Nice, thanks for sending the actual (x(t)).
We now use
[
x(t)=e^{-t/2},\quad 0\le t\le \pi,\quad 0\ \text{otherwise}
]
with its FT
[
X(j\omega)=\frac{1-0.2079,e^{-j\pi\omega}}{0.5+j\omega}
]
Here's MATLAB code for **Question 1: reconstruct (x(t)) from (X(j\omega)) using the Inverse Fourier Transform and plot it.**

```
clc; clear; close all;

%% Parameters
h  = 0.001;                % time step
t  = 0:h:pi;               % x(t) is non-zero only on [0, pi]
x  = exp(-t/2);            % given signal x(t)

dw = 0.05;                 % frequency step
w  = -60:dw:60;            % frequency axis (finite approx to -inf..inf)

%% (Optional) numerical FT from definition (Method-1 idea)
X_num = zeros(size(w));
for k = 1:length(w)
    X_num(k) = sum( x .* exp(-1j*w(k).*t) ) * h;
end

%% Use analytic X(jw) given in question
X_ana = (1 - 0.2079 .* exp(-1j*pi*w)) ./ (0.5 + 1j*w);

% choose which spectrum to use for reconstruction:
Xw = X_ana;           % or X_num if you want to stay purely numerical

%% Inverse Fourier Transform to reconstruct x(t)
t_rec = 0:h:pi;       % points where we reconstruct x(t)
xr = zeros(size(t_rec));

for n = 1:length(t_rec)
    xr(n) = sum( Xw .* exp(1j*w*t_rec(n)) ) * dw / (2*pi);
end

%% Plot original vs reconstructed signal
figure;
plot(t, x, 'LineWidth', 2); hold on;
plot(t_rec, real(xr), '--', 'LineWidth', 2);
```

```
xlabel('t'); ylabel('Amplitude');
legend('Original x(t) = e^{-t/2}', 'Reconstructed x_r(t)');
title('Reconstruction of x(t) from X(j\omega)');
grid on;
```

What this does:

- Uses your **actual signal** $x(t)=e^{-t/2}$ on $[0,\pi]$.
- Uses the **given $X(j\omega)$** to compute a numerical inverse FT:
  $$
  x(t)\approx \frac{1}{2\pi}\sum X(j\omega)e^{j\omega t}, \Delta\omega
  $$
- Plots original vs reconstructed so you can see how good the approximation is.

You're right, I had taken the wrong transfer function earlier.
Now we use the **correct one**:

$$
H(D)=\frac{1}{D+2} \quad\Longrightarrow\quad H(j\omega)=\frac{1}{j\omega+2}
$$

and the input

$$
x(t)=
\begin{cases}
e^{-t/2}, & 0\le t\le \pi\\[2mm]
0, & \text{otherwise}
\end{cases}
$$

1. Zero-state response manually (FT / convolution view)

From $H(j\omega)=\dfrac{1}{j\omega+2}$ we know

$$
h(t)=\mathcal{F}^{-1}{H(j\omega)}=e^{-2t}u(t)
$$

Zero-state response:

$$
y(t)=x(t)*h(t) = \int_{-\infty}^{\infty} x(\tau),h(t-\tau),d\tau
$$

Use supports: $x(\tau)\neq 0$ only for $0\le \tau\le \pi$,
$h(t-\tau)=e^{-2(t-\tau)}u(t-\tau)$.
So

$$
y(t)=\int_{0}^{\pi} e^{-\tau/2},e^{-2(t-\tau)}u(t-\tau),d\tau
=e^{-2t}\int_{0}^{\min(t,\pi)} e^{(3/2)\tau},d\tau
$$

Now piecewise:

(i) $t<0$

No overlap ⇒ $y(t)=0$.

