Homework 3 Questions

Problem 1: (20 points) Use CKY Algorithm to decide if $w = aaabbb$ is in $L(G)$ if the grammar for G is:

S	$\rightarrow$	$\boldsymbol{\varepsilon \mid AB \mid XB}$
T	$\rightarrow$	$AB \mid XB$
X	$\rightarrow$	AT
A	$\rightarrow$	a
B	$\rightarrow$	b

Completed CKY Triangular Table:

$X_{1,6}$ $\{S,T\}$					
$X_{1,5}$ $\{X\}$	$X_{2,6}$ $\emptyset$				
$X_{1,4}$ $\emptyset$	$X_{2,5}$ $\{S,T\}$	$X_{3,6}$ $\emptyset$			
$X_{1,3}$ $\emptyset$	$X_{2,4}$ $\{X\}$	$X_{3,5}$ $\emptyset$	$X_{4,6}$ $\emptyset$		
$X_{1,2}$ $\emptyset$	$X_{2,3}$ $\emptyset$	$X_{3,4}$ $\{S,T\}$	$X_{4,5}$ $\emptyset$	$X_{5,6}$ $\emptyset$	
$X_{1,1}$ {A}	$X_{2,2}$ {A}	$X_{3,3}$ {A}	$X_{4,4}$ {B}	$X_{5,5}$ {B}	$X_{6,6}$ {B}
a	a	a	b	b	b

Steps using CKY Algorithm:

- Row 1

$X_{i,j} = (X_{i,i}, X_{i+1,j})$		
$X_{1,2}$	$(X_{1,1}, X_{2,2})$ $\{A\},\{A\}$ $\{AA\}$	$\emptyset$
$X_{2,3}$	$(X_{2,2}, X_{3,3})$ $\{A\},\{A\}$ $\{AA\}$	$\emptyset$
$X_{3,4}$	$(X_{3,3}, X_{4,4})$ $\{A\},\{B\}$ $\{AB\}$	$\{S,T\}$
$X_{4,5}$	$(X_{4,4}, X_{5,5})$ $\{B\},\{B\}$ $\{BB\}$	$\emptyset$
$X_{5,6}$	$(X_{5,5}, X_{6,6})$ $\{B\},\{B\}$ $\{BB\}$	$\emptyset$

- Row 2

$X_{i,j} = (X_{i,i}, X_{i+1,j}) \cup (X_{i,i+1}, X_{i+2,j})$		
$X_{1,3}$	$(X_{1,1}, X_{2,3}) \cup (X_{1,2}, X_{3,3})$ $\{A\}\{\emptyset\} \cup \{\emptyset\}\{A\}$ $\{A\}$	$\emptyset$
$X_{2,4}$	$(X_{2,2}, X_{3,4}) \cup (X_{2,3}, X_{4,4})$ $\{A\}\{ST\} \cup \{\emptyset\}\{B\}$ $\{AS, AT, B\}$ $\{\emptyset, X, \emptyset\}$	$\{X\}$
$X_{3,5}$	$(X_{3,3}, X_{4,5}) \cup (X_{3,4}, X_{5,5})$ $\{A\}\{\emptyset\} \cup \{ST\}\{B\}$ $\{A, SB, TB\}$ $\{\emptyset, \emptyset, \emptyset\}$	$\emptyset$
$X_{4,6}$	$(X_{4,4}, X_{5,6}) \cup (X_{4,5}, X_{6,6})$ $\{B\}\{BB\} \cup \{BB\}\{B\}$ $\{BB\}$	$\emptyset$

- Row 3

$X_{i,j} = (X_{i,i}, X_{i+1,j}) \cup (X_{i,i+1}, X_{i+2,j}) \cup (X_{i,i+2}, X_{i+3,j})$		
$X_{1,4}$	$(X_{1,1}, X_{2,4}) \cup (X_{1,2}, X_{3,4}) \cup (X_{1,3}, X_{4,4})$ $\{A\}\{X\} \cup \{\emptyset\}\{ST\} \cup \{\emptyset\}\{B\}$ $\{AX, S, T, B\}$ $\{\emptyset, \emptyset, \emptyset, \emptyset\}$	$\emptyset$
$X_{2,5}$	$(X_{2,2}, X_{3,5}) \cup (X_{2,3}, X_{4,5}) \cup (X_{2,4}, X_{5,5})$ $\{A\}\{\emptyset\} \cup \{\emptyset\}\{\emptyset\} \cup \{X\}\{B\}$ $\{A, XB\}$ $\{\emptyset, \{S, T\}\}$	$\{S, T\}$
$X_{3,6}$	$(X_{3,3}, X_{4,6}) \cup (X_{3,4}, X_{5,6}) \cup (X_{3,5}, X_{6,6})$ $\{A\}\{\emptyset\} \cup \{ST\}\{\emptyset\} \cup \{\emptyset\}\{B\}$ $\{A, S, T, B\}$ $\{\emptyset, \emptyset, \emptyset, \emptyset\}$	$\emptyset$

