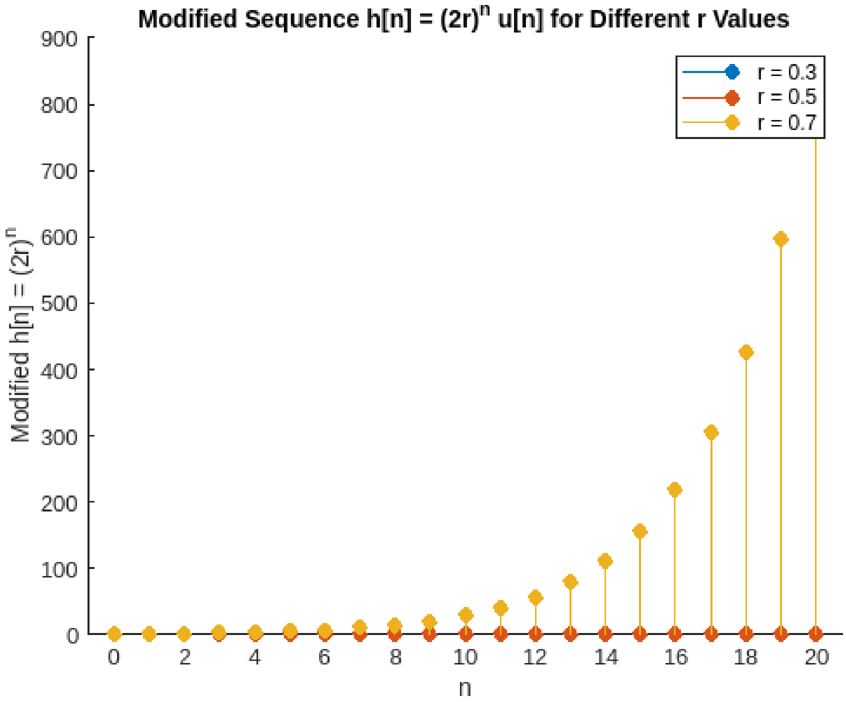
DSP lab 11
Name: Basil khowaja bk08432

Task 1)

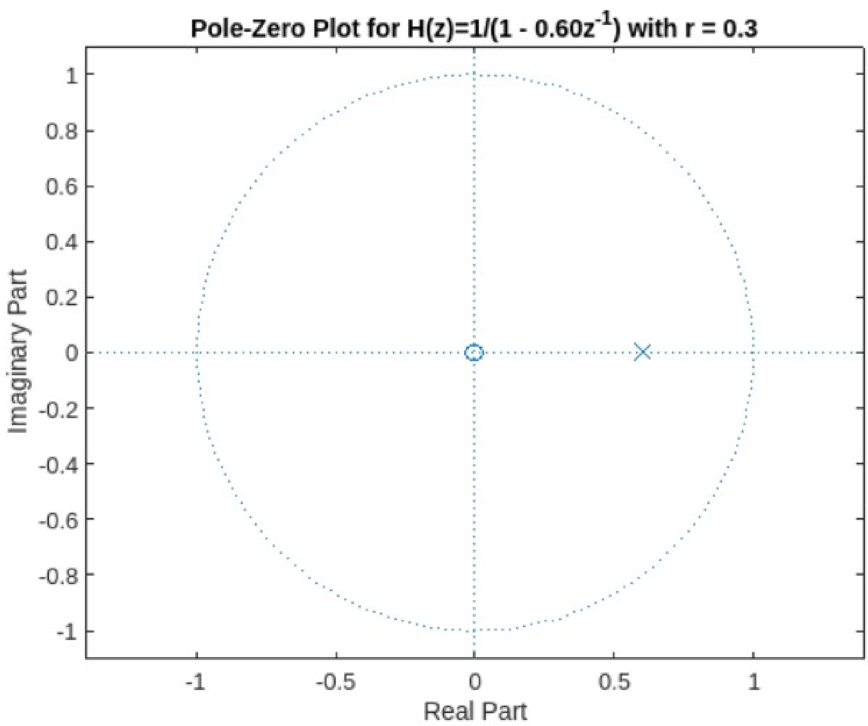
code:

```
clear; close all; clc;
n = 0:20;
h = 2.^n;
r_values = [0.3, 0.5, 0.7];
figure;
hold on;
for i = 1:length(r_values)
    r = r_values(i);
    h_mod = (2 * r).^n;
    stem(n, h_mod, 'filled', 'DisplayName', sprintf('r = %.1f', r));
end
xlabel('n');
ylabel('Modified h[n] = (2r)^n');
title('Modified Sequence h[n] = (2r)^n u[n] for Different r Values');
legend;
hold off;

r_chosen = 0.3;
b = 1;
a = [1, -2 * r_chosen];
figure;
zplane(b, a);
title(sprintf('Pole-Zero Plot for H(z)=1/(1 - %.2fz^{-1}) with r = %.1f', 2*r_chosen,
r_chosen));

[residues, poles, direct_terms] = residuez(b, a);
disp('Partial Fraction Expansion using residuez:');
disp('Residues:');
disp(residues);
disp('Poles:');
disp(poles);
disp('Direct Terms:');
disp(direct_terms);

if (2 * r_chosen < 1)
    stability = 'Stable';
else
    stability = 'Unstable';
end
fprintf('For r = %.1f, the pole is at z = %.2f.\n', r_chosen, 2*r_chosen);
fprintf('The system is %s because |2r| = %.2f < 1.\n', stability, 2*r_chosen);
fprintf('The Region of Convergence (ROC) is |z| > %.2f.\n', 2*r_chosen);
```

outputs:



Modified Sequence h[n] = (2r)^n u[n] for Different r Values
r = 0.3
r = 0.5
r = 0.7
Modified h[n] = (2r)^n
n
0
100
200
300
400
500
600
700
800
900
0
2
4
6
8
10
12
14
16
18
20



Pole-Zero Plot for H(z)=1/(1 - 0.60z^-1) with r = 0.3
Imaginary Part
Real Part
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.5
0
0.5
1

```
Partial Fraction Expansion using residuez:
Residues:
     1
Poles:
    0.6000
Direct Terms:
For r = 0.3, the pole is at z = 0.60.
The system is Stable because |2r| = 0.60 < 1.
The Region of Convergence (ROC) is |z| > 0.60.
```

```
H(z) = Numerator/Denominator with z^-1 polynomials:
Numerator: b = [1]
Denominator: a = [1, -0.60]
Which corresponds to: 1 / (1 - 0.60 z^-1)
```

Observation:

When we pre-multiply the sequence 2n by rn, we get the sequence (2r)n. For $r=0.3$, the value 2r equals 0.6, which is less than 1, so the sequence gets smaller over time (it decays) and its total sum is finite. For $r=0.5$, 2r equals 1, so the sequence stays constant, and for $r=0.7$, 2r equals 1.4, causing the sequence to grow quickly (diverge). Focusing on $r=0.3$, we form the Z-transform $H(z)=1/1-0.6z-1$, which has a single pole at 0.6. Since 0.6 is inside the unit circle (meaning its magnitude is less than 1), the system is stable. The partial fraction expansion confirms this by showing one residue of 1, one pole at 0.6, and no extra terms. Essentially, pre-multiplying by rn (with $r<0.5$) makes the sequence decay, so the Z-transform is well defined and the system is stable.

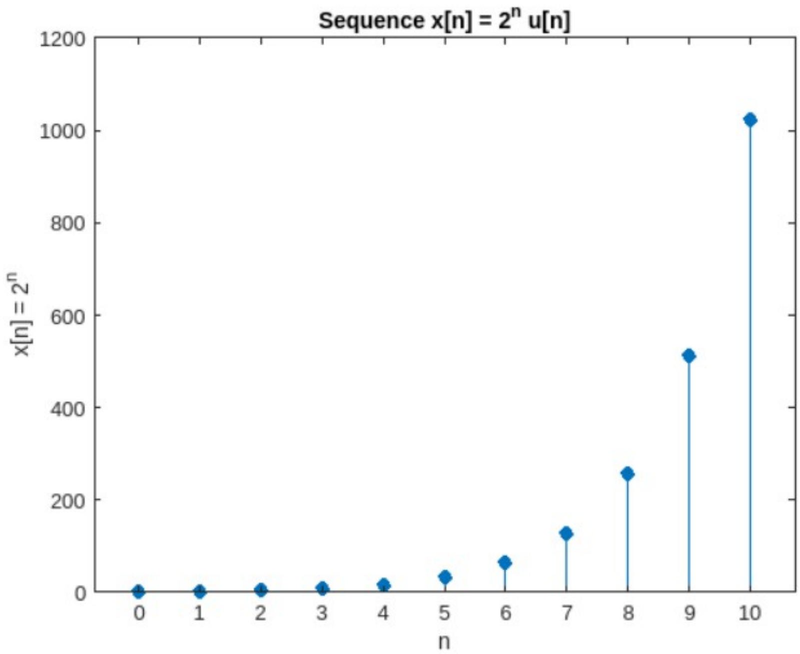
Task 2)

code:

