(1)Write a Program to Sampling of a Sinusoidal Signal and Reconstruction of Analog Signal.

#Python:

```
import numpy as np
import matplotlib.pyplot as plt

# Define the parameters of the signal
f = 10  # Frequency of the sinusoid (in Hz)
fs = 200  # Sampling rate (in Hz)
t = np.arange(0, 1, 1 / fs)  # Time vector
x = np.sin(2 * np.pi * f * t)  # Generate the sinusoidal signal

# Plot the original signal
plt.subplot(3, 1, 1)
plt.plot(t, x)
plt.xlabel('Time (s)')
plt.ylabel('Amplitude')
plt.title('Original Signal')

# Sample the signal
Ts = 1 / fs  # Sampling interval (in seconds)
n = np.arange(0, 1 + Ts, Ts)  # Sampling instants
xn = np.sin(2 * np.pi * f * n)  # Sampled signal

# Plot the sampled signal
plt.subplot(3, 1, 2)
plt.stem(n, xn)
plt.xlabel('Time (s)')
plt.ylabel('Amplitude')
plt.title('Sampled Signal')

# Reconstruct the analog signal using ideal reconstruction
xr = np.zeros_like(t)  # Initialize the reconstructed signal
for i in range(len(n)):
    xr += xn[i] * np.sinc((t - (i - 1) * Ts) / Ts)

# Plot the reconstructed signal
plt.subplot(3, 1, 3)
plt.plot(t, xr)
plt.xlabel('Time (s)')
plt.ylabel('Amplitude')
plt.title('Reconstructed Signal')
plt.show()
```

(2)Write a Program to Implement Z-transform of a Discrete Time Function, Inverse Z-transform, Pole-zeros diagram and Root of a system.

#MatLab:

```
syms z n
a=1/16^n;                    %x(n) = [1/16^n]u(n)
ZTrans=ztrans(a);         %Z transform
disp(ZTrans);
InvrZ=iztrans(ZTrans);      %InverseZtransform
disp(InvrZ);

B=[0 1 1];
A=[1 -2 3];
pl = roots(A); % To display pole value
```

```
disp(pl);
zr= roots(B); % To display zero value
disp(zr);
figure(1);
zplane(B,A); % Compute and display pole-zero diagram
```

(3) **Write a Program to Implement The Discrete Fourier Transform (DFT) and Fast Fourier Transform (FFT).**

#Python:

DFT

```
import numpy as np
import matplotlib.pyplot as plt

n = np.arange(-1, 4)
x = np.arange(1, 6)
k = np.arange(501)
w = (np.pi / 500) * k

X = np.sum(x[:, np.newaxis] * np.exp(-1j * np.pi / 500 * n[:, np.newaxis] * k), axis=0)

magX = np.abs(X)
angX = np.angle(X)
realX = np.real(X)
imagX = np.imag(X)
plt.figure(figsize=(12, 8))

plt.subplot(2, 2, 1)
plt.plot(k / 500, magX)
plt.grid()
plt.xlabel('Frequency in pi units')
plt.title('Magnitude part')

plt.subplot(2, 2, 2)
plt.plot(k / 500, angX / np.pi)
plt.grid()
plt.xlabel('Frequency in pi units')
plt.title('Angle part')

plt.subplot(2, 2, 3)
plt.plot(k / 500, realX)
plt.grid()
plt.xlabel('Frequency in pi units')
plt.title('Real part')

plt.subplot(2, 2, 4)
plt.plot(k / 500, imagX)
plt.grid()
plt.xlabel('Frequency in pi units')
plt.title('Imaginary part')

plt.tight_layout()
plt.show()
```

FFT

```
import numpy as np
import matplotlib.pyplot as plt

N = 256
T = 1 / 128
k = np.arange(N)
time = k * T
f = 0.25 + 2 * np.sin(2 * np.pi * 5 * k * T) + np.sin(2 * np.pi * 12.5 * k * T) + 1.5 * np.sin(2 * np.pi * 20 * k * T) + 0.5 * np.sin(2 * np.pi * 35 * k * T)

plt.subplot(2, 1, 1)
plt.plot(time, f)
plt.title('Signal sampled at 128Hz')

F = np.fft.fft(f)
magF = np.abs(np.concatenate(([F[0] / N], F[1:N // 2] / (N / 2))))
hertz = k[:N // 2] * (1 / (N * T))

plt.subplot(2, 1, 2)
plt.stem(hertz, magF)
plt.title('Frequency Components')

plt.tight_layout()
plt.show()
```

(4) **Write a Program to Designing Finite Impulse Response (FIR) Filters and Infinite Impulse Response (IIR) Filters.**

