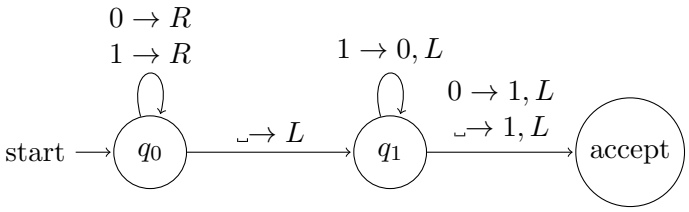
1. Suppose you have a PDA with *two* stacks instead of one, each of which can be used independently of the other for both pushes and pops. Describe how you would use the stacks in such a PDA to recognize the language $L_1 = \{a^n b^n c^n : n \geq 0\}$.

   Note that this language *is not* context free; the fact that this is possible to do establishes that two-stack PDAs recognize more languages than one-stack ones.

   You do not need to draw a state diagram for this unless you wish to do so. If you do so, please state clearly what your notation is and means. A short written description is sufficient. For example, if you were describing the use of the stack in a single-stack PDA for $\{a^n b^n\}$, you could say "We push a symbol for each a and then pop one when we read the b. If the stack is empty at the end, we saw an equal number of each."

2. Draw a Turing Machine diagram to recognize the language L_1. Give a brief outline of how it works, similar to L_3's outline from lecture.

3. Describe in English the function computed by the following Turing Machine. The input is $\Sigma = \{0, 1\}$ and is the representation of a positive integer in binary. You may assume the input is non-empty. The symbol ␣ refers to a blank space. Provide the semantics, not the mechanics, of the machine.

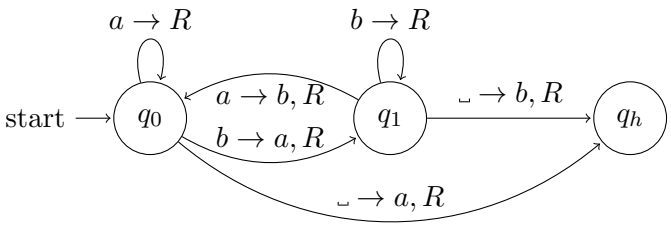


4. Draw a Turing Machine whose input alphabet is $\Sigma = \{0, 1\}$, where the input tape is the representation of a positive integer in binary. Your machine should compute the function $f(\langle n \rangle) = \lfloor n/4 \rfloor$; that is, the floor of $n/4$. This is the same as integer division in a language like C++.

   You may assume that your input is non-empty and has at least three bits (i.e., the input $n \geq 4$).

   Give a brief description of how your machine works.

5. Give a plain English description for the function $f : \{a, b\}^* \to \{a, b\}^*$ computed by the Turing Machine below. In this, q_h is a halting state.



6. Draw a Turing Machine that recognizes the language where there are an equal number of the symbol a and b. These can appear in any order in the input string.

   You may use the following strategy for a successful Turing Machine for this problem. Using this strategy is not a requirement.

   1. Sweep left to right across the tape and mark the first 'a'.
   2. If stage 1 marked an 'a', return the head to the start of the tape and do the same for 'b'. Reject if a corresponding 'b' is not found.
   3. Otherwise, stage 1 did not mark an 'a'. Sweep across the tape left to right and reject if a 'b' is found.
   4. Return the head to the start of the tape. Go back to stage 1 and repeat until all 'a' symbols are marked.

7. Fermat's Last Theorem was a famous unproven theorem until recently: it was asserted in the first half of the 17th century and was not proven until the mid-1990s.

   Fermat's Last Theorem states that, for any $n > 2$, there are no positive integer solutions to $a^n + b^n = c^n$.

   Pretend you are taking this class before it was proven, but you also have a decider for the halting problem. How could you use that to prove or disprove Fermat's Last Theorem?

8. Show that each of the following languages are decidable. You may want to consult your notes for what has been shown in class to be decidable. For the regular or context free language portions, you can check the "derivative DFAs/CFGs" from lecture. On the exam, you will be given a list of which languages were shown in lecture to be decidable.

   (a) $L_a = \{\langle D \rangle : D \text{ is a DFA that accepts a palindrome } \}$
   (b) $L_b = \{\langle D \rangle : D \text{ is a DFA and } L(d) \text{ is not recognized by a DFA smaller than } D.\}$
   (c) $L_c = \{\langle G, w \rangle : G \text{ is a CFG that generates a string containing } w \text{ as a substring.}$
   (d) $L_d = \{\langle A \rangle : A \text{ is a DFA and } L(A) \text{ is an infinite language}\}$.

9. Show that the following languages are undecidable. On the real exam, you will be given a list of which problems were shown in lecture to be undecidable.

   (a) $L_a = \langle M \rangle : M$ is a Turing Machine that accepts w^R whenever it accepts $w\}$, where w^R is the reverse of string w.
   (b) $L_b = \langle M, q \rangle : M$ is a Turing Machine and, within M, state q is a "useless" state $\}$. A "useless" state is one that is never entered on any input string.
   (c) $L_c = \langle M, w \rangle : M$ is a Turing Machine that, at some point while running on input w attempts to move its head left when the head is on the left-most initial input cell.