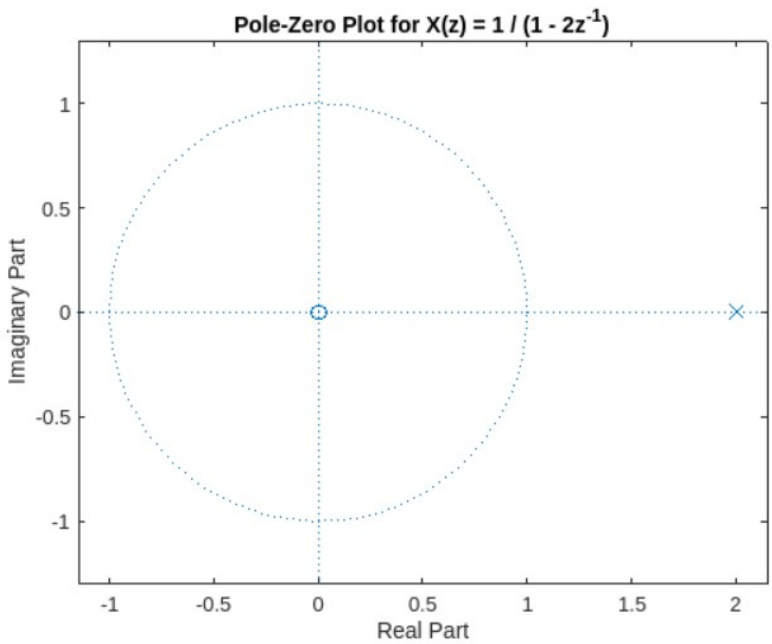
```
%task2:
clear; close all; clc;

n = 0:10;
x = 2.^n;

figure;
stem(n, x, 'filled');
xlabel('n');
ylabel('x[n] = 2^n');
title('Sequence x[n] = 2^n u[n]');

%theory: X(z) = 1 / (1 - 2*z^(-1)),  ROC: |z| > 2
%we represent X(z) in terms of z^(-1) as:
b = 1;          %num coff
a = [1, -2];     %denom coff

[residues, poles, direct_terms] = residuez(b, a);

disp('--- Partial Fraction Expansion using residuez ---');
disp('Residues:');
disp(residues);
disp('Poles:');
disp(poles);
disp('Direct Terms:');
disp(direct_terms);

figure;
zplane(b, a);
title('Pole-Zero Plot for X(z) = 1 / (1 - 2z^{-1})');

%the pole is at z=2.
%Because this is a right-sided sequence the ROC is |z| > 2.
%this ROC does NOT come inside the unit circle (|z|=1) so the system is not stable.

fprintf('\nAnalysis:\n');
fprintf('The pole is at z = 2.\n');
fprintf('For a right-sided sequence, the ROC is |z| > 2.\n');
fprintf('Since the unit circle (|z|=1) is NOT inside the ROC, the system is not stable.\n');
```

outputs:

```
Analysis:
The pole is at z = 2.
For a right-sided sequence, the ROC is |z| > 2.
Since the unit circle (|z|=1) is NOT inside the ROC, the system is not stable.
```




```
--- Partial Fraction Expansion using residuez ---
Residues:
     1
Poles:
     2
Direct Terms:
```



Observation:

The sequence x[n]=2nu[n] grows very rapidly, doubling its value at each step. Its Z-transform is given by X(z)=1/(1−2z−1). This expression has one pole at z=2. Since the sequence is right-sided (defined for n≥0), its region of convergence (ROC) must be outside this pole, meaning ∣z∣>2. However, the unit circle ∣z∣=1, which determines system stability, is not included within this ROC. Because the ROC doesn't include the unit circle, the system is unstable. The partial fraction expansion supports this observation, confirming one pole at z=2 and a residue of 1, clearly explaining the instability and rapid growth of the sequence.

Task 3)

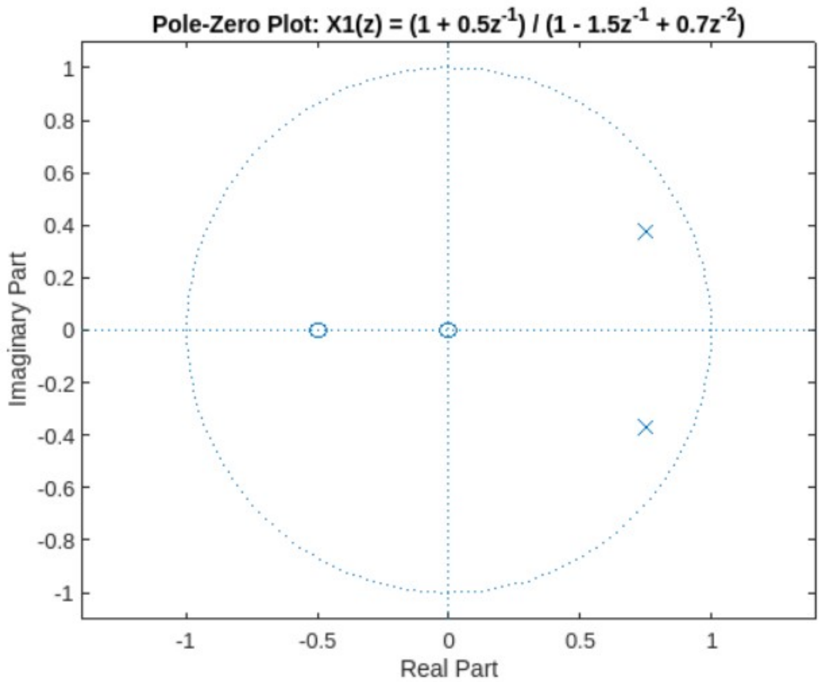
code:

```
%task3:
clear; close all; clc;
%X1(z) = (1 + 0.5 z^-1) / (1 - 1.5 z^-1 + 0.7 z^-2)
b1 = [1, 0.5];
a1 = [1, -1.5, 0.7];

[r1, p1, d1] = residuez(b1, a1);

fprintf('=== CASE 1 ===\n');
fprintf('X1(z) = (1 + 0.5 z^-1) / (1 - 1.5 z^-1 + 0.7 z^-2)\n');
disp('Partial Fraction Expansion:');
disp('Residues:'), disp(r1);
disp('Poles:'), disp(p1);
disp('Direct Terms:'), disp(d1);

figure;
zplane(b1, a1);
title('Pole-Zero Plot: X1(z) = (1 + 0.5z^{-1}) / (1 - 1.5z^{-1} + 0.7z^{-2})');

% X2(z) = (2 + 3 z^-1) / (1 - 0.8 z^-1 - 0.2 z^-2)
b2 = [2, 3];
a2 = [1, -0.8, -0.2];

[r2, p2, d2] = residuez(b2, a2);

fprintf('\n=== CASE 2 ===\n');
fprintf('X2(z) = (2 + 3 z^-1) / (1 - 0.8 z^-1 - 0.2 z^-2)\n');
disp('Partial Fraction Expansion:');
disp('Residues:'), disp(r2);
disp('Poles:'), disp(p2);
disp('Direct Terms:'), disp(d2);

figure;
zplane(b2, a2);
title('Pole-Zero Plot: X2(z) = (2 + 3z^{-1}) / (1 - 0.8z^{-1} - 0.2z^{-2})');

%%X3(z) = (4 - z^-1 + 0.5 z^-2) / (1 + 0.3 z^-1 - 0.6 z^-2)
b3 = [4, -1, 0.5];
a3 = [1, 0.3, -0.6];

[r3, p3, d3] = residuez(b3, a3);
```

```
fprintf('\n=== CASE 3 ===\n');
fprintf('X3(z) = (4 - z^-1 + 0.5 z^-2) / (1 + 0.3 z^-1 - 0.6 z^-2)\n');
disp('Partial Fraction Expansion:');
disp('Residues:'), disp(r3);
disp('Poles:'), disp(p3);
disp('Direct Terms:'), disp(d3);

figure;
zplane(b3, a3);
title('Pole-Zero Plot: X3(z) = (4 - z^{-1} + 0.5z^{-2}) / (1 + 0.3z^{-1} - 0.6z^{-2})');
```

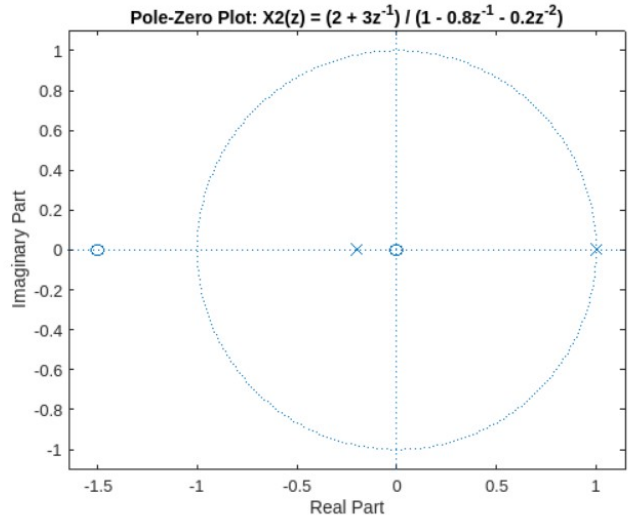
Outputs:
For case 1:

```
Partial Fraction Expansion:
Residues:
   0.5000 - 1.6855i
   0.5000 + 1.6855i
Poles:
   0.7500 + 0.3708i
   0.7500 - 0.3708i
Direct Terms:
```



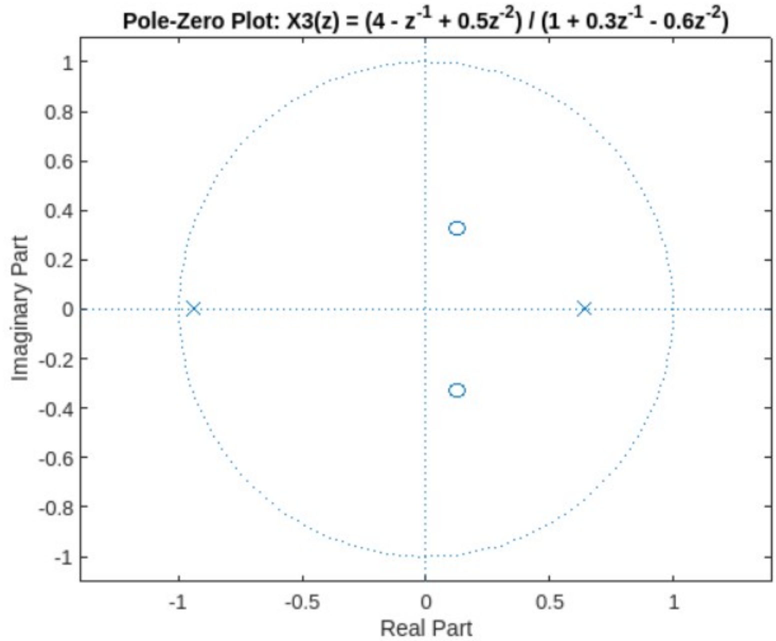
for case 2:

```
Partial Fraction Expansion:
Residues:
    4.1667
   -2.1667
Poles:
    1.0000
   -0.2000
Direct Terms:
```



for case 3:

```
Residues:
    3.3514
    1.4819
Poles:
   -0.9390
    0.6390
Direct Terms:
   -0.8333
```



Observation:

we analyze three different rational functions using partial fraction expansion and pole-zero plots. The first function X1(z)=(1+0.5z−1)/(1−1.5z−1+0.7z−2) has two complex poles at 0.75±0.3708i, located inside the unit circle. Because poles are inside the unit circle, this system is stable. It means the impulse response of the system decays over time, resulting in a converging output.

