1. I have the 8-digit ID number 21703381 which leads to following values for frequency, amplitude, and phase of the three signals which chosen according to the project description and within principal interval.

   - Frequency: $\omega_0 = 33$ rad/s
   - Amplitudes: $A_1 = 3, A_2 = 8, A_3 = 1$
   - Phase values (degrees): $\phi_1 = 33°, \phi_2 = -22°, \phi_3 = 21°$
   - Phase values (radians): $\phi_1 =$ 0.576 rad , $\phi_2 =$ -0.384 rad, $\phi_3 =$ 0.367 rad

2. In order to allow user to type the values I used the `input()` command.

   Following is the MATLAB section to take user input for Part (i).

```
%% Part (i)

% read user input
omega_0 = input('Enter the omega_0 value: ');
A1 = input('Enter the A1 value: ');
A2 = input('Enter the A2 value: ');
A3 = input('Enter the A3 value: ');
phi_1 = input('Enter the phi_1 value (deg): ');
phi_2 = input('Enter the phi_2 value (deg): ');
phi_3 = input('Enter the phi_3 value (deg): ');
```

3. Following is the MATLAB section to perform calculations for finding the amplitude A and phase ϕ of the phasor addition. Steps well documented in the code section. First the polar representation obtained from the Amplitude and phase the user provides, then the polar representation converted back to the Cartesian representation to easily add up the real and imaginary parts. Final step was converting back to the polar representation which yield the signal with Amplitude: 10.9134, phase: -5.2815 for my university id.

```
%% Part (ii)

% (a) represent signals as phasors
X_1p = A1 * exp(1i*degree_to_radian(phi_1));
X_2p = A2 * exp(1i*degree_to_radian(phi_2));
X_3p = A3 * exp(1i*degree_to_radian(phi_3));

% (b) convert back to the rectangular form
X_1r = to_cartesian(A1, phi_1);
X_2r = to_cartesian(A2, phi_2);
X_3r = to_cartesian(A3, phi_3);

% (c) add the real and imaginary parts
X = X_1r + X_2r + X_3r;

% (d) convert back to the polar
X_polar = to_polar(X);
```

```

X_res = X_polar(1,1) * exp(1i*degree_to_radian(X_polar(1,2)));


% Report A and phi values
fprintf('A: %.4f, phi: %.4f \n', X_polar(1,1), X_polar(1,2));
```

4. Resulting sinusoidal: $x(t) = 10.91cos(33.00t - 0.09)$

```
%% Part (iii)

% resulting sinusoidal
fprintf('Resulting sinusoidal: ');
print_signal(X_polar(1,1), omega_0, degree_to_radian(X_polar(1,2)));
```

5. Following is the code section where I represent the phasor addition geometrically. It plots phasors of the three sinusoidal signals together with the resulting phasor on the complex z plane with real horizontal and imaginary vertical axes.

```
%% Part (iv)

% define vectors
res_phasor = (X_res);
phasors = [X_1p; X_2p; X_3p];

% Plot
figure;
plot(real(phasors), imag(phasors), '->','LineWidth',1)
hold on
plot(real(res_phasor), imag(res_phasor), '->','LineWidth',3.5)
hold off
xlim([-30 30]);
ylim([-30 30]);
title('Part (iv): phasor addition plot');
xlabel('Real part');
ylabel('Imaginary part');
grid on;
```

