Theory of Computation, CSCI 438, Spring 2022
Exam 3, April 13 **Name _____**

Essay Questions

(15 pts.)

Definitions

1. State the Church-Turing Thesis. (5 pts.)

   Church-Turing Thesis – Our intuitive notion of algorithm is equivalent to Turning machine algorithms.

   Figure 3.22, page 183
   This can also be stated with Church's λ-calculus.

2. Give the format in which we will be writing algorithms in this class **and** what restrictions exist for this formalism. (We started completely defining machines, regular expressions and grammars. However, from now on we will be giving formalisms at a higher level.) (10 pts.)

   M = "On input <X, Y, …, Z> that encodes ….
   1.
   2.
   .
   .
   . "

   Restrictions:
   - X, Y, …, Z must be encodings that can be placed on a Turing tape
   - Each step of the algorithm must be something that can be accomplished on a Turing machine
   - There must be only a finite number of steps
   - Each step must be unambiguous and can occur in a finite amount of time
   - Must complete with an accept whenever the input to the machine meets the criteria.

Problem Solving

3. Can a Turing Machine ever write the blank sysmbol onto its tape? (3 pts.)

Yes.

4. Can the tape alphabet Γ be the same as the input alphabet Σ? Explain why or why not. (3 pts.)

No becuase blank is always in Γ but never in Σ.

5. Can a Turing Machine's head ever be in the same location in two successive steps? Explain why or why not. (3 pts.)

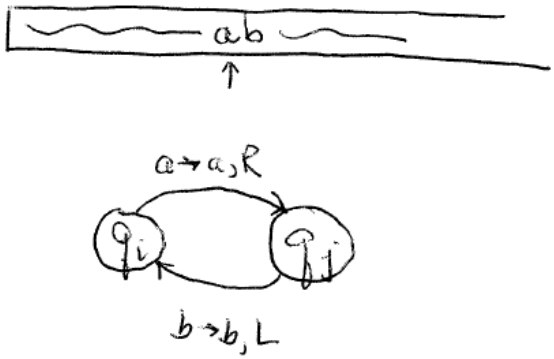
Yes, when the read/write head is on the left end of the tape, and a left move is requested.

6. Can a Turing Machine contain only a single state? Explain why or why not. (3 pts.)

No, it must at least have a q_{accept} and q_{reject}, which cannot be the same. One of these can be the start state.

7. Draw a sample tape, containing symbols, and state transitions, that demonstrates that a Turing Machine can loop forever. (3 pts.)

Sample answer:



8. Define a Turing machine that places a $ onto the front of a tape, shifting all non-blank symbols one position to the right, returning the read/write head to the position one place to the right of the $.
   - Let the language alphabet be Σ={a, b, c} and the tape alphabet be Γ={a, b, c, $, _}.
   - Define this machine completely. That is, give a high-level and detailed plan, and the TM itself.

High-level plan: (5 pts.)

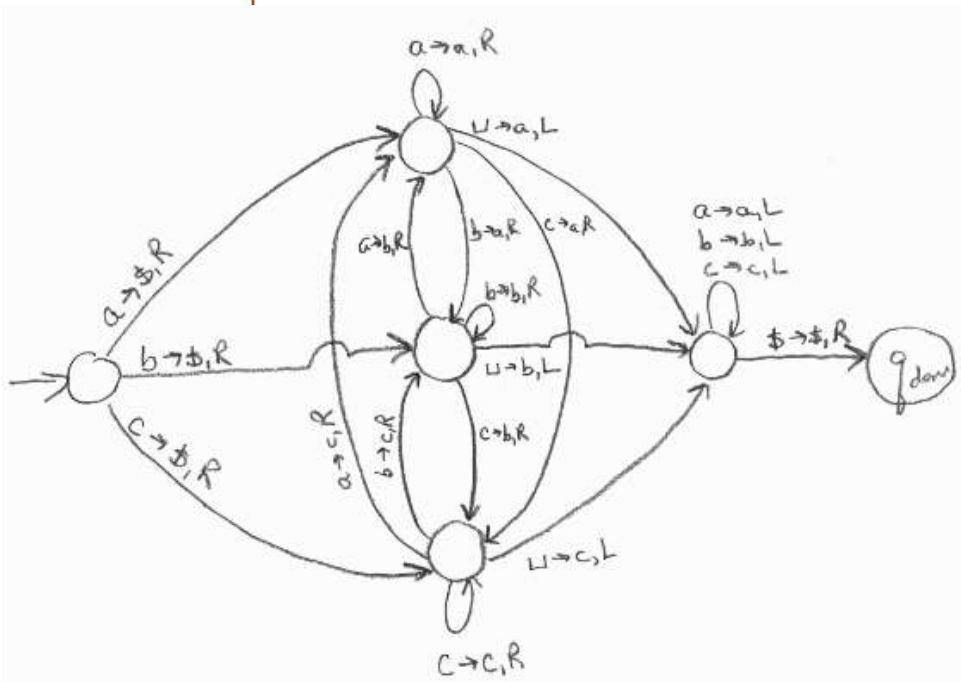
Write a $ in the first position, remembering the symbol that was there.
Continue doing this until a blank is reached. Write the final remembered symbol and return the read/write head back to the front of the tape.