The second function X2(z)=(2+3z−1)/(1−0.8z−1−0.2z−2) has poles at z=1 and z=−0.2. Since one pole is exactly on the unit circle (∣z∣=1), the system is marginally stable or borderline unstable. The impulse response for this system does not decay but maintains constant amplitude or oscillations, resulting in neither clear convergence nor divergence.

The third function X3(z)=(4−z−1+0.5z−2)/(1+0.3z−1−0.6z−2) has two poles at z=−0.939 and z=0.639, both within the unit circle. Thus, the third system is stable as well, meaning its impulse response also converges and gradually decreases in amplitude.

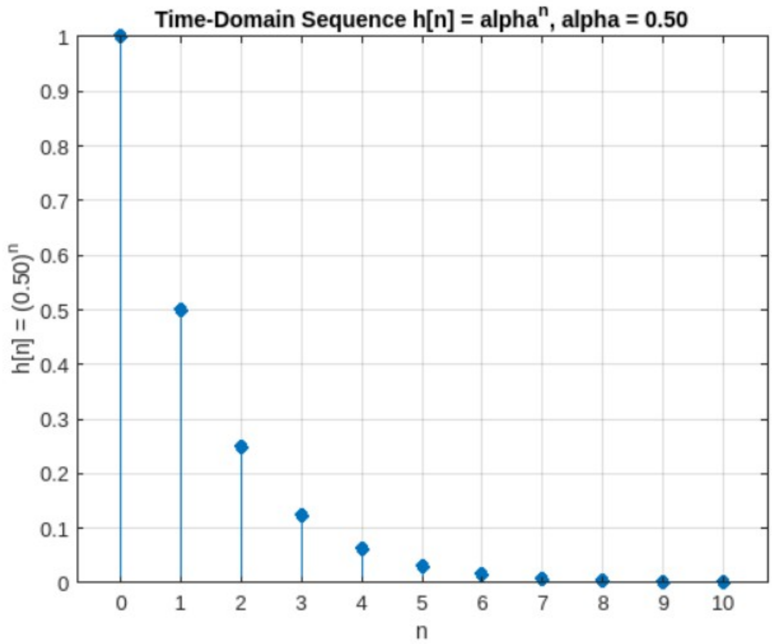
Task 4)

code:

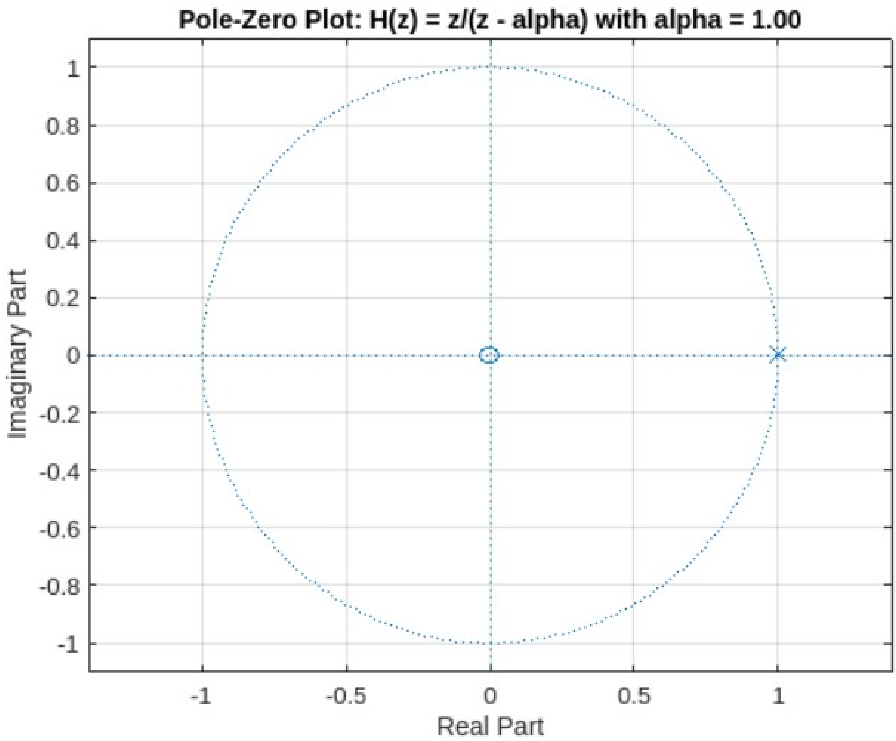
```
%task 4
clear; close all; clc;
alpha_values = [0.5, 1, 2, -0.5, -1, -2];
n = 0:10;
classificationTable = cell(length(alpha_values), 6);

for i = 1:length(alpha_values)
    alpha = alpha_values(i);
    b = 1;              %num
    a = [1, -alpha];     %denom

    figure;
    zplane(b, a);
    title(sprintf('Pole-Zero Plot: H(z) = z/(z - alpha) with alpha = %.2f', alpha));
    grid on;

    h = alpha.^n;
    figure;
    stem(n, h, 'filled');
    xlabel('n');
    ylabel(sprintf('h[n] = (%.2f)^n', alpha));
    title(sprintf('Time-Domain Sequence h[n] = alpha^n, alpha = %.2f', alpha));
    grid on;
    %Monotonic?
    %    => "Yes" if alpha > 0 (no sign changes), "No" otherwise
    if alpha > 0
        monotonic = 'Yes';
    else
        monotonic = 'No';
    end
    %Alternating?
    %    => "Yes" if alpha < 0 (sign alternates), "No" otherwise
    if alpha < 0
```

```
        alternating = 'Yes';
    else
        alternating = 'No';
    end
    %Convergent?
    %    => "Yes" if |alpha| < 1, "No" otherwise
    if abs(alpha) < 1
        convergent = 'Yes';
    else
        convergent = 'No';
    end
    %Constant?
    %    => "Yes" if alpha == 1, "No" otherwise
    if alpha == 1
        constant = 'Yes';
    else
        constant = 'No';
    end
    %Divergent?
    %    => "Yes" if |alpha| > 1 or alpha == -1, "No" otherwise
    %    (alpha = -1 oscillates between +1/-1, not convergent => "Yes" for divergent)
    if (abs(alpha) > 1) || (alpha == -1)
        divergent = 'Yes';
    else
        divergent = 'No';
    end

    classificationTable{i,1} = alpha;         % alpha value
    classificationTable{i,2} = monotonic;     % "Yes"/"No"
    classificationTable{i,3} = alternating;   % "Yes"/"No"
    classificationTable{i,4} = convergent;    % "Yes"/"No"
    classificationTable{i,5} = constant;      % "Yes"/"No"
    classificationTable{i,6} = divergent;     % "Yes"/"No"
end

disp('====================================================
=========');
disp('          Classification for H(z) = z/(z - alpha)');
disp('====================================================
=========');
disp(' alpha   Monotonic  Alternating  Convergent  Constant  Divergent');
disp('-----------------------------------------------------------');

for i = 1:length(alpha_values)
    alpha  = classificationTable{i,1};
    mono   = classificationTable{i,2};
    alt    = classificationTable{i,3};
    conv   = classificationTable{i,4};
    cst    = classificationTable{i,5};
    divg   = classificationTable{i,6};
    fprintf(' %5.2f    %7s     %7s       %7s     %7s     %7s\n', ...
            alpha, mono, alt, conv, cst, divg);
end

disp('====================================================
=========');
```

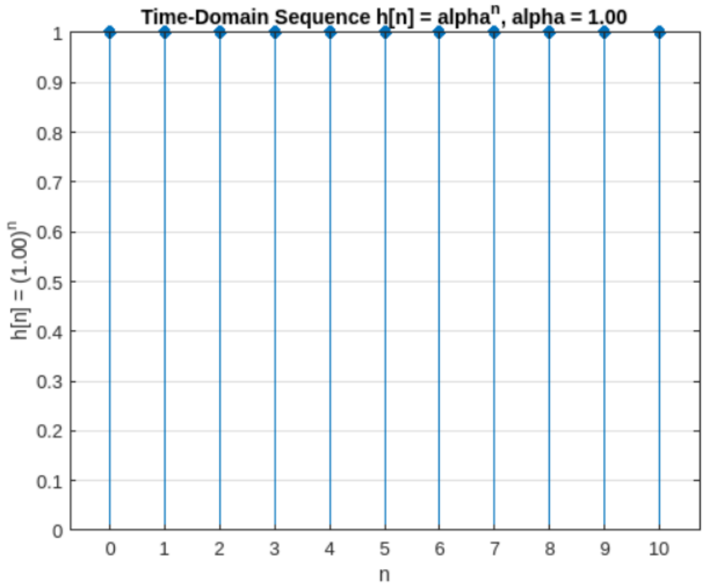
outputs:



