Note: this problem is designed to get you to think about how to apply the tools and knowledge you have gained so far to a novel situation (which is what engineers do).

The "tally count" number system is used here not because it is generally useful or common, but *exactly the opposite* – because it forces you to think about the entire design process rather than relying on memorized facts. If you get stuck here, do not just look at the solution, because the solution is ultimately meaningless; it is the process of creating a solution that is important.

Towards that end, if you do get stuck, there is a "tips" document available that can help.

Design a 2-bit +2-bit input, 4-bit output "tally count" adder.

- In the "tally" number system, numerical information is encoded using the *number* of active signals; e.g. with two bits, both 10 and 01 represent the concept of "one".
- Design the device to always right-justify the output; e.g. "0101" should never be an output, because the concept of "two" should always be encoded as "0011".
    - You cannot assume that the inputs are right-justified.
- Name the inputs $TA_{1\text{-}0}$ and $TB_{1\text{-}0}$.
- Name the outputs $TS_{3\text{-}0}$.

a)
Approach the design "from scratch," i.e., as a simple combinational logic problem. Express your resulting design as a Boolean expression for each output.

b)
Repeat the process for a 3-bit + 3-bit -> 6-bit adder. Since at this point it would be difficult to do "from scratch," approach the design assuming that you have access to already-made building blocks (all of which can be any size you want):

- unsigned binary adders
- line decoders
- priority encoders
- multiplexers
- demultiplexers

You can also include logic gates as needed (AND, OR, etc.)

Express your resulting design as a block-level schematic, labeling the connections to TA, TB, and TS.