The goal of this problem is to design a reliable latch.

A latch can be designed using a multiplexor where the G input is used to determine whether or not a new data value should be loaded into the latch.

The way the latch works is that when G = 0, the old value of Q is fed back through the multiplexor so it retains its old value.

When G = 1, the multiplexor selects D as the input so Q gets updated to be the value of D.

The logic function that describes the operation of the latch is Q = (NOT(G) AND Q) OR (G AND D).

We want to build this latch using only AND, OR, and inverter gates.

We are given three proposed designs for the latch.

For each proposed design, we want to specify whether the design is BAD, GOOD, or OBESE.

BAD means that the latch does not work reliably.

GOOD means that the latch works reliably.

OBESE means that the latch works, but uses more gates than necessary.

Latch proposal A is shown here.

Taking a closer look at this circuit, we see that the logic function that it implements is Q = (G AND Q) OR D.

This is not the correct logic equation for a latch, so this design is BAD.

Next, we take a look at proposals B and C.

The logic equation for proposal B is Q = (NOT(G) AND Q) OR (G AND D)).

This is exactly the same logic equation that we specified for our latch.

However, this implementation is not lenient because it does not guarantee that you will not see glitches on your output signal when G changes value.

Proposal C, however, includes the same logic function Q = (NOT(G) AND Q) OR (G AND D)) plus one more term which is OR (Q AND D).

If you create a karnaugh map for both of these logic functions, you see that the QD term is redundant because it doesn't add any additional 1's to the karnaugh map.

This means that logically the two functions are equivalent, but the effect of the extra term is that it makes the implementation lenient.

So Proposal C is the GOOD proposal and B is BAD.