- Row 4

$X_{i,j} = (X_{i,i}, X_{i+1,j}) \cup (X_{i,i+1}, X_{i+2,j}) \cup (X_{i,i+2}, X_{i+3,j}) \cup (X_{i,i+3}, X_{i+4,j})$		
$X_{1,5}$	$(X_{1,1}, X_{2,5}) \cup (X_{1,2}, X_{3,5}) \cup (X_{1,3}, X_{4,5}) \cup (X_{1,4}, X_{5,5})$ $\{A\}\{ST\} \cup \{\emptyset\}\{\emptyset\} \cup \{\emptyset\}\{\emptyset\} \cup \{\emptyset\}\{B\}$ $\{AS, AT, B\}$ $\{\emptyset, X, \emptyset\}$	$\{X\}$

$X_{2,6}$	$(X_{2,2},\ X_{3,6}) \cup (X_{2,3},\ X_{4,6}) \cup (X_{2,4},\ X_{5,6}) \cup (X_{2,5},\ X_{6,6})$ $\{A\}\{\emptyset\} \cup \{\emptyset\}\{\emptyset\} \cup \{X\}\{\emptyset\} \cup \{ST\}\{B\}$ $\{A, X, SB, TB\}$ $\{\emptyset, \emptyset, \emptyset, \emptyset\}$	$\emptyset$

- Row 5

$X_{i,j} = (X_{i,i},\ X_{i+1,j}) \cup (X_{i,i+1},\ X_{i+2,j}) \cup (X_{i,i+2},\ X_{i+3,j}) \cup (X_{i,i+3},\ X_{i+4,j}) \cup (X_{i,i+4},\ X_{i+5,j})$

$X_{1,6}$	$(X_{1,1},\ X_{2,6}) \cup (X_{1,2},\ X_{3,6}) \cup (X_{1,3},\ X_{4,6}) \cup (X_{1,4},\ X_{5,6}) \cup (X_{1,5},\ X_{6,6})$ $\{A\}\{\emptyset\} \cup \{\emptyset\}\{\emptyset\} \cup \{\emptyset\}\{\emptyset\} \cup \{\emptyset\}\{\emptyset\} \cup \{X\}\{B\}$ $\{A,\ XB\}$ $\{\emptyset, \{S, T\}\}$	$\{ST\}$

Conclusion:

Using the CKY algorithm and triangular table, I was able to prove that the string $w = aaabbb$ is in $L(G)$. After successfully populating the table, I can verify $X_{1,6} = \{S, T\}$. Since the start symbol, $S \in X_{1,6}$, then $w \in L(G)$.

Problem 2: (20 points) In each section below, give the sequence of configurations that Turing Machine M_1 enters when started on the indicated input string.

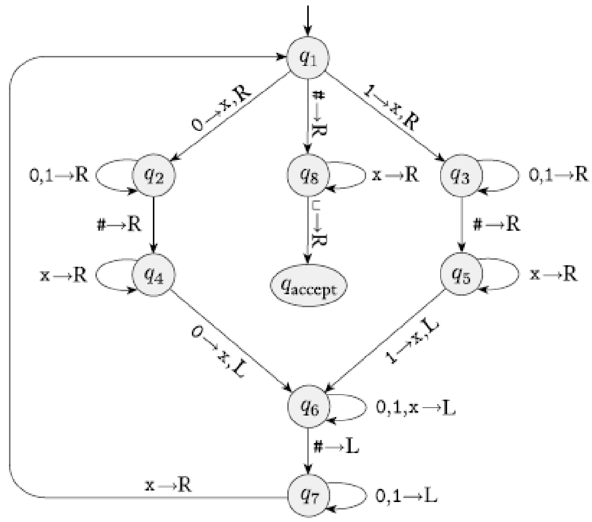
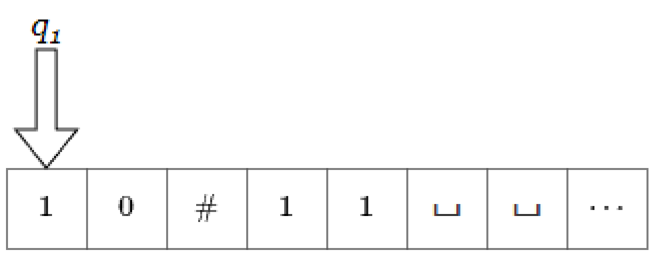