(ii) $0\le t\le \pi$

Upper limit = t:

[
\begin{aligned}
y(t)
&= e^{-2t}\int_{0}^{t} e^{\frac{3}{2}\tau} d\tau
= e^{-2t}\left[\frac{2}{3} e^{\frac{3}{2}\tau}\right]_0^{t} \
&= \frac{2}{3} e^{-2t}(e^{\frac{3}{2}t}-1)
= \frac{2}{3}\left(e^{-t/2}-e^{-2t}\right)
\end{aligned}
]

(iii) (t\ge \pi)

Upper limit = (\pi):
[
\begin{aligned}
y(t)
&= e^{-2t}\int_{0}^{\pi} e^{\frac{3}{2}\tau} d\tau
= e^{-2t}\left[\frac{2}{3} e^{\frac{3}{2}\tau}\right]_0^{\pi} \
&= \frac{2}{3}\left(e^{\frac{3}{2}\pi}-1\right)e^{-2t}
\end{aligned}
]

✅ Final zero-state response

[
\boxed{
y(t)=
\begin{cases}
0, & t<0\[2mm]
\frac{2}{3}\left(e^{-t/2}-e^{-2t}\right), & 0\le t\le \pi\[2mm]
\frac{2}{3}\left(e^{\frac{3}{2}\pi}-1\right)e^{-2t}, & t\ge \pi
\end{cases}
}
]
This is exactly the inverse FT of
(Y(j\omega)=X(j\omega)H(j\omega)) with (H(j\omega)=1/(j\omega+2)).

2. MATLAB implementation using FT approach

Here we:

1. Define (x(t)).
2. Numerically compute its FT (X(j\omega)).
3. Multiply by (H(j\omega)=1/(j\omega+2)).
4. Do numerical inverse FT to get (y(t)) and plot.

```
clc; clear; close all;

%% Time axis for x(t)
h  = 0.001;              % time step
t  = 0:h:pi;             % x(t) non-zero only here
x  = exp(-t/2);          % x(t) = e^{-t/2}, 0<=t<=pi

%% Frequency axis
```

```
dw = 0.05;
w  = -80:dw:80;

%% Numerical Fourier Transform of x(t)
Xw = zeros(size(w));
for k = 1:length(w)
    Xw(k) = sum( x .* exp(-1j*w(k).*t) ) * h;
end

%% System transfer function H(jw) = 1/(j*w + 2)
Hw = 1 ./ (1j*w + 2);

%% Output spectrum
Yw = Xw .* Hw;

%% Inverse FT to obtain y(t)
t_out = 0:h:8;            % output time axis (0 to a bit beyond pi)
y_rec = zeros(size(t_out));

for n = 1:length(t_out)
    y_rec(n) = sum( Yw .* exp(1j*w*t_out(n)) ) * dw / (2*pi);
end

%% Analytic y(t) for comparison
y_analytic = zeros(size(t_out));
t = t_out;

% 0 <= t <= pi
idx1 = (t >= 0) & (t <= pi);
y_analytic(idx1) = (2/3)*(exp(-t(idx1)/2) - exp(-2*t(idx1)));

% t >= pi
idx2 = (t > pi);
y_analytic(idx2) = (2/3)*(exp(1.5*pi) - 1).*exp(-2*t(idx2));

%% Plot comparison
figure;
plot(t_out, real(y_rec), 'LineWidth', 1.8); hold on;
plot(t_out, y_analytic, '--', 'LineWidth', 1.8);
xlabel('t'); ylabel('Amplitude');
legend('Numerical FT-based y(t)', 'Analytic y(t)');
title('Zero-State Response for H(D) = 1/(D+2)');
```

```
grid on;
```

- **Use this** for your Q2 answer (manual piecewise expression + MATLAB code).
- This now matches the correct (H(D)=1/(D+2)), not the earlier wrong one.

Alright — now solving **Q3 and Q4** from the same worksheet.

📌 Question 3

Discuss/Evaluate/Comprehend how the Fourier spectrum modifies when the following properties are applied to the above rectangle signal (or any signal).

a) Scalar addition
b) Scalar multiplication
c) Conjugation
d) Addition of two time signals
e) Time shifting
f) Time scaling
g) Frequency shifting
h) Time convolution
i) Frequency convolution
Let the basic Fourier transform pair be:
[x(t) \longleftrightarrow X(j\omega)
]

(a) Scalar Addition
[x(t) + c \quad \longleftrightarrow \quad X(j\omega) + 2\pi c\delta(\omega)
]
Adding a DC term adds a **delta spike at ω=0**.