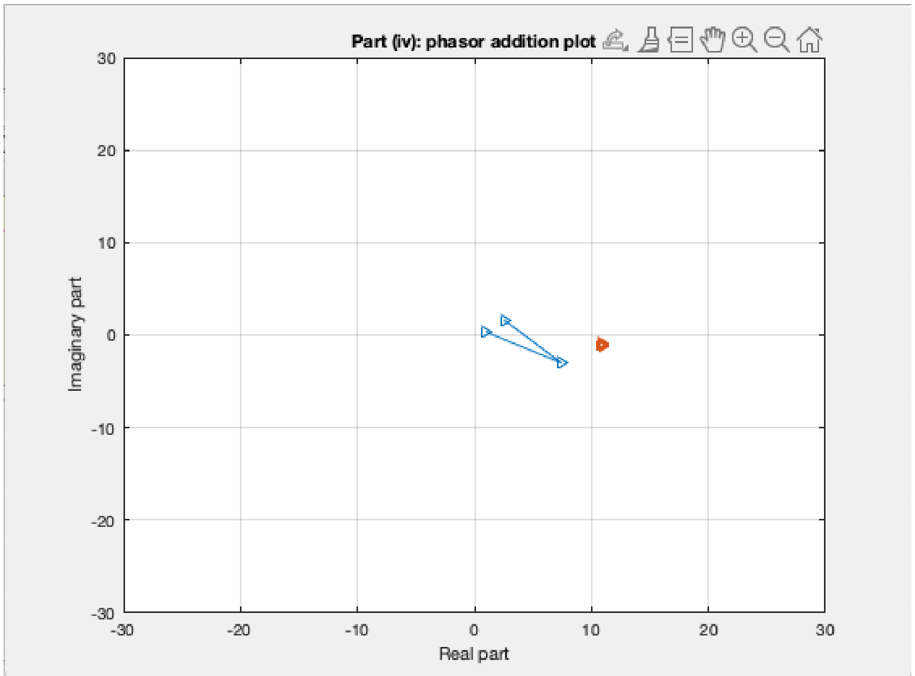

Figure 1: Geometric representation of the phasor addition

6. Following is the code section where I represent the phasor addition geometrically by adding up phasors end to end. To achieve adding vectors end to end I used the step-wise vectorsum via using the MATLAB command. `cumsum()`.

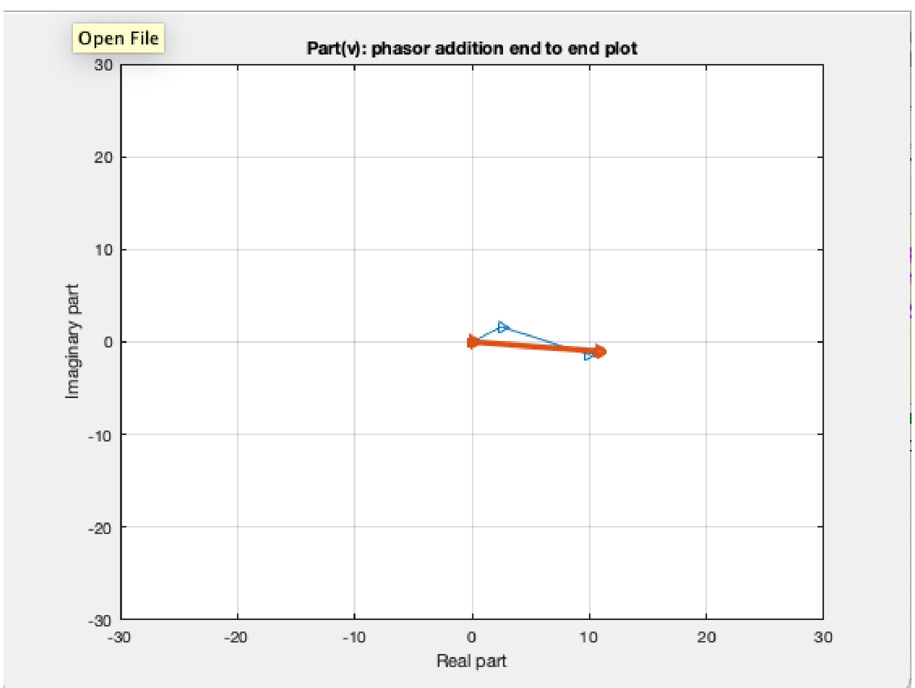

Figure 2: Adding three phasors end to end to demonstrate phasor addition

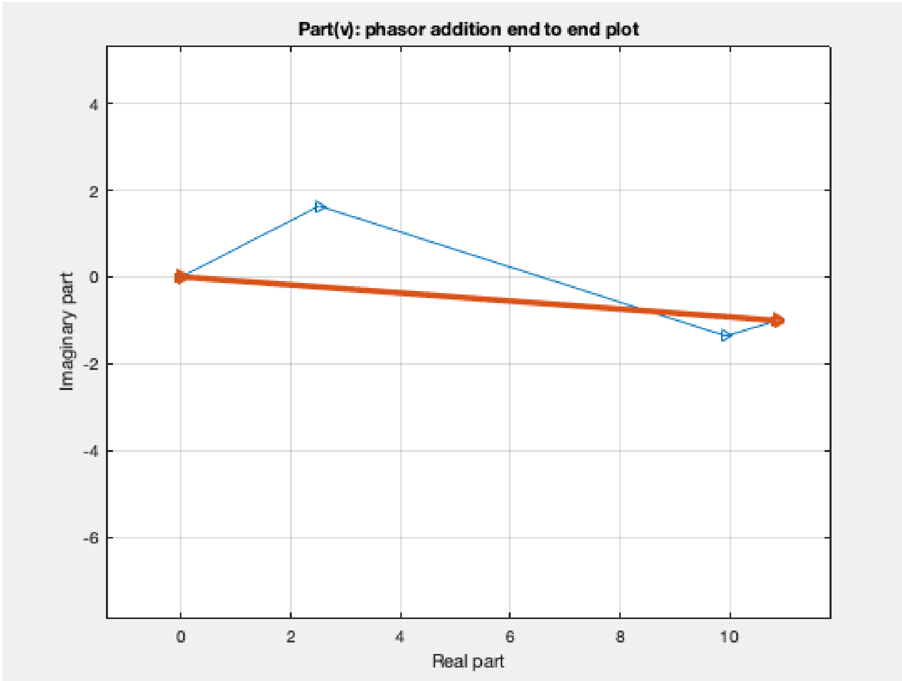

Figure 3: Figure 2 scaled.