FIGURE 3.10
State diagram for Turing machine M_1

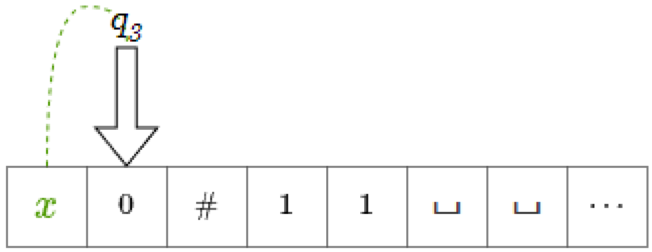
TM M_1 7-Tuple $(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$

$Q = \{q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_a, q_r\}$	Set of States
$\Sigma = \{0, 1, \#\}$	Input Alphabet
$\Gamma = \{0, 1, \#, x, \sqcup\}$	Tape Alphabet
q_1	Start State
q_{accept}	Set of Accept States
q_{reject}	Set of Reject States

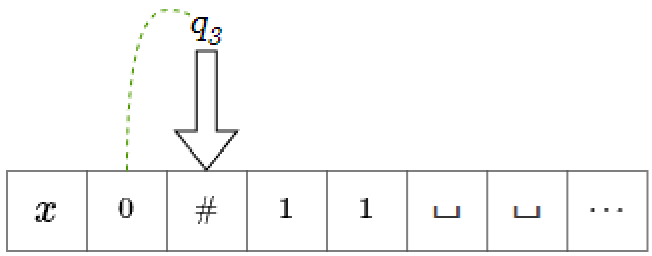
- 10#11 - Rejected
  - q_1 (starting state) q_1 10#11



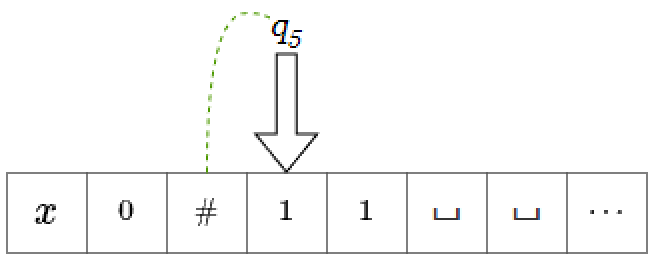
- $\delta(q_1, 1) = (q_3, x, R)$ $x\ q_3\ 0\#11$



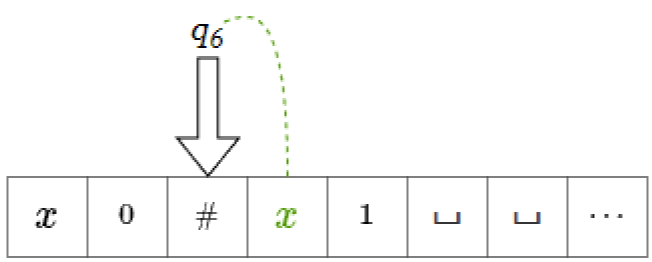
- $\delta(q_3, 0) = (q_3, -, R)$ $x0\ q_3\ \#11$



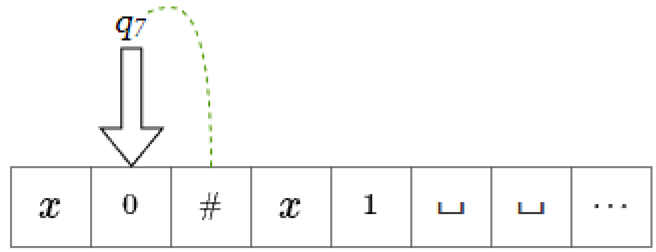
- $\delta(q_3, \#) = (q_5, -, R)$ $x0\#\ q_5\ 11$



