Problem 1: a
Priority Encoder:
First, start with an Encoder:

\[
\begin{array}{c}
1_3 \\
1_2 \\
1_1 \\
1_0 \\
\end{array}
\xrightarrow{E}
\begin{array}{c}
A_1 \\
A_0 \\
\end{array}
\]
with behavior of: 1-hot inputs converted to BCD of input bits, so 0100 → 10.

So, we must find away to convert the 01xx of priority encoding into the 0100 of the normal Encoder.

To do this, we create an element that detects if you have seen an 1
\[
\text{saw-one}[i-1]
\]
so, for any \( i \), you want a 1 if \[\text{saw-one}[i] \text{ hasn't seen a 1, then } \text{ if } x[i] = 1, \text{ y}[i] = 1, \text{ else it is not, so } y[i] = \text{saw-one}[i] \]

Where:
\[
\begin{align*}
\text{y}[i] &= \text{saw-one}[i-1] \text{ and } x[i] \\
\text{saw-one}[i] &= \text{saw-one}[i-1] \text{ or } x[i] \\
\end{align*}
\]

\( V \) is invalid when \( D_3..0 \) are all 0s;

\[
V = D_3' D_2' D_1' D_0'
\]

- precoder -
so, then the full priority encoder is:

\[ \text{<decoder>} \quad \text{<encoder>} \]

b. Priority Encoder: Dual Mode

Define:

\[ D_3 \quad D_2 \quad D_1 \quad D_0 \quad A_1 \quad A_0 \quad V \]

\[ \text{<Priority encoder>} \]

We can simply connect 2 priority encoders, and then choose between them:
Problem 2: 4-bit comparator

Define:
\[ E_Q(x,y) = (x \text{ xor } y)' \]
\[ GE_Q(x,y) = (x \text{ and } y') \text{ or } E_Q(x,y) \]

Then, by logic,

\[ GE_Q_2(x_1, y_1, x_0, y_0) = GE_Q(x_1y_1) \text{ or } (E_Q(x_1y_1) \text{ and } GE_Q(x_0y_0)) \]

then, for \( GE_Q_3(x_2, y_2, x_1, y_1, x_0, y_0) \):

\[ GE_Q_3 = GE_Q(x_2y_2) \text{ or } (E_Q(x_2y_2) \text{ and } GE_Q_2(x_1y_1x_0y_0)) \]

then for \( GE_Q_4(x_3, y_3, x_2, y_2, x_1, y_1, x_0, y_0) \):

\[ GE_Q_4 = GE_Q(x_3y_3) \text{ or } (E_Q(x_3y_3) \text{ and } GE_Q_3(x_2y_2x_1y_1x_0y_0)) \]
Problem 3:

\[
\begin{array}{c}
\begin{array}{cccc}
X_2 & X_1 & X_0 \\
Y_2 & Y_1 & Y_0 \\
\hline
S_2 & S_1 & S_0
\end{array}
\end{array}
\]

Define:

\[
\begin{align*}
g_0 &= X_0 Y_0 \\
p_1 &= X_1 + Y_1 \\
g_1 &= X_1 Y_1 \\
c_2 &= g_1 + p_1 g_0 \\
g_2 &= X_2 Y_2
\end{align*}
\]

\[
S_2 = g_2 c_2 + x' 2 Y_2' c_2 + x 2 Y_2 c_2' + x' 2 Y_2 c_2'
\]
Problem 4: Check Generator

Define:

2-bit full adder:

\[
\begin{array}{c}
A \\
B \\
C_{in}
\end{array}
\begin{array}{c}
\downarrow
\end{array}
\begin{array}{cc}
S \\
C
\end{array}
\]

where:

\[
S = A \text{ xor } B \text{ xor } C_{in}
\]

\[
C = (A \text{ and } B) \text{ or } (B \text{ and } C_{in}) \text{ or } (A \text{ and } C_{in})
\]

N-bit adder:
So, check Generator:

- All lines are 1 bit wide

Then, for the Check Tester,

- Do the same as above to the Y received value, and generate your own "check" for the Y value. Then, compare your results to the received check value.

results: \( R_3:R_0 \)  
check: \( T_3:T_0 \)

Then, \( V \) is true only when \( R[3:0] = T[3:0] \)

\[
V = (T_0 \text{xnor } R_0)(T_1 \text{xnor } R_1)(T_2 \text{xnor } R_2)(T_3 \text{xnor } R_3)
\]