(b) Scalar Multiplication
[a,x(t) \quad \longleftrightarrow \quad aX(j\omega)
]
Spectrum amplitude scales by the same constant.

(c) Conjugation
[x^*(t) \quad \longleftrightarrow \quad X^*(-j\omega)]
Spectrum flips and conjugates.

(d) Addition of Two Signals
[x_1(t)+x_2(t)\longleftrightarrow X_1(j\omega)+X_2(j\omega)
]
Spectrum adds pointwise.

(e) Time Shifting
x(t-t_0)\quad\longleftrightarrow\quad e^{-j\omega t_0}X(j\omega)
]
Spectrum multiplied by **complex exponential phase shift**.

(f) Time Scaling
[x(at)\quad\longleftrightarrow\quad \frac{1}{|a|}X\left(\frac{\omega}{a}\right)
]
Compression in time ↔ expansion in frequency and vice-versa.

(g) Frequency Shifting
[e^{j\omega_0 t}x(t)\quad\longleftrightarrow\quad X(j(\omega-\omega_0))
]
Spectrum shifts left/right.

(h) Time Convolution

[x(t) * h(t)\quad\longleftrightarrow\quad X(j\omega)H(j\omega)
]
Convolution in time becomes multiplication in frequency.

(i) Frequency Convolution
[x(t)h(t)\quad\longleftrightarrow\quad \frac{1}{2\pi}X(j\omega)*H(j\omega)
]
Multiplication in time becomes convolution in frequency.

📌 Question 4

Compute the Fourier Transform of the following signals and sketch the spectra:
unit triangle, **unit half-triangle**, **unit rectangle**, **signum**, **sinc**, **cos(t) from 0 to π/2**, **sin(t) from 0 to π**, and **combinations**
Below are the standard known results:

1. Unit Rectangle
[x(t)=1,\ |t|\le \frac{T}{2}
]
[X(j\omega)=T,\text{sinc}\left(\frac{\omega T}{2}\right)
]

2. Unit Triangle
[\Lambda(t)=1-|t|,\ |t|\le 1
]
[X(j\omega)=\text{sinc}^2\left(\frac{\omega}{2}\right)
]

3. Half Triangle
[x(t)=t,\ 0\le t\le 1
]
[X(j\omega)=\frac{1 - e^{-j\omega}(1+j\omega)}{\omega^2}
]

4. Signum Signal
[\text{sgn}(t)=\begin{cases}1 & t>0\-1 & t<0\end{cases}]
[X(j\omega)=\frac{2}{j\omega}]

5. Sinc signal
[x(t)=\text{sinc}(t)=\frac{\sin(\pi t)}{\pi t}
]
[X(j\omega)=\text{rect}\left(\frac{\omega}{2\pi}\right)]

6. (\cos(t)) from (0) to (\pi/2)
[x(t)=\cos t \cdot u(t)u(\tfrac{\pi}{2}-t)
]
[X(j\omega) = \frac{\omega \sin(\omega\pi/2) + \cos(\omega\pi/2)}{\omega^2 - 1}
]

7. (\sin(t)) from (0) to (\pi)
[X(j\omega)=\frac{\omega \cos(\omega\pi)-\sin(\omega\pi)}{\omega^2+1}
]

8. Combination
If (x(t)=x_1(t)+x_2(t)):
[X(j\omega)=X_1(j\omega)+X_2(j\omega)]
Would you like MATLAB code to plot all spectra?
I can generate:

Option A – Plot for rectangle, triangle, sinc, signum
Option B – Code for cos, sin limited-time signals
Option C – Full combined script
Just tell me:
A / B / C ?
Example:
give option C MATLAB
and I will produce clean plotting code for all signals.