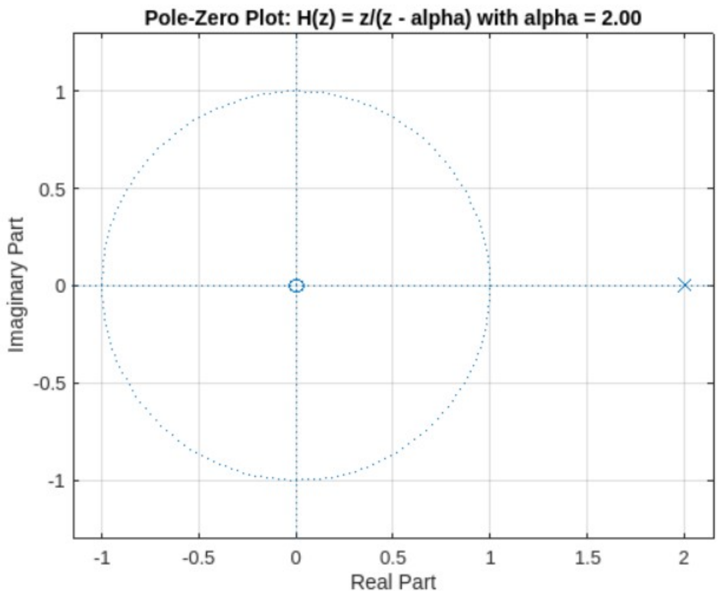
Time-Domain Sequence h[n] = alpha^n, alpha = 0.50
h[n] = (0.50)^n
n



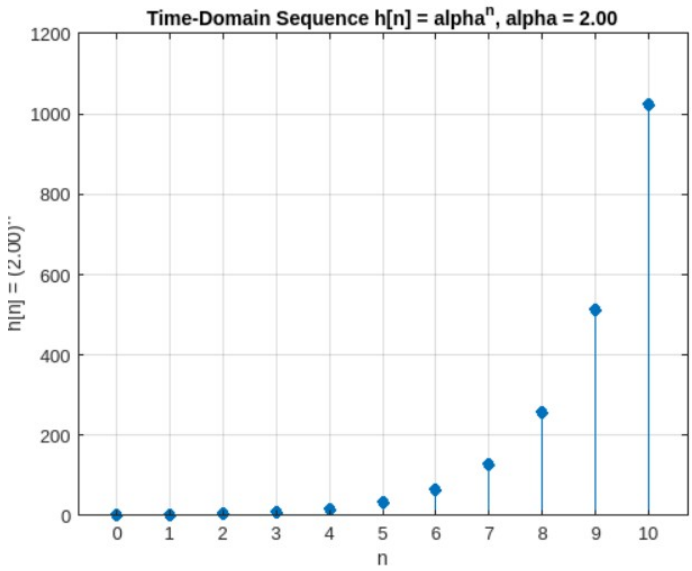
Pole-Zero Plot: H(z) = z/(z - alpha) with alpha = 1.00
Imaginary Part
Real Part



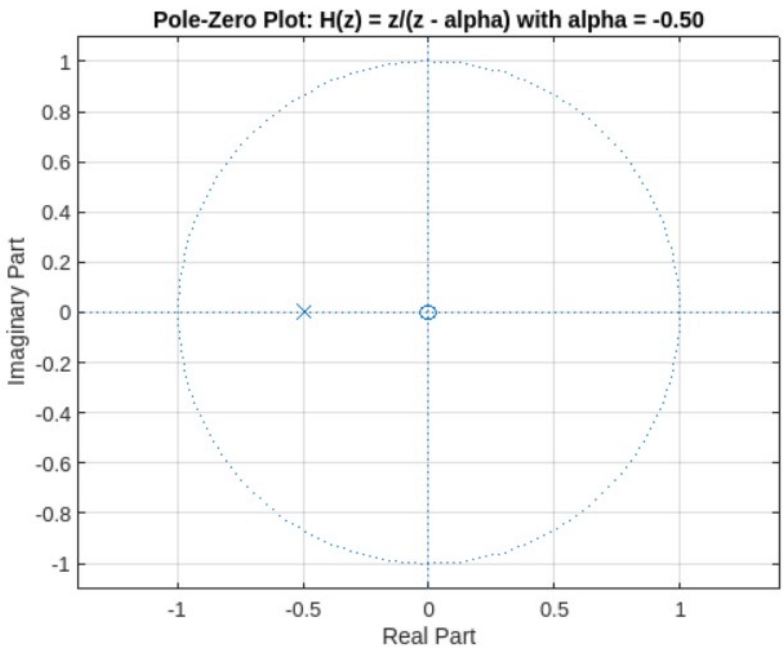
Time-Domain Sequence h[n] = alpha^n, alpha = 1.00
h[n] = (1.00)^n
n



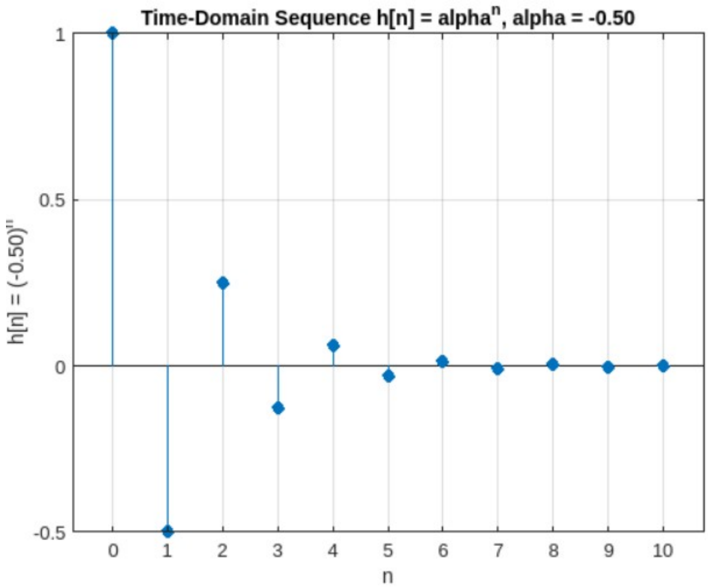
Pole-Zero Plot: H(z) = z/(z - alpha) with alpha = 2.00
Imaginary Part
Real Part



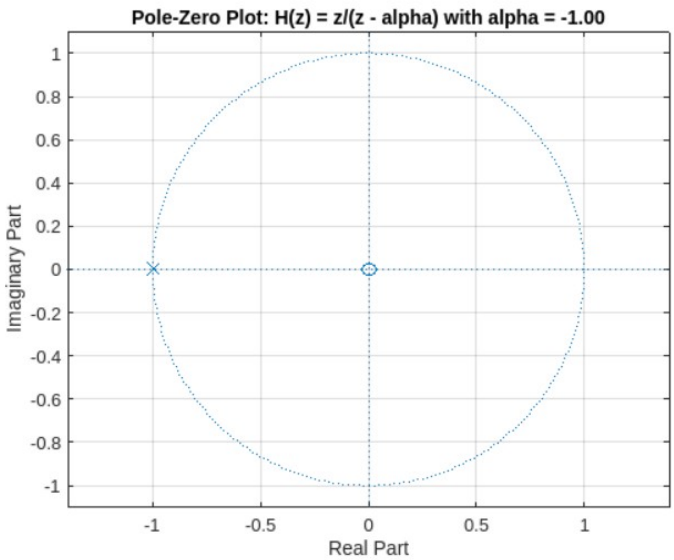
Time-Domain Sequence h[n] = alpha^n, alpha = 2.00
1200
1000
800
600
400
200
0
0 1 2 3 4 5 6 7 8 9 10
n



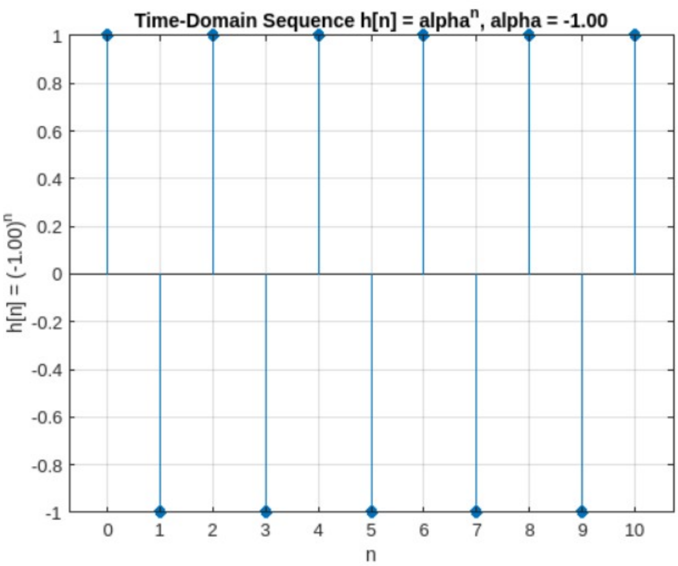
Pole-Zero Plot: H(z) = z/(z - alpha) with alpha = -0.50
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
Imaginary Part
-1 -0.5 0 0.5 1
Real Part



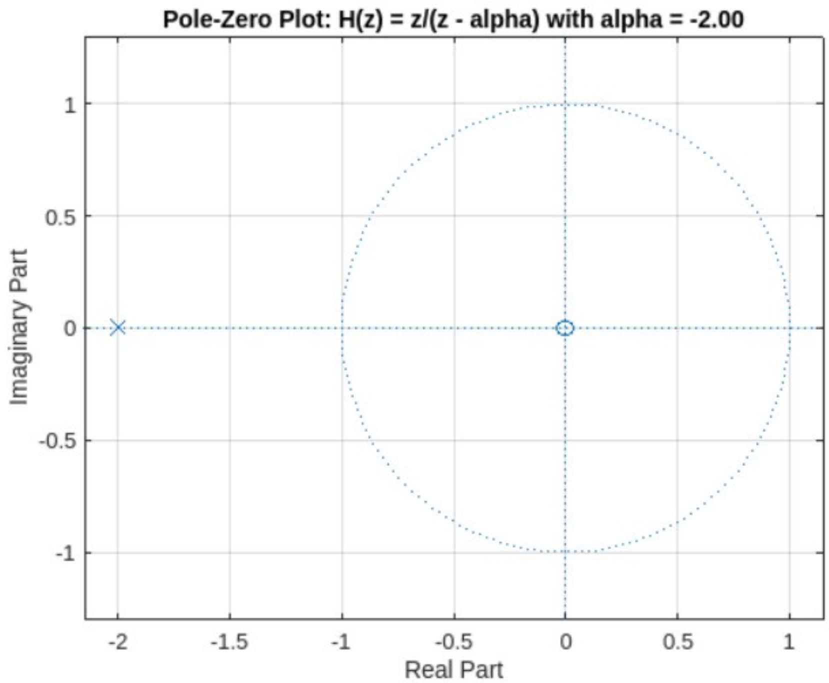
Time-Domain Sequence h[n] = alpha^n, alpha = -0.50
h[n] = (-0.50)^n
n



