1. Prove that every regular language L is decidable by providing a TM that decides L. Explain.

We know that a language is regular if there exists a DFA that accepts it. In order to represent L, we use A_{DFA}, and we then provide a Turing Machine M that decides A_{DFA}. This TM we provide will effectively simulate A_{DFA} and will accept/reject depending on the DFA's simulated final state.

M = On input ⟨B, w⟩, where B is a DFA and w is a string:
1. Simulate B on input w
2. If the simulation ends in an accept state, accept. If it ends in a nonaccepting state, reject.

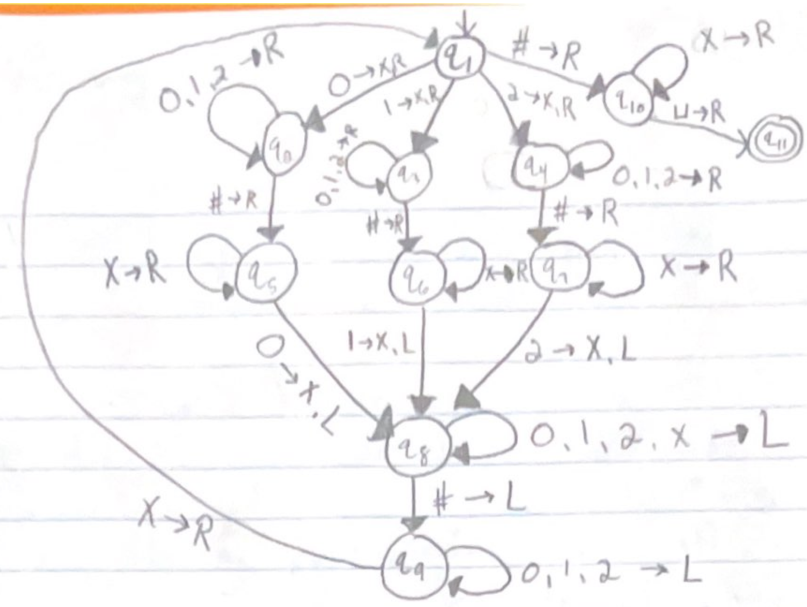
2. Give a TM that takes in binary string w and outputs w#w.
1. Read the first symbol, mark it! then move right until you reach the end of the input.
2. Write a # at the end followed by what was read
3. Go back to the beginning, mark/read the next unmarked symbol
4. Go to the end of the tape, write symbol
5. Rinse & repeat until there are no unmarked symbols and accept once that is done

3. Give a detailed description of a TM that accepts all strings in the form w#w where $w \in \{0,1,2\}^*$

<u>High level description</u>

- Read first character & mark it
- Continue until #, read next char
- If it matches our first char, mark it
- Go back to the beginning & find next unmarked char
- Repeat until no more unmarked chars in front of #
- If unmarked chars remain after # reject, otherwise accept

3. Cont.



4. Prove that the set of all infinite binary strings is not Countable. (use diagonalization)

We are going to use a proof by contradiction, assume we have S = set of all binary strings and we have $S = \{s_1, s_2, \ldots, s_n\}$ where each s_n is an infinite binary sequence. In order to choose the m^{th} element of s_n we create a new enumeration $s_{n,m}$. Now we can create a new sequence $p = p_1 p_2 p_3 \ldots$ where $p_n = 0$ when $s_{n,n} = 1$ and $p_n = 1$ when $s_{n,n} = 0$. In order to show that p isn't in S suppose that $p = s_q$ for some q. Then $p_q = s_{q,q}$ but by construction $p_q \neq s_{q,q}$ so that isn't possible, thus S is not countable.

5. Consider the problem of deciding whether a regular expression R is equivalent to a given DFA A. Define formally this decision problem and prove it's decidable

$X = \{\langle M, R \rangle \mid M$ is a DFA and R is a regular expression $L(M) = L(R)\}$

We know from class that there exists an algorithm that can convert any DFA into a regular expression. We also know that any algorithm can be modeled as a TM. We then define the TM

T = input $\langle M, R \rangle$ where M is a DFA and R is a reg expression

1. Convert M into a reg expression R_m
2. If R and R_m are equivalent then accept otherwise reject