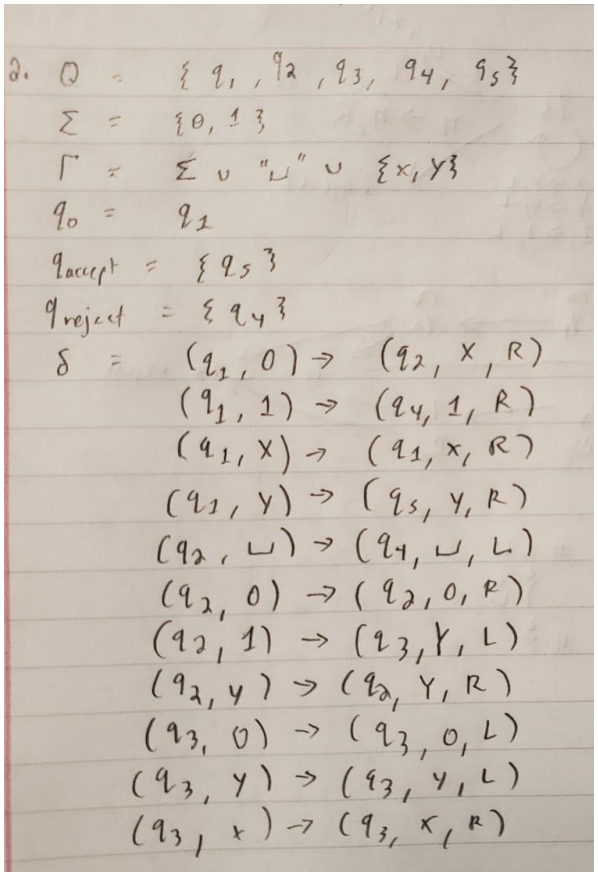
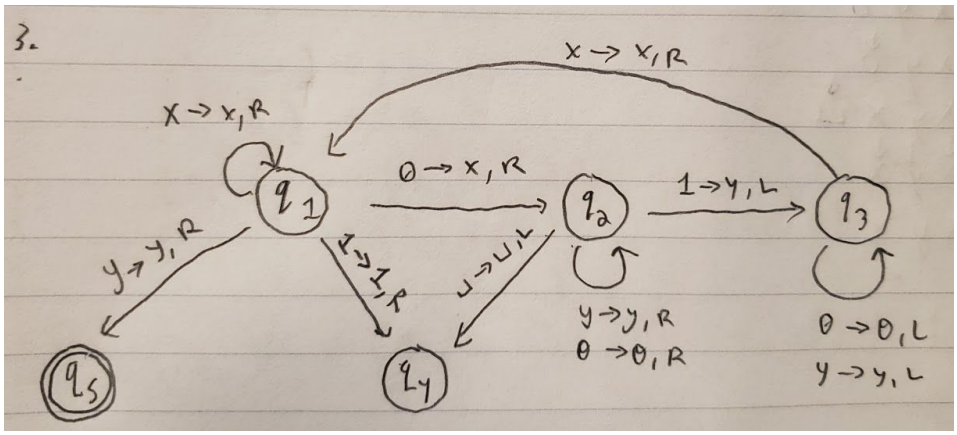
Joseph O'Neill

A. **1.** This language is a language of strings that for each 0 on the left side will be followed by the same number of ones on the right side. Therefore, the turing machine will go through and make sure there are an equal number of 0s and 1s on the left and right sides respectively.

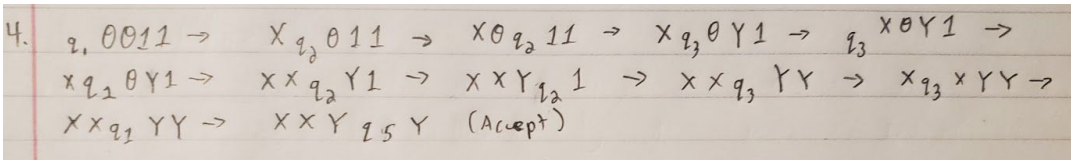
2.



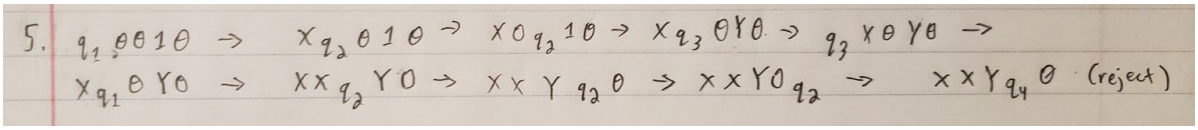
3.



4.