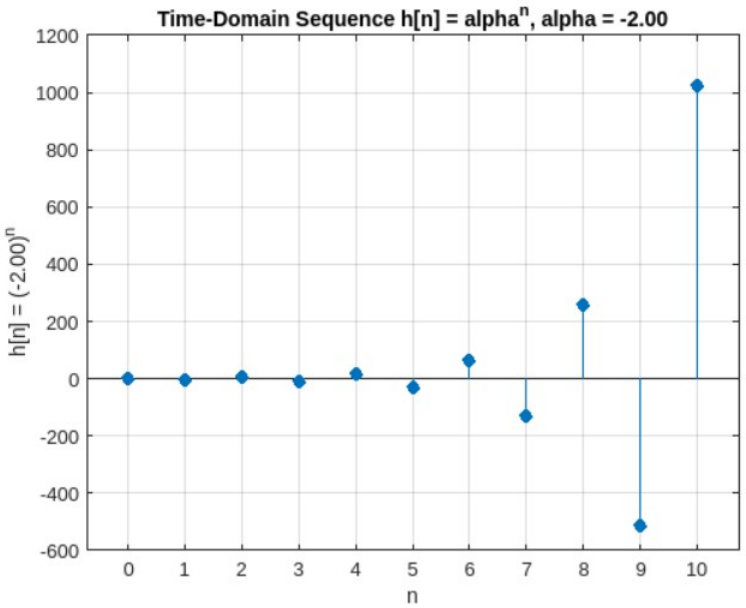
Pole-Zero Plot: H(z) = z/(z - alpha) with alpha = -1.00
Imaginary Part
Real Part



Time-Domain Sequence h[n] = alpha^n, alpha = -1.00
h[n] = (-1.00)^n
n



Pole-Zero Plot: H(z) = z/(z - alpha) with alpha = -2.00
Imaginary Part
Real Part
1
0.5
0
-0.5
-1
-2
-1.5
-1
-0.5
0
0.5
1



Time-Domain Sequence h[n] = alpha^n, alpha = -2.00
h[n] = (-2.00)^n
n
1200
1000
800
600
400
200
0
-200
-400
-600
0
1
2
3
4
5
6
7
8
9
10

```
============================================================
         Classification for H(z) = z/(z - alpha)
============================================================
 alpha   Monotonic  Alternating  Convergent  Constant  Divergent
------------------------------------------------------------
  0.50       Yes          No          Yes          No          No
  1.00       Yes          No           No         Yes          No
  2.00       Yes          No           No          No         Yes
 -0.50        No         Yes          Yes          No          No
 -1.00        No         Yes           No          No         Yes
 -2.00        No         Yes           No          No         Yes
============================================================
```

Observation:

For α = 0.5, the sequence is positive, decreasing gradually, and converges toward zero, thus it's monotonic, convergent, and stable. The pole at z = 0.5 is inside the unit circle, confirming system stability.
For α = 1, the sequence is constant (always equal to one). The pole is exactly on the unit circle at z = 1, causing the system to be marginally stable rather than strictly stable or unstable.

For α = 2, the sequence grows exponentially, making it monotonic and divergent. The pole at z = 2 lies outside the unit circle, indicating instability.

When α is negative (e.g., α = -0.5, -1, -2), the sequence alternates signs. For α = -0.5, the magnitude of the terms decreases, making it alternating yet convergent. Its pole at z = -0.5 is within the unit circle, confirming stability. For α = -1, the sequence oscillates without diminishing, hence alternating and divergent, as the pole is at z = -1 (on the unit circle), indicating marginal stability. Finally, for α = -2, the sequence oscillates with exponentially increasing magnitude, thus alternating and divergent. The pole at z = -2 is outside the unit circle, clearly showing instability.

Task 6)

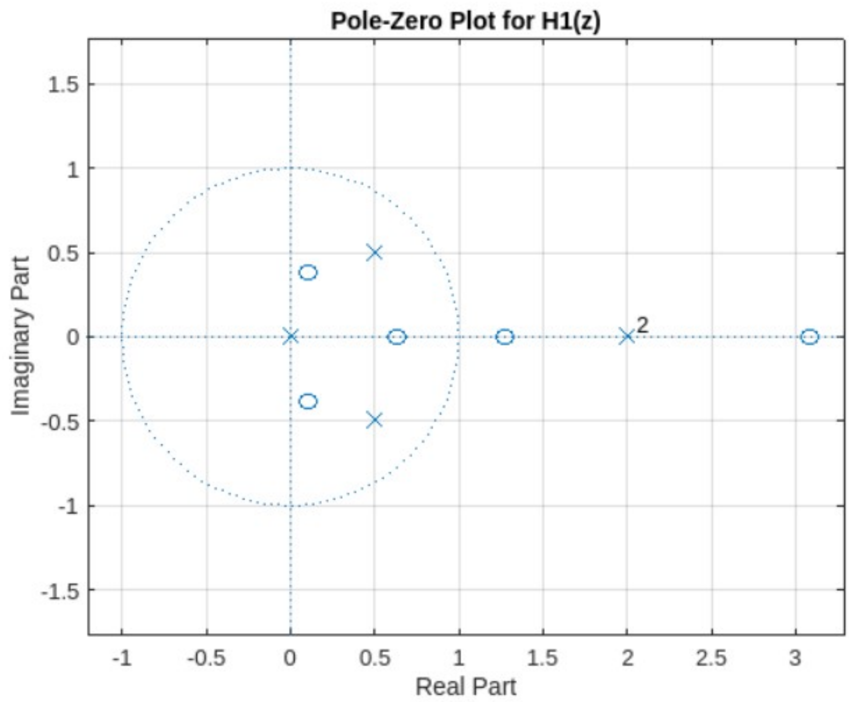
code:

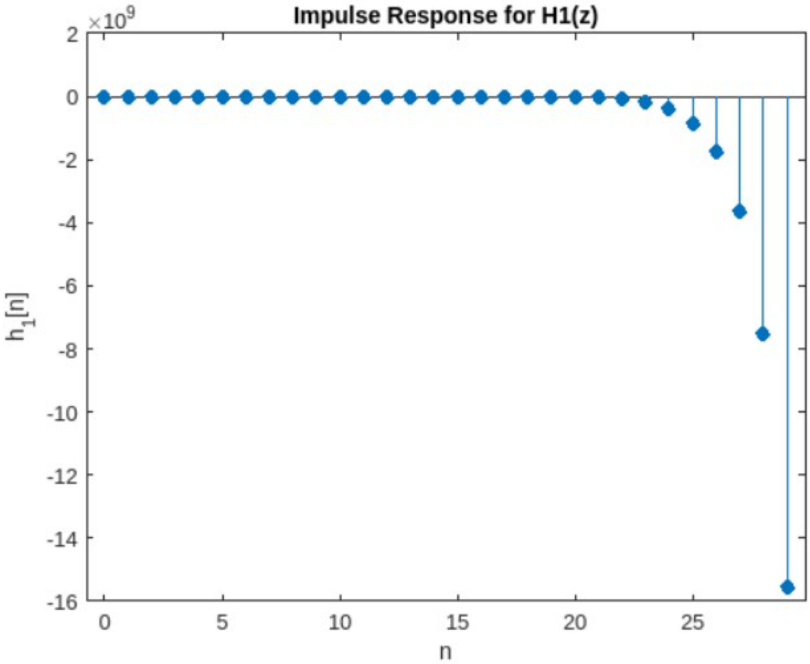
```
% Task 6
clear; close all; clc;

% H1(z) = (5 - 26z^-1 + 39.5z^-2 - 23.5z^-3 + 8z^-4 - 2z^-5)
%         / (1 - 5z^-1 + 8.5z^-2 - 6z^-3 + 2z^-4)
b1 = [5, -26, 39.5, -23.5, 8, -2];
a1 = [1, -5, 8.5, -6, 2];

figure;
zplane(b1, a1);
title('Pole-Zero Plot for H1(z)');
grid on;

[imp1, n1] = impz(b1, a1, 30);
figure;
stem(n1, imp1, 'filled');
xlabel('n'); ylabel('h_1[n]');
title('Impulse Response for H1(z)');

poles1 = roots(a1);
if all(abs(poles1) < 1)
    fprintf('H1(z): Stable => all poles inside unit circle.\n');
    fprintf('Impulse response converges.\n');
else
    fprintf('H1(z): Unstable => some pole(s) >= 1 in magnitude.\n');
    fprintf('Impulse response diverges.\n');
end
fprintf('-------------------------------------------------------\n\n');


%H2(z) = (3z^2 - z - 0.75)/z^2,  => 3 - z^-1 - 0.75z^-2
%ROC: |z| > 0.5
b2 = [3, -1, -0.75];
a2 = 1;

figure;
zplane(b2, a2);
title('Pole-Zero Plot for H2(z)');
grid on;

[imp2, n2] = impz(b2, a2, 30);
figure;
stem(n2, imp2, 'filled');
xlabel('n'); ylabel('h_2[n]');
title('Impulse Response for H2(z)');

fprintf('H2(z): FIR => poles at z=0 => inside unit circle => stable.\n');
fprintf('Impulse response is finite in length => convergent.\n');
fprintf('-------------------------------------------------------\n\n');

% H3(z)=(3 - 2z^-1 + 0.5z^-2) / (1 - 1.5z^-1 + z^-2 - 0.25z^-3)
```

```
% Case 1: ROC: |z| > 0.5
% Case 2: ROC: |z| > 2
b3 = [3, -2, 0.5];
a3 = [1, -1.5, 1, -0.25];

%Case 1
figure;
zplane(b3, a3);
title('Pole-Zero Plot for H3(z), Case 1: |z|>0.5');
grid on;

[imp3_case1, n3] = impz(b3, a3, 30);
figure;
stem(n3, imp3_case1, 'filled');
xlabel('n'); ylabel('h_3[n]');
title('Impulse Response for H3(z), Case 1: |z|>0.5');

poles3 = roots(a3);
if all(abs(poles3) < 1)
    fprintf('H3(z), Case 1: Poles <1 => stable => convergent.\n');
else
    fprintf('H3(z), Case 1: Unstable => diverges.\n');
end
fprintf('(Assuming a right-sided system, ROC is outside the largest pole => |z|>0.5.)\n\n');

%Case 2
figure;
zplane(b3, a3);
title('Pole-Zero Plot for H3(z), Case 2: |z|>2');
grid on;

fprintf('H3(z), Case 2: If we force ROC |z|>2, but largest pole is <2,\n');
fprintf('that implies a non-causal (left-sided) realization.\n');
fprintf('Impulse response for a right-sided system is still the same as Case 1.\n');
fprintf('------------------------------------------------------\n\n');


% H4(z) = 1 / (1 - 1.25z^-1 + 0.5z^-2 - 0.0625z^-3)
b4 = 1;
a4 = [1, -1.25, 0.5, -0.0625];

figure;
zplane(b4, a4);
title('Pole-Zero Plot for H4(z)');
grid on;

[imp4, n4] = impz(b4, a4, 30);
figure;
stem(n4, imp4, 'filled');
xlabel('n'); ylabel('h_4[n]');
title('Impulse Response for H4(z)');

poles4 = roots(a4);
if all(abs(poles4) < 1)
    fprintf('H4(z): All poles inside unit circle => stable.\n');
else
    fprintf('H4(z): Some pole(s) >=1 => unstable.\n');
end
fprintf('------------------------------------------------------\n\n');
```