7. Appendix

The MATLAB codes

```
%% ASSIGNMENT 1
% Author: Zeynep CANKARA
% Course: EEE391 Signals and Systems

%% Helpers
% degree to radian
degree_to_radian = @(deg) (deg * pi / 180);
% radian to degree
radian_to_degree = @(rad) (rad * 180 / pi);
% to cartesian
to_cartesian = @(len, deg) len .* exp(1i * degree_to_radian(deg));
% to polar
to_polar = @(num) [abs(num) radian_to_degree(angle(num))];


%% Part (i)

% read user input
omega_0 = input('Enter the omegao value: ');
A1 = input('Enter the A1 value: ');
A2 = input('Enter the A2 value: ');
A3 = input('Enter the A3 value: ');
```

```
phi_1 = input('Enter the phi1 value (deg): ');
phi_2 = input('Enter the phi2 value (deg): ');
phi_3 = input('Enter the phi3 value (deg): ');

%% Part (ii)

% (a) represent signals as phasors
X_1p = A1 * exp(1i*degree_to_radian(phi_1));
X_2p = A2 * exp(1i*degree_to_radian(phi_2));
X_3p = A3 * exp(1i*degree_to_radian(phi_3));

% (b) convert back to the rectangular form
X_1r = to_cartesian(A1, phi_1);
X_2r = to_cartesian(A2, phi_2);
X_3r = to_cartesian(A3, phi_3);

% (c) add the real and imaginary parts
X = X_1r + X_2r + X_3r;

% (d) convert back to the polar
X_polar = to_polar(X);

X_res = X_polar(1,1) * exp(1i*degree_to_radian(X_polar(1,2)));


% Report A and phi values
fprintf('A: %.4f, phi: %.4f \n', X_polar(1,1), X_polar(1,2));

%% Part (iii)

% resulting sinusoidal
fprintf('Resulting sinusoidal: ');
print_signal(X_polar(1,1), omega_0, degree_to_radian(X_polar(1,2)));

%% Part (iv)

% define vectors
res_phasor = (X_res);
phasors = [X_1p; X_2p; X_3p];

% Plot
figure;
plot(real(phasors), imag(phasors), '->','LineWidth',1)
hold on
plot(real(res_phasor), imag(res_phasor), '->','LineWidth',3.5)
hold off
xlim([-30 30]);
ylim([-30 30]);
```

```
title('Part (iv): phasor addition plot');
xlabel('Real part');
ylabel('Imaginary part');
grid on;

%% Part (v)

% define vectors
phasors = cumsum([X_1p; X_2p; X_3p]);
res_phasor = [0; res_phasor];
phasors = [0; phasors]; % concat with origin as starting point

% Plot
figure;
plot(real(phasors), imag(phasors), '->','LineWidth',1)
hold on
plot(real(res_phasor), imag(res_phasor), '->','LineWidth',3.5)
hold off
xlim([-30 30]);
ylim([-30 30]);
title('Part(v): phasor addition end to end plot');
xlabel('Real part');
ylabel('Imaginary part');
grid on;


%% Function declerations
function print_signal(A, omega_0, phi)
    if phi < 0
        fprintf('x(t) = %.2f cos(%.2ft - %.2f)\n', A, omega_0, abs(phi));
    else
        fprintf('x(t) = %.2f cos(%.2ft + %.2f)\n', A, omega_0, phi);
    end
end
```

The output dumps, contains screen shot of the workspace and command window

Workspace

Name	Value
A1	3
A2	8
A3	1
degree_to_radi...	@(deg)(deg*pi/180)
omega_0	33
phasors	[0.0000 + 0.0000...
phi_1	33
phi_2	-22
phi_3	21
radian_to_degr...	@(rad)(rad*180/pi)
res_phasor	[0.0000 + 0.0000...
to_cartesian	@(len,deg)len.*ex...
to_polar	@(num)[abs(num)...
X	10.8671 - 1.0046i
X_1p	2.5160 + 1.6339i
X_1r	2.5160 + 1.6339i
X_2p	7.4175 - 2.9969i
X_2r	7.4175 - 2.9969i
X_3p	0.9336 + 0.3584i
X_3r	0.9336 + 0.3584i
X_polar	[10.9134,-5.2815]
X_res	10.8671 - 1.0046i

Figure 4: Workspace

Command Window

```
>> eee391_hw1
Enter the ωo value: 33
Enter the A1 value: 3
Enter the A2 value: 8
Enter the A3 value: 1
Enter the φ1 value (deg): 33
Enter the φ2 value (deg): -22
Enter the φ3 value (deg): 21
A: 10.9134, φ: -5.2815
Resulting sinusoidal: x(t) = 10.91 cos(33.00t - 0.09)
>>
```

Figure 5: Command Window