Detailed plan: (5 pts.)

Read symbol and go to a state that remembers that symbol, writing the $.
Repeat until reach blank.
    Read symbol and go to a state that remembers that symbol, writing the previously remembered symbol.
Blank reached. Write remembered symbol.
Travel left across 'a's, 'b's and 'c's until reach $.
Travel right one position

Turing machine: (5 pts.)

Sample answer:



9. Consider a machine that is similar to a TM but has three possible moves, left, right or stay. Call such a machine a TM_{Stay}. Prove that a language is Turing-recognizable iff some Turing machine with stays recognizes it. (10 pts.)

A language is Turing-recognizable iff some TM_{Stay} recognizes it.

Proof:
⇒ (only if)
Suppose that a language is Turing-recognizable. Then some Turing Machine recognizes the language. This Turing machine can be considered a TM_{Stay} that never makes a stay move. Thus, this regular Turning machine is the TMStay that recognizes the language.

⇐ (if)
Suppose a language is recognized by a TM_{Stay}. Consider the following regular TM which:

1. When the TM_{Stay} makes a left or right move, the regular Turing machine does the same.
2. When the TM_{Stay} makes a 'stay' move, the regular Turing machine simulates that move, by doing the same writing but on a right move. It then does an immediate left move. (It must do the right move first, since if the read/write head was at the beginning of the tape, the left move would be ineffective.)

This TM will recognize the same language as TM_{Stay} recognized. Thus, when a language is recognized by a TM_{STay}, it is also Turing-recognizable.

10. Give a context-free grammar for the set of strings over the alphabet {a, b} that has twice as many a's as b's. (10 pts.)

$S \rightarrow SS \mid aSaSb \mid aSbSa \mid bSaSa \mid \varepsilon$

11. Define a push-down automaton (PDA) for the language L below , on $\Sigma=\{0,1\}$:

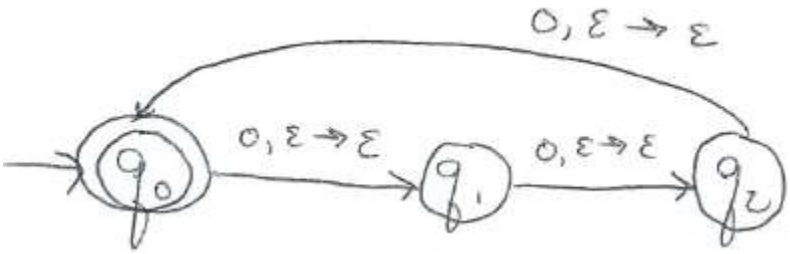
$L = \{ 0^n 0^{2n} \mid n \geq 0 \}$. (10 pts.)

Plan:

This language is the same as 0^{3n}, which is regular the push-down automaton can ignore the stack. Simply accept each time 3 0s are consumed.

Machine (PDA):

Possible answer:



12. Consider the language L = $\{w \mid n_a(w) \leq n_b(w) \leq n_c(w)\}$. Prove that L is context-free by creating a PDA or context-free grammar for it, or use the pumping lemma for context-free languages to prove that L is not context free. (10 pts.)

Claim L is not context-free.

Proof: Suppose, by way of contradiction, that L is context-free. Then the pumping lemma must hold for L. Let p be the pumping length. Consider the string $s = a^p b^p c^p$. Clearly $s \in L$ and $|s| \geq p$. The pumping lemma guarantees that s can be divided into five pieces s=uvxyz where $|vxy| \leq p$, $|vy| > 0$ and $s_i = uv^i xy^i z \in L$ for all $i \geq 0$.

Consider all possible breakdowns s=uvxyz where $|vxy| \leq p$, $|vy| > 0$.

Case 1: vxy is entirely in one set of symbols.
Case 2: vxy includes both a's and b'a, or both b'a and c's.

Consider each case separately.

Case 1: vxy is entirely in one set of symbols.
If vxy is entirely in the a's or entirely in the b's, pump up to get a string which has more a's or more b's than c's, so is not in L.

If vxy is entirely in the c's, pump down to get a string which has fewer c's than b's, so is not in the language.

Case 2: vxy includes both a's and b'a, or both b'a and c's.
If vxy includes both a's and b's, pump up and there will not be enough c's so the string won't be in the language.

If vxy includes both b's and c's, pump down and there will be too many a's so the string won't be in the language.

It has been shown that there is no way to divide s into uvxyz so the pumping lemma holds. Thus, the pumping lemma does not hold for L, and L must not have been context-free.