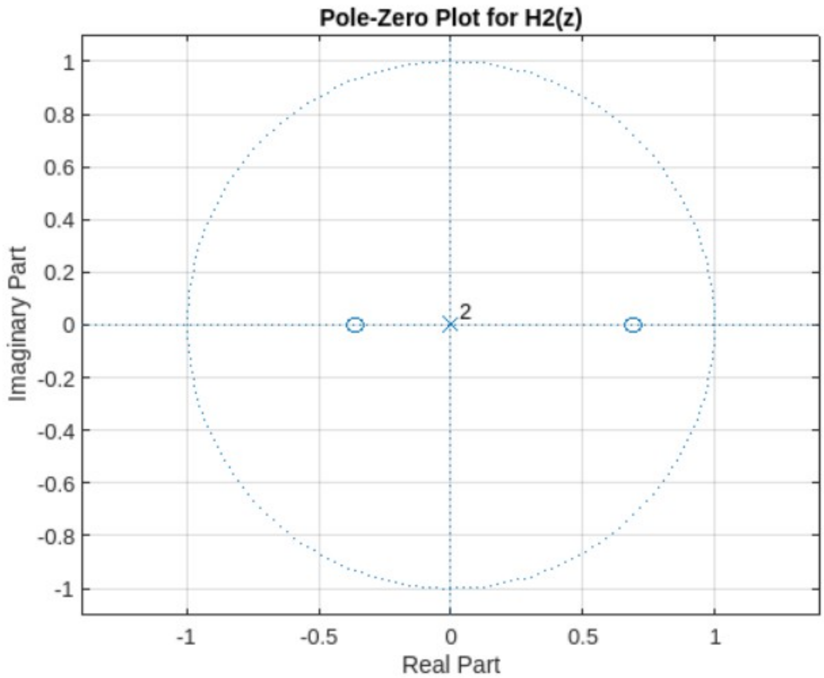
```
% H5(z)=(1 + 0.4z^-1 - 2.2z^-2 + 0.8z^-3) / (1 - 1.3z^-1 + 0.4z^-2)
b5 = [1, 0.4, -2.2, 0.8];
a5 = [1, -1.3, 0.4];

figure;
zplane(b5, a5);
title('Pole-Zero Plot for H5(z)');
grid on;

[imp5, n5] = impz(b5, a5, 30);
figure;
stem(n5, imp5, 'filled');
xlabel('n'); ylabel('h_5[n]');
title('Impulse Response for H5(z)');

poles5 = roots(a5);
if all(abs(poles5) < 1)
    fprintf('H5(z): Poles inside unit circle => stable => convergent.\n');
else
    fprintf('H5(z): Unstable => diverges.\n');
end
```

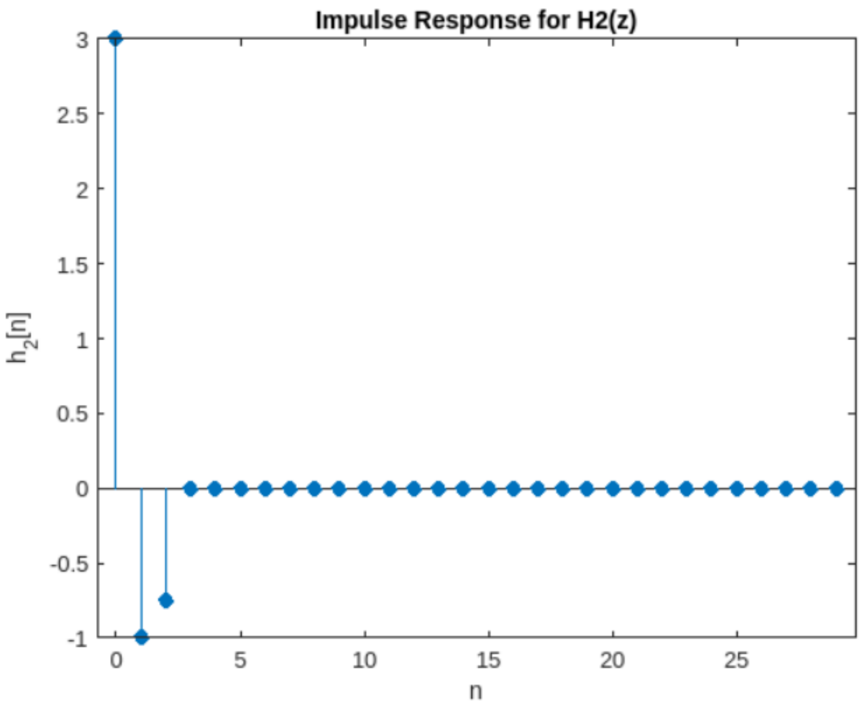
outputs:



