**Experiment 1 (PLOTTING)**

% Create a vector 'x' ranging from 0 to 10 with a step size of 0.1

x = 0:0.1:10;

% Calculate the sine values of each element in vector 'x' and store it in 'y'

y = sin(x);

% Calculate the cosine values of each element in vector 'x' and store it in 'z'

z = cos(x);

% Create a 3-row, 1-column subplot layout and select the first subplot

subplot(3,1,1);

% Plot the sine function using the values in vectors 'x' and 'y'

plot(x, y);

% Add a grid to the current subplot

grid on;

% Select the second subplot for subsequent plotting

subplot(3,1,2);

% Plot the cosine function using the values in vectors 'x' and 'z'

plot(x, z);

% Add a grid to the current subplot

grid on;

% Keep the current plot active, allowing for additional plots

hold on;

% Select the third subplot for subsequent plotting

subplot(3,1,3);

% Create a stem plot of the cosine function in the third subplot

stem(x, z);

% Add a grid to the current subplot

grid on;

% Keep the current plot active

hold on;

% Add a red stem plot of the sine function to the third subplot

stem(x, y, 'r');

**Experiment 2 (Generating a Signal)**

% Generation of sinusoidal signals: x(t) = 2sin(2πt - π/2)

% Define the time vector 't' from -5 to 5 with a step size of 0.01

t = -5:0.01:5;

% Generate the sinusoidal sequence x(t) = 2sin(2πt - π/2)

x = 2 \* sin(2 \* pi \* t - pi/2);

% Plot the sinusoidal sequence

plot(t, x);

% Add a grid to the plot

grid on;

% Set the axis limits for better visualization

axis([-6 6 -3 3]);

% Label the y-axis as 'x(t)'

ylabel('x(t)');

% Label the x-axis as 'Time (sec)'

xlabel('Time (sec)');

% Add a title to the plot

title('Figure 2.1');

**Experiment 3 (Generation of discrete time signals)**

% Generation of discrete time signals

% Define the discrete time vector 'n' from -5 to 5

n = -5:5;

% Define the discrete signal sequence 'x'

x = [0 0 1 1 -1 0 2 -2 3 0 -1];

% Plot the discrete signal using stems

stem(n, x);

% Set the axis limits for better visualization

axis([-6 6 -3 3]);

% Label the x-axis as 'n'

xlabel('n');

% Label the y-axis as 'x[n]'

ylabel('x[n]');

% Add a title to the plot

title('Figure 2.2');

**Experiment 4 (Generation of random sequences)**

% Generation of random sequences

% Define the time vector 'n' from 0 to 10

n = 0:10;

% Generate a random sequence 'x' with values between 0 and 1

x = rand(1, length(n));

% Generate a random sequence 'y' with values from a standard normal distribution

y = randn(1, length(n));

% Plot the sequence 'x'

plot(n, x);

grid on;

% Keep the current plot active to add another plot

hold on;

% Plot the sequence 'y' in red ('r')

plot(n, y, 'r');

% Label the y-axis as 'x & y'

ylabel('x & y');

% Label the x-axis as 'n'

xlabel('n');

% Add a title to the plot

title('Figure 2.3');

**Experiment 5 (Generating a discrete periodic signal Signal)**

% Given sequence

n = 0:4;

x = [1 1 2 -1 0];

% Plot the original sequence x(n) in the first subplot

subplot(2,1,1);

stem(n, x);

grid on;

axis([0 4 -1 2]); % Set axis limits

xlabel('n');

ylabel('x(n)');

title('Figure 2.4(a)');

% Create a periodic extension of x(n) in the second subplot

xtilde = [x, x, x]; % Create a periodic extension

length\_xtilde = length(xtilde);

n\_new = 0:length\_xtilde-1;

% Plot the periodic sequence xtilde(n) in the second subplot in red

subplot(2,1,2);

stem(n\_new, xtilde, 'r');

grid on;

xlabel('n');

ylabel('periodic x(n)');

title('Figure 2.4(b)');

**Experiment 6 (Generating Square wave) using loop**

% Clear command window and workspace

clear;

clc;

% Input the value of odd 'n' from the user

n = input('Insert the value of odd n: ');

% Create a time vector 't' from 0 to 1 with a step size of 0.001

t = 0:0.001:1;

% Initialize the variable 'sum' to store the cumulative sum

sum = 0;

% Generate the signal by summing sine waves with odd harmonics

for f = 1:2:n

w = sin(2 \* pi \* f \* t);

sum = sum + w;

end

% Plot the generated signal in a single subplot

subplot(1,1,1);

