axes will range from -2 to 2.

- **PROGRAM:**

```
% Define the parameter t from 0 to 2pi
```

```
t = linspace(0, 2*pi, 500); % 500 points for a smooth curve
```

```
% Parametric equations
```

```
x = 1.5 * sin(5 * t);
```

```
y = 1.5 * cos(3 * t);
```

```

% Plot the parametric equations
figure;
plot(x, y, 'b', 'LineWidth', 2); % Plot in blue with line width of 2

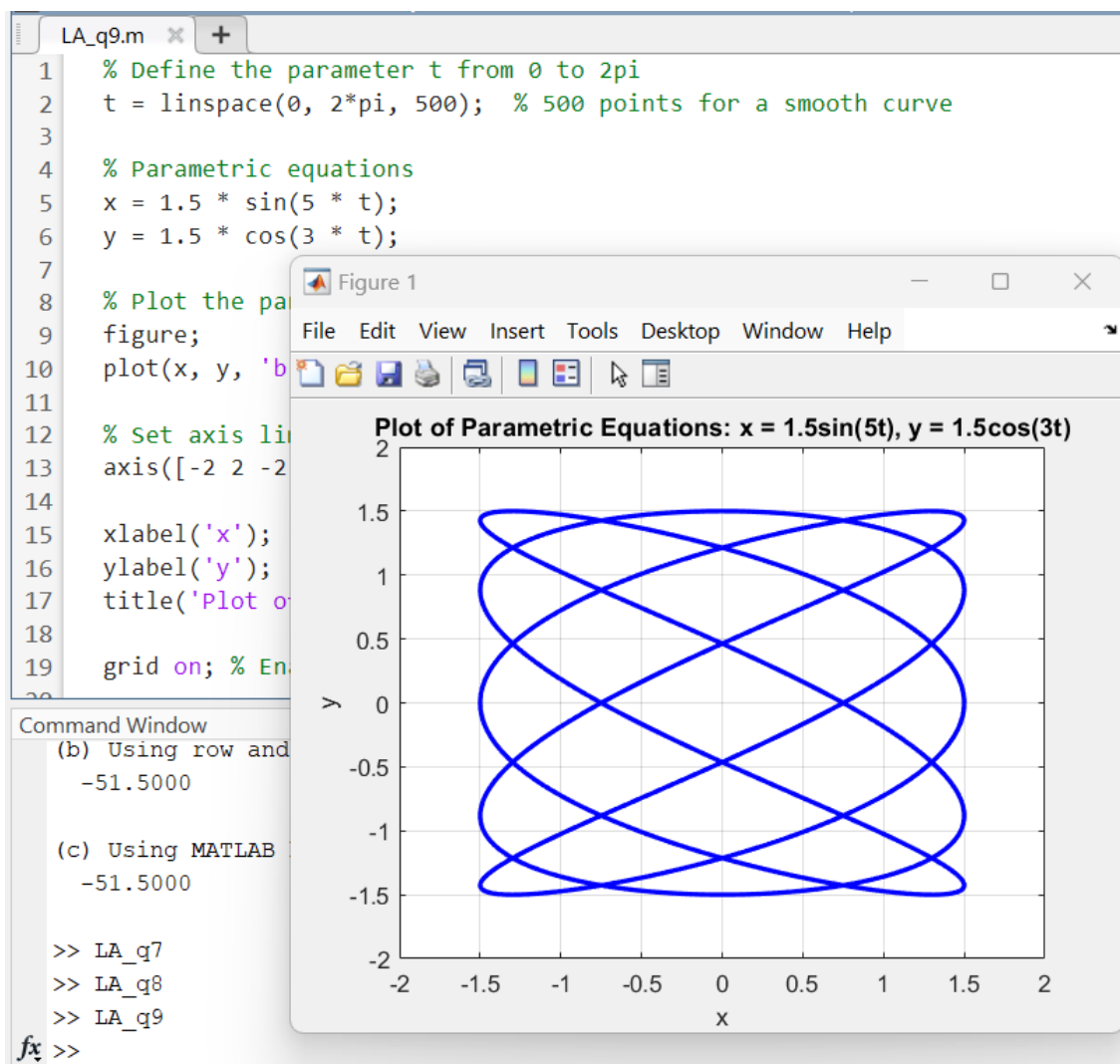
% Set axis limits
axis([-2 2 -2 2]); % Set x and y axes to range from -2 to 2

xlabel('x');
ylabel('y');
title('Plot of Parametric Equations: x = 1.5sin(5t), y = 1.5cos(3t)');

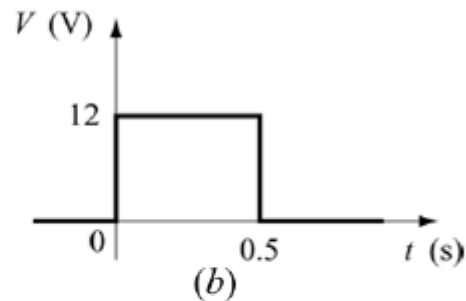
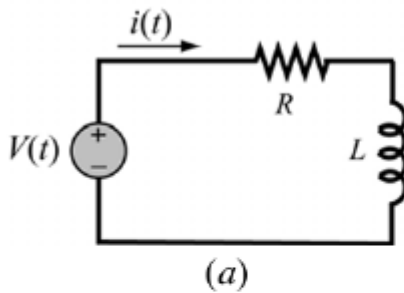
grid on; % Enable grid

```

- **OUTPUT:**



10. A resistor, $R = 4 \, \Omega$, and an inductor, $L = 1.3 \, \text{H}$, are connected in a circuit to a voltage source as shown in Figure (a) (an RL circuit). When the voltage



source applies a rectangular voltage pulse with an amplitude of $V = 12 \, \text{V}$ and a duration of $0.5 \, \text{s}$, as shown in Figure (b), the current $i(t)$ in the circuit as a function of time is given by:

$$i(t) = \frac{V}{R}(1 - e^{(-Rt)/L}) \quad \text{for } 0 \leq t \leq 0.5 \, \text{s}$$

$$i(t) = e^{-(Rt)/L} \frac{V}{R}(e^{(0.5R)/L} - 1) \quad \text{for } 0.5 \leq t \, \text{s}$$

Make a plot of the current as a function of time for $0 \leq t \leq 2 \, \text{s}$.

• **PROGRAM:**

% Define given parameters

$R = 4$; % Resistance

$L = 1.3$; % Inductance

$V = 12$; % Voltage

% Define time intervals

$t1 = \text{linspace}(0, 0.5, 500)$; % Time range for $0 \leq t \leq 0.5$

$t2 = \text{linspace}(0.5, 2, 500)$; % Time range for $0.5 < t \leq 2$

% Compute current for each interval

$i1 = (V / R) * (1 - \exp(-R * t1 / L))$; % For $0 \leq t \leq 0.5$

$i2 = \exp(-R * (t2 - 0.5) / L) * (V / R) * (\exp(R * 0.5 / L) - 1)$; % For $0.5 < t \leq 2$

% Combine time and current values

$t = [t1, t2]$;

$i = [i1, i2]$;

% Plot the current as a function of time

figure;

```

plot(t, i, 'b', 'LineWidth', 2);
xlabel('Time (s)');
ylabel('Current i(t) (A)');
title('Current i(t) vs. Time in an RL Circuit');
grid on;

```

- **OUTPUT:**