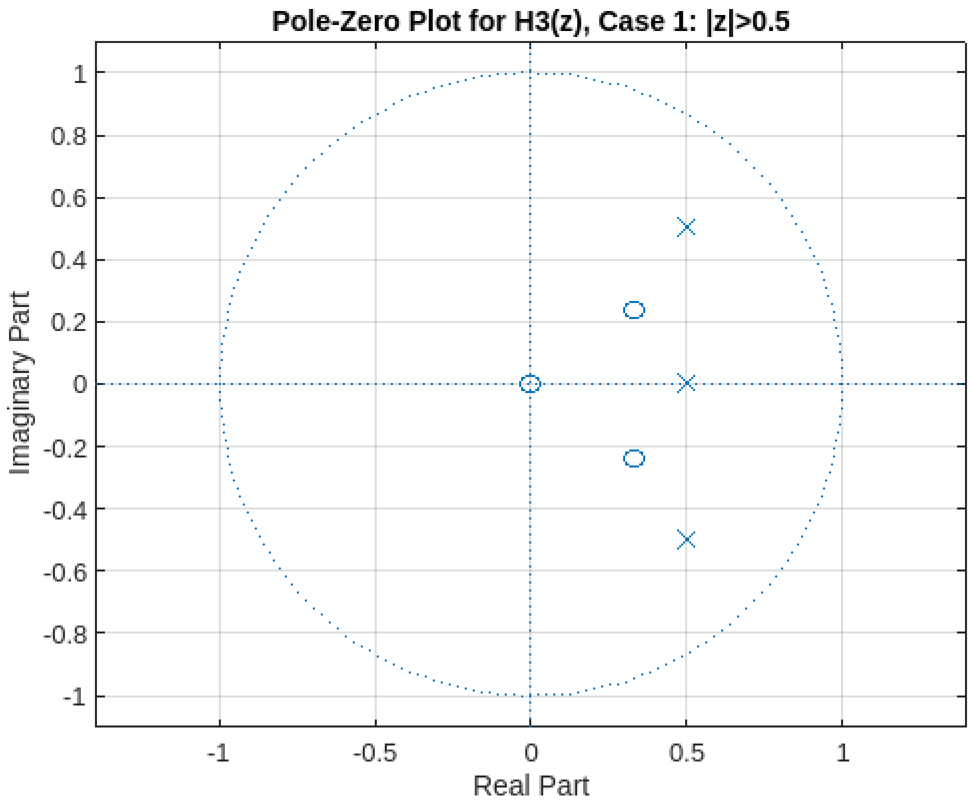


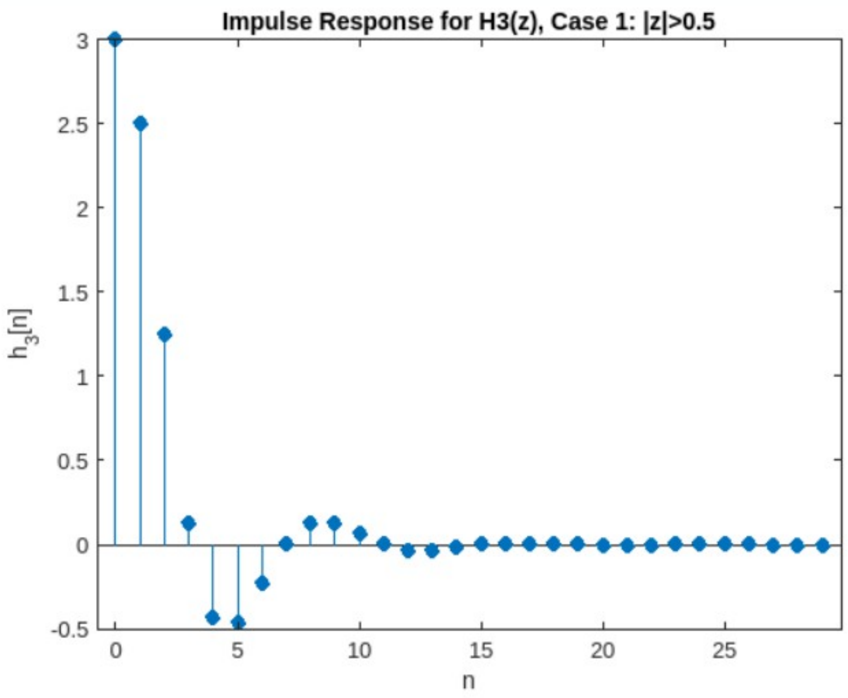
H1(z): Unstable => some pole(s) >= 1 in magnitude.
Impulse response diverges.



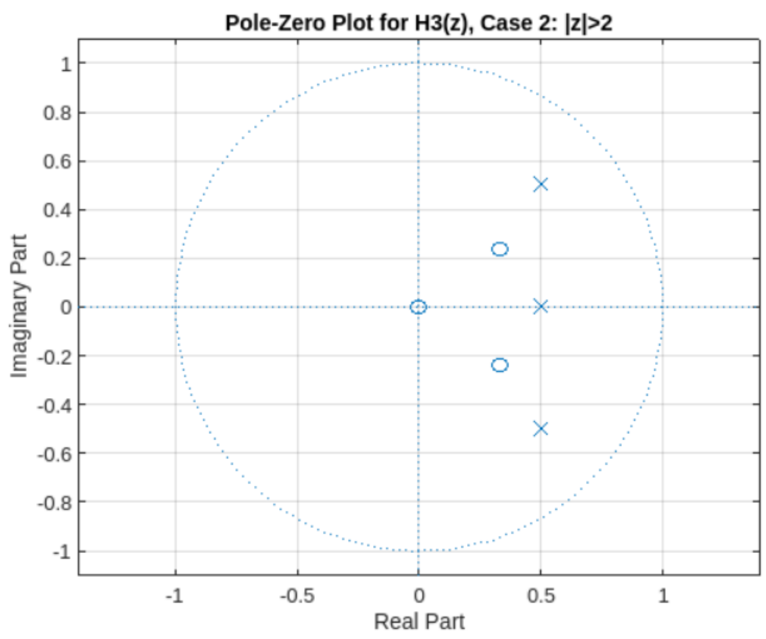


H2(z): FIR => poles at z=0 => inside unit circle => stable.
Impulse response is finite in length => convergent.

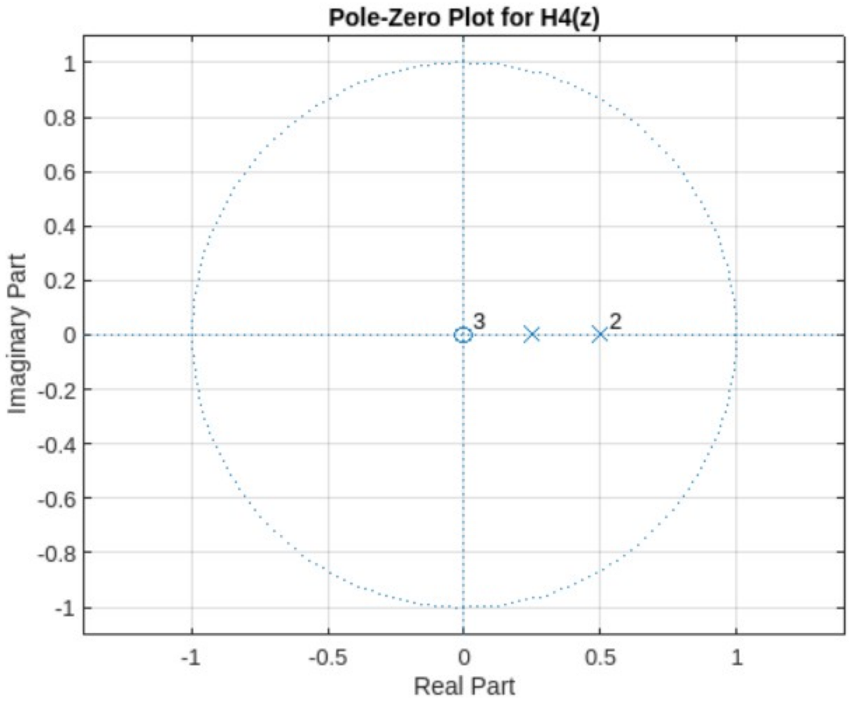


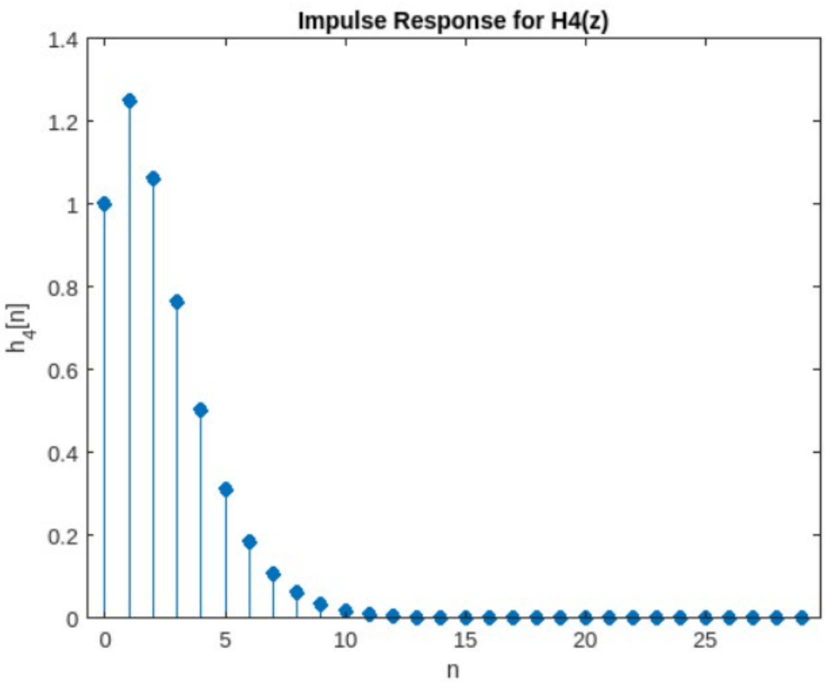


H3(z), Case 1: Poles <1 => stable => convergent.
(Assuming a right-sided system, ROC is outside the largest pole => |z|>0.5.)

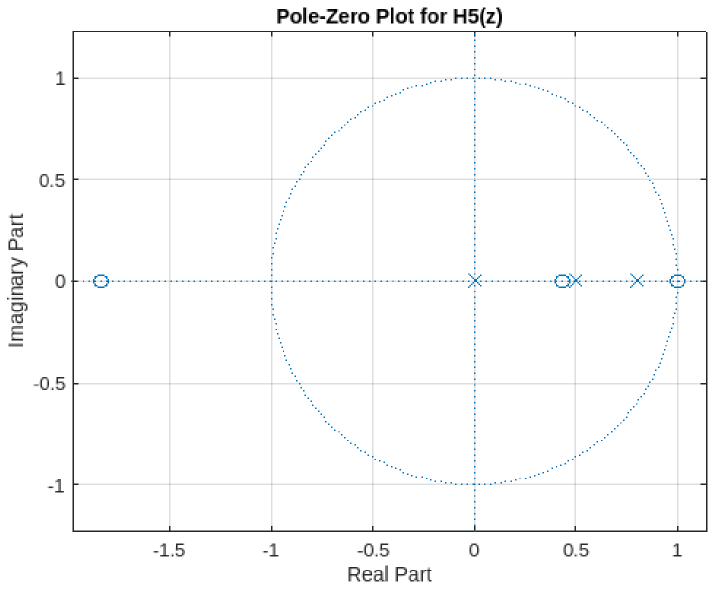


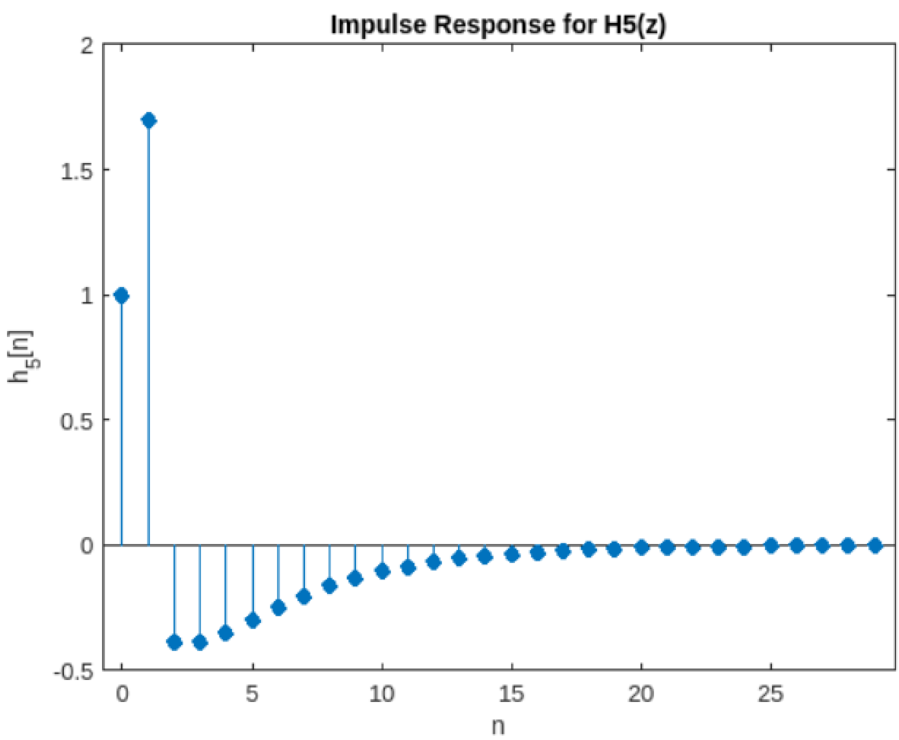
H3(z), Case 2: If we force ROC |z|>2, but largest pole is <2,
that implies a non-causal (left-sided) realization.
Impulse response for a right-sided system is still the same as Case 1.






```
H4(z): All poles inside unit circle => stable.
```






```
H5(z): Poles inside unit circle => stable => convergent.
```