#MatLab:

FIR

Low pass Filter:

```
%Suppose out target is to pass all frequencies below 1200 Hz
fs=8000; % sampling frequency n=50; % order of the filter w=1200/ (fs/2);
b=fir1(n,w,'low'); % Zeros of the filter
freqz(b,1,128,8000); % Magnitude and Phase Plot of the filter figure(2)
[h,w]=freqz(b,1,128,8000);
plot(w,abs(h)); % Normalized Magnitude Plot
grid figure(3) zplane(b,1);
```

High Pass Filter:

```
%Now our target is to pass all frequencies above 1200 Hz fs=8000;
n=50;
w=1200/ (fs/2); b=fir1(n,w,'high');
freqz(b,1,128,8000); figure(2) [h,w]=freqz(b,1,128,8000);
plot(w,abs(h)); % Normalized Magnitude Plot
grid figure(3) zplane(b,1);
```

Band Pass Filter:
fs=8000; n=40;
b=fir1(n,[1200/4000 1800/4000],'bandpass'); freqz(b,1,128,8000)
figure(2) [h,w]=freqz(b,1,128,8000);
plot(w,abs(h)); % Normalized Magnitude Plot
grid figure(3) zplane(b,1);

Band Stop Filter:
fs=8000;
15
n=40;
b=fir1(n,[1200/4000 2800/4000],'stop');
freqz(b,1,128,8000) figure(2) [h,w]=freqz(b,1,128,8000);
plot(w,abs(h)); % Normalized Magnitude Plot
grid figure(3) zplane(b,1);
Notch Filter:
fs=8000; n=40;
b=fir1(n,[1500/4000 1550/4000],'stop'); freqz(b,1,128,8000)
figure(2) [h,w]=freqz(b,1,128,8000);
plot(w,abs(h)); % Normalized Magnitude Plot
grid figure(3) zplane(b,1);

Multiband Filter: n=50;
w=[0.2 0.4 0.6]; b=fir1(n,w); freqz(b,1,128,8000) figure(2) [h,w]=freqz(b,1,128,8000);
plot(w,abs(h)); % Normalized Magnitude Plot
grid figure(3) zplane(b,1);

IIR

Low Pass Filter:
%Suppose our target is to design a filter to pass all frequencies below 1200 Hz with pass band %ripples = 1 dB and minimum stop band attenuation of 50 dB at 1500 Hz. The sampling %frequency for the filter is 8000 Hz;

fs=8000;
[n,w]=buttord(1200/4000,1500/4000,1,50); % finding the order of the filter
[b,a]=butter(n,w); % finding zeros and poles for filter
figure(1) freqz(b,a,512,8000); figure(2)
[h,q] = freqz(b,a,512,8000);
plot(q,abs(h)); % Normalized Magnitude plot grid
figure(3) f=1200:2:1500;
freqz(b,a,f,8000) % plotting the Transition band figure(4)
zplane(b,a) % pole zero constellation diagram

High Pass Filter:
%We will consider same filter but our target now is to pass all frequencies above 1200 Hz

```
[n,w]=buttord(1200/5000,1500/5000,1,50);
[b,a]=butter(n,w,'high'); figure(1) freqz(b,a,512,10000); figure(2)
[h,q] = freqz(b,a,512,8000);
plot(q,abs(h)); % Normalized Magnitude plot grid
figure(3) f=1200:2:1500; freqz(b,a,f,10000) figure(4) zplane(b,a);
```

Band Pass Filter:

```
%with pass band ripples = 1 dB and minimum stop band attenuation of 50 dB. The
%sampling frequency for the filter is 8000 Hz;
[n,w]=buttord([1200/4000,2800/4000],[400/4000, 3200/4000],1,50);
[b,a]=butter(n,w,'bandpass'); figure(1) freqz(b,a,128,8000) figure(2)
[h,w]=freqz(b,a,128,8000); plot(w,abs(h))
grid figure(3) f=600:2:1200;
freqz(b,a,f,8000); % Transition Band figure(4)
f=2800:2:3200;
freqz(b,a,f,8000); % Transition Band figure(5)
zplane(b,a);
```

Band Stop Filter:

```
[n,w]=buttord([1200/4000,2800/4000],[400/4000, 3200/4000],1,50); [b,a]=butter(n,w,'stop');
figure(1) freqz(b,a,128,8000) [h,w]=freqz(b,a,128,8000);
figure(2) plot(w,abs(h));
grid figure(3) f=600:2:1200;
freqz(b,a,f,8000); % Transition Band figure(4)
f=2800:2:3200;
freqz(b,a,f,8000); % Transition Band figure(5)
zplane(b,a);
```