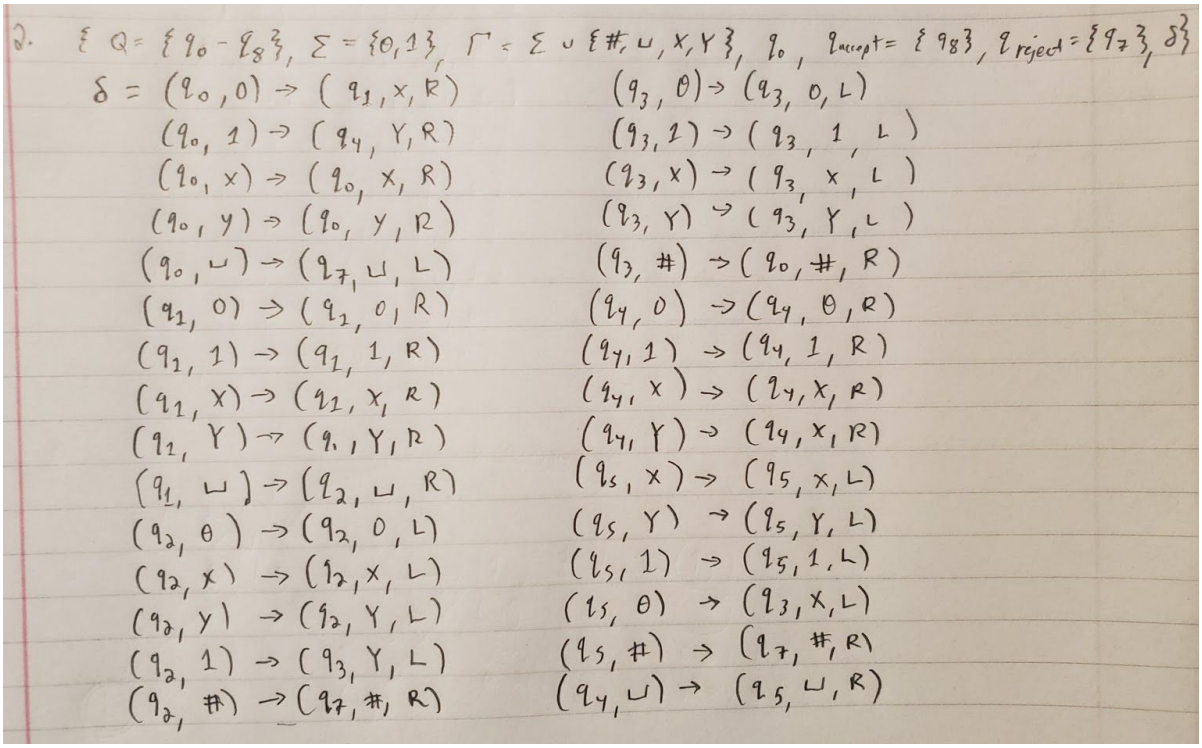


5.

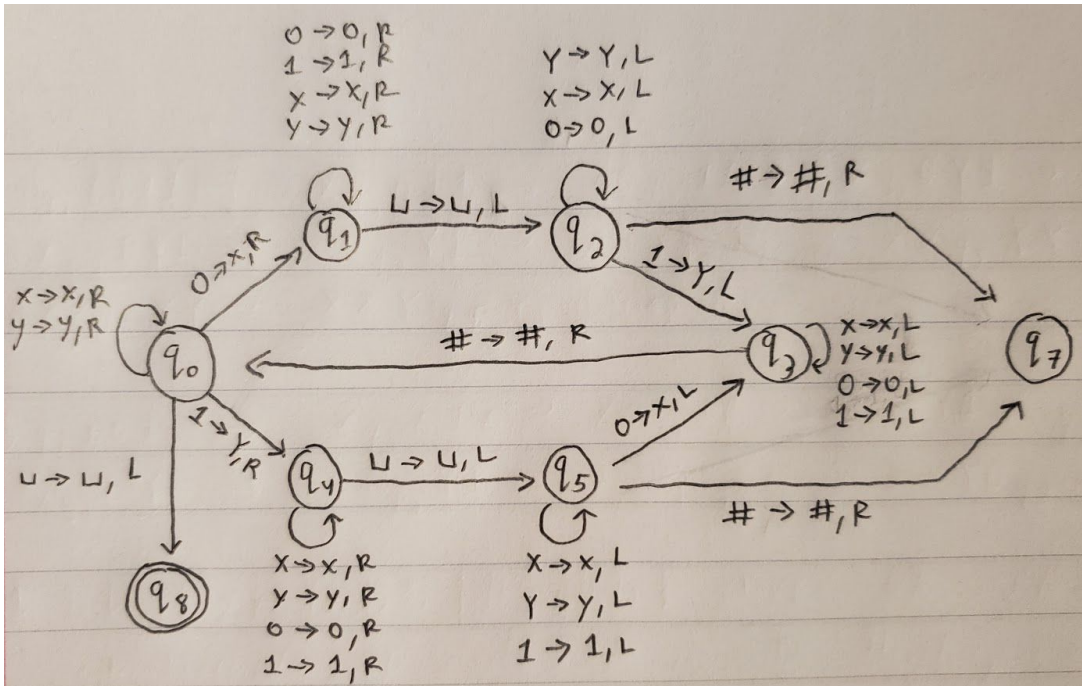


B. **1.** L = { W ∈ {0,1}* | $|W|_0 = |W|_1$ } where $|W|_0$ and $|W|_1$ are the numbers of 0 and 1 in W. Therefore, we are dealing with a language of strings that has the same number of 0s and 1s in any order. It should check the first letter and find the opposite (0 => 1, 1 => 0) and if it doesn't have it, reject, else carry on.

2.



3.



C. You would simulate a PDA in a turing machine simply by instead of using the stack, you use the tape to go through the strings. You can have the rules that would match the language the PDA uses. Since it reads/writes instead of popping off the stack, you can use the same languages as a PDA on a turing machine.

D. NTM - accepted strings are of a single string (w) doubled (ww)

- Splits string in 2 halves: w1, w2
- Deterministic ™ to see if both halves are equal: w1 == w2
- If so, accept, else, reject