Observation:

we analyzed various discrete-time systems through their pole-zero plots and impulse responses. For H1(z), poles were found outside the unit circle (magnitude greater than 1), causing the impulse response to diverge. Hence, the system is unstable. For H2(z), no poles exist (all poles at z=0), making it a Finite Impulse Response (FIR) system. Its impulse response quickly settles to zero, confirming stability and convergence. In the analysis of H3(z), Case 1 assumed a causal (right-sided) system with ROC |z| > 0.5. Here, all poles were within the unit circle, giving a stable and convergent impulse response. However, in Case 2, setting ROC as |z| > 2 implied a non-causal (left-sided) system, though the right-sided impulse response remains unchanged. H4(z) showed all poles strictly inside the unit circle, indicating stability, and the impulse response decreased smoothly to zero, verifying convergence. Lastly, H5(z) also had poles inside the unit circle, ensuring a stable and convergent system, with the impulse response stabilizing rapidly to zero.

Task 7

Problem 1)

code:

```
% Task 7, Problem 1:
clear; close all; clc;
syms z n real
Hz = (z + 0.5) / (z + 0.25);
h_sym = iztrans(Hz, z, n);

disp('--- Symbolic Transfer Function H(z) ---');
disp(Hz);
disp(' ');

disp('--- Symbolic Impulse Response h[n] ---');
disp(h_sym);
disp(' ');

disp('--- First 5 Samples of h[n] from Symbolic Math ---');
for k = 0:4
    val_k = subs(h_sym, n, k);
    val_num = double(val_k);
    fprintf('h[%d] = %f\n', k, val_num);
end
disp(' ');
b = [1, 0.5];
a = [1, 0.25];
N = 5;
impulse_input = [1, zeros(1, N-1)];
h_filter = filter(b, a, impulse_input);

disp('--- Verification via filter() for first 5 samples ---');
for k = 0:4
    fprintf('h[%d] = %f\n', k, h_filter(k+1));
end
```

output:

```
--- Symbolic Transfer Function H(z) ---
```

$$\frac{z+\frac{1}{2}}{z+\frac{1}{4}}$$

```
--- Symbolic Impulse Response h[n] ---
```

$$2\,\delta_{n,0} - \left(-\frac{1}{4}\right)^n$$

```
--- First 5 Samples of h[n] from Symbolic Math ---

h[0] = 1.000000
h[1] = 0.250000
h[2] = -0.062500
h[3] = 0.015625
h[4] = -0.003906
```

```
--- Verification via filter() for first 5 samples ---

h[0] = 1.000000
h[1] = 0.250000
h[2] = -0.062500
h[3] = 0.015625
h[4] = -0.003906
```

Observation:

the symbolic transfer function H(z)=z+0.5/z+0.25 was analyzed. The symbolic impulse response obtained is h[n]=2δ[n]−(−1/4)^n. This indicates the response has an immediate impulse (with magnitude 2 at n=0), and a decaying alternating exponential component with base −1/4. The first few samples clearly show this behavior: at n=0, there's a strong initial response (h[0]=1); then the values quickly decrease and alternate in sign, becoming very small as n increases (h[1]=0.25, h[2]=−0.0625, and so on). The numerical verification with MATLAB's `filter()` function perfectly matches the symbolic calculation, confirming the accuracy of the results. Since the impulse response shrinks quickly towards zero, the system is stable and its output response settles rapidly.

Problem 2)

code:

```
% Task 7, Problem 2:
clear; close all; clc;
syms z n real
Hz = (z^2 + 0.5)/(z^2 + 0.25);
disp('--- Symbolic Transfer Function H(z) ---');
disp(Hz);

h_sym = iztrans(Hz, z, n);
disp('--- Symbolic Impulse Response h[n] ---');
disp(h_sym);

disp('--- First 6 Samples of h[n] (Symbolic) ---');
for k = 0:5
    val_k = subs(h_sym, n, k);
    val_num = double(val_k);
    fprintf('h[%d] = %f\n', k, val_num);
end


b = [1, 0, 0.5];
a = [1, 0, 0.25];

N = 6;
impulse_input = [1, zeros(1, N-1)];
h_filter = filter(b, a, impulse_input);

disp('--- Numerical Impulse Response via filter() ---');
for k = 0:5
    fprintf('h[%d] = %f\n', k, h_filter(k+1));
end


disp(' ');
disp('--- Part (b): Find x[n] if y[n] = (1/2)^n u[n] ---');

Yz = z/(z - 0.5);
Xz_simpl = simplify(Yz / Hz);

disp('--- Symbolic X(z) that yields y[n] = (1/2)^n ---');
disp(Xz_simpl);

x_sym = iztrans(Xz_simpl, z, n);
disp('--- Symbolic x[n] ---');
disp(x_sym);

disp('--- First 6 Samples of x[n] (Symbolic) ---');
for k = 0:5
    x_val_k = subs(x_sym, n, k);
    x_val_num = double(x_val_k);
    fprintf('x[%d] = %s  (%.4f)\n', k, string(x_val_k), x_val_num);
end
```

output:

```
--- Symbolic Transfer Function H(z) ---
```

$$\frac{z^2+\frac{1}{2}}{z^2+\frac{1}{4}}$$

```
--- Symbolic Impulse Response h[n] ---
```

$$2\,\delta_{n,0}+\frac{\left(-\frac{1}{2}\,\mathrm{i}\right)^{n-1}\,\mathrm{i}}{4}-\frac{\left(\frac{1}{2}\,\mathrm{i}\right)^{n-1}\,\mathrm{i}}{4}$$

```
--- First 6 Samples of h[n] (Symbolic) ---
h[0] = 1.000000
h[1] = 0.000000
h[2] = 0.250000
h[3] = 0.000000
h[4] = -0.062500
h[5] = 0.000000
--- Numerical Impulse Response via filter() ---
h[0] = 1.000000
h[1] = 0.000000
h[2] = 0.250000
h[3] = 0.000000
h[4] = -0.062500
h[5] = 0.000000
```

```
--- Part (b): Find x[n] if y[n] = (1/2)^n u[n] ---
--- Symbolic X(z) that yields y[n] = (1/2)^n ---
```

$$\frac{z\,\left(z^2+\frac{1}{4}\right)}{\left(z^2+\frac{1}{2}\right)\,\left(z-\frac{1}{2}\right)}$$

```
--- Symbolic x[n] ---
```

$$\frac{2\,\left(\frac{1}{2}\right)^n}{3}-\frac{\sqrt{2}\,\left(-1+\frac{\sqrt{2}\,\mathrm{i}}{2}\right)\,\left(\frac{\sqrt{2}\,\mathrm{i}}{2}\right)^{n-1}\,\mathrm{i}}{12}-\frac{\sqrt{2}\,\left(1+\frac{\sqrt{2}\,\mathrm{i}}{2}\right)\,\left(-\frac{\sqrt{2}\,\mathrm{i}}{2}\right)^{n-1}\,\mathrm{i}}{12}$$

--- First 6 Samples of x[n] (Symbolic) ---

x[0] = 1 (1.0000)

```
x[1] = 1/3 - (2^(1/2)*((2^(1/2)*1i)/2 + 1)*1i)/12 - (2^(1/2)*((2^(1/2)*1i)/2 -
1)*1i)/12  (0.5000)
x[2] = 0  (0.0000)
x[3] = (2^(1/2)*((2^(1/2)*1i)/2 - 1)*1i)/24 + (2^(1/2)*((2^(1/2)*1i)/2 +
1)*1i)/24 + 1/12  (0.0000)
x[4] = 1/8  (0.1250)
x[5] = 1/48 - (2^(1/2)*((2^(1/2)*1i)/2 + 1)*1i)/48 - (2^(1/2)*((2^(1/2)*1i)/2 -
1)*1i)/48  (0.0625)
```

Observation:

The transfer function H(z) is given as (z2+0.5)/(z2+0.25). The symbolic impulse response h[n] shows an oscillatory behavior due to complex exponential terms and has real values with alternate zero terms, indicating that the impulse response oscillates and decays. The numerical values for the first 6 samples using symbolic math and MATLAB's filter match precisely, validating the symbolic solution.

In part (b), given output y[n]=(1/2)^n.u[n], the calculated symbolic input x[n] has complex terms, indicating a complicated relationship between input and output. The first 6 samples of x[n] are real numbers, with some terms becoming exactly zero. This solution also includes alternating zeros, showing that the input sequence required to achieve the given output has a non-trivial form. The symbolic and numerical calculations are consistent, confirming the correctness of the results.