% Plot the cumulative sum of sine waves

plot(t, sum);

% Add a grid to the plot

grid on;

% Provide a title for the plot

title('Sum of Sine Waves with Odd Harmonics');

% Label the x-axis as 'Time'

xlabel('Time');

% Label the y-axis as 'Amplitude'

ylabel('Amplitude');

**Experiment 7 (Generating Unit Step Discrete Time Signal)**

% Clear the command window, current variables, and close all open figures

clc;

clear all;

close all;

% Prompt the user to enter the range 'N'

N = input('Enter the range: ');

% Define the time vector 'n' from -N to N

n = -N:1:N;

% Create a unit step function 'y' (0 for n < 0, 1 for n >= 0)

y = [zeros(1, N), 1, ones(1, N)];

% Plot the unit step function using stems

stem(n, y);

% Set the axis limits for better visualization

axis([-(N+1) N+1 -0.5 1.5]); % [-x x -y y]

% Label the x-axis as 'Time'

xlabel('Time');

% Label the y-axis as 'Amplitude of Y'

ylabel('Amplitude of Y');

% Add a title to the plot

title('Generating Unit Step Function');

***Experiment 8 :*** Unit impulse signal

%Code of implementation of impulse signal in Matlab:

clc;

clear all;

m1=input('enter the value of x-axis in negative side:');

m2=input('enter the value of x-axis in positive side:');

n=m1:m2;

x=(n==0);%it works as if statement like n=-5:5( 0 0 0 0 0 1 0 0 0 0 0 0)

stem(n,x);

xlabel('n');

ylabel('amplitude');

title('Unit impulse signal');

or,  
  
clc;

clear all;

close all;

m1=input('enter the value of x-axis in negative side:');

m2=input('enter the value of x-axis in positive side:');

n=-m1:m2;

d=[zeros(1,m1) 1 zeros(1,m2)];

stem(n,d);

xlabel('n');

ylabel('amplitude');

title('unit impulse signal');

**Experiment 9** Generating and plotting ramp discrete time signal.

% Close all open figures and clear the command window

close all;

clc;

% Input the lower limit 'n1' from the user

n1 = input('Enter lower limit: ');

% Input the upper limit 'n2' from the user

n2 = input('Enter upper limit: ');

% Create a vector 'n' from 'n1' to 'n2' with a step of 1

n = n1:1:n2;

% Generate a ramp function 'x' (n for n >= 0, 0 for n < 0)

x = n .\* (n >= 0);

% Plot the ramp function using stems in blue ('b')

stem(n, x, 'b');

% Set the axis limits for better visualization

axis([n1-1 n2+1 -1 n2+1]); % [-x, x, -y, y]

% Label the x-axis as 'time'

xlabel('time');

% Label the y-axis as 'Amplitude of Y'

ylabel('Amplitude of Y');

% Add a title to the plot

title('Ramp Function');

**Experiment 10 (Time reversal using a discrete sinusoidal function [use of fliplr( ) and values of x-axis(angle) in radian)**

% Close all open figures and clear the command window

close all;

clc;

% Define the time vector t1 from 0 to 2\*pi with a step of 0.2

t1 = 0:0.2:2\*pi;

% Generate the original signal x1 using sine function

x1 = sin(t1);

% Generate the time-reversed signal x2 by flipping x1 and negating the time values

x2 = fliplr(x1);

t2 = -fliplr(t1);

% Plot the original signal in the first subplot

subplot(2,1,1);

stem(t1, x1, 'LineWidth', 2);

xlim([-10 10]); % Set x-axis limits

title('\bf\fontsize{25}Original Signal');

xlabel('\bf\fontsize{20}Samples');

ylabel('\bf\fontsize{20}Amplitude');

grid on;

ax = gca;

ax.XAxis.FontSize = 15;

ax.XAxis.FontWeight = 'bold';

ax.YAxis.FontSize = 15;

ax.YAxis.FontWeight = 'bold';

% Plot the time-reversed signal in the second subplot

subplot(2,1,2);

stem(t2, x2, 'LineWidth', 2);

xlim([-10 10]); % Set x-axis limits

title('\bf\fontsize{25}Time Reversed Signal');

xlabel('\bf\fontsize{20}Samples');

ylabel('\bf\fontsize{20}Amplitude');

grid on;

ax = gca;

ax.XAxis.FontSize = 15;

ax.XAxis.FontWeight = 'bold';

ax.YAxis.FontSize = 15;

ax.YAxis.FontWeight = 'bold';