- $\delta(q_5, 1) = (q_6, x, L)$ $x0\ q_6\ \#x1$

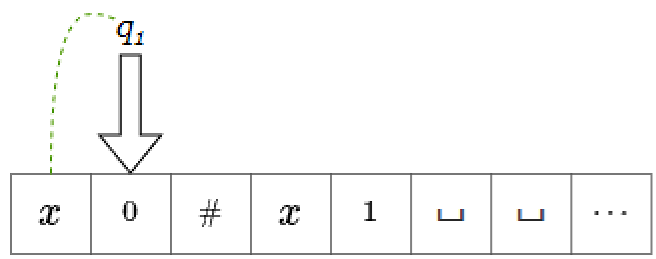


- $\delta(q_6, \#) = (q_7, -, L)$ $x\ q_7\ 0\#x1$

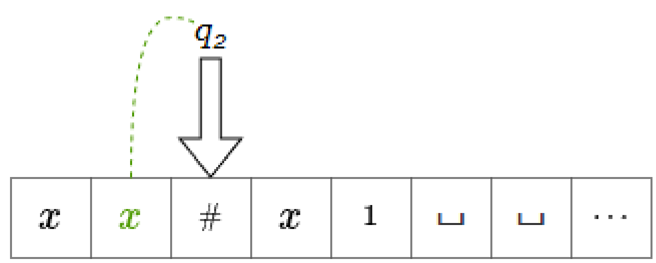


- $\delta(q_7, 0) = (q_7, -, L)$ $q_7\ x0\#x1$

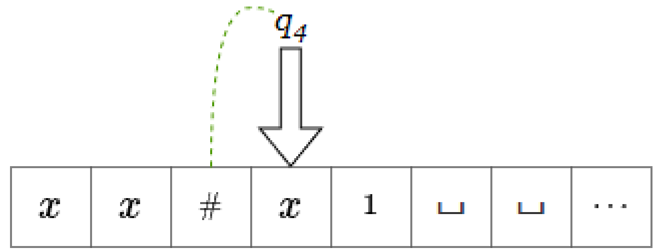
- $\delta(q_7, x) = (q_1, -, R)$ $x\ q_1\ 0\#x1$



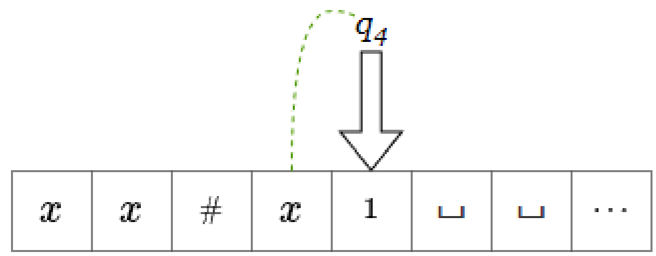
- $\delta(q_1, 0) = (q_2, x, R)$ $xx\ q_2\ \#x1$



- $\delta(q_2, \#) = (q_4, -, R)$ $xx\#\ q_4\ x1$

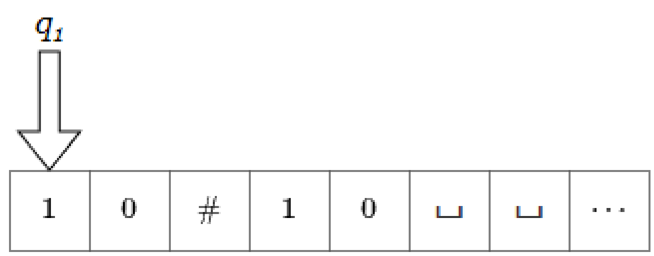


- $\delta(q_4, x) = (q_4, -, R)$ $xx\#x\ q_4\ 1$

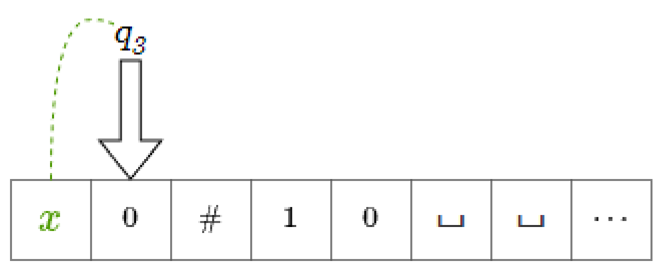


- $\delta(q_4, 1) =$ Halted, invalid input for q_4, Enters q_{reject} $xx\#x1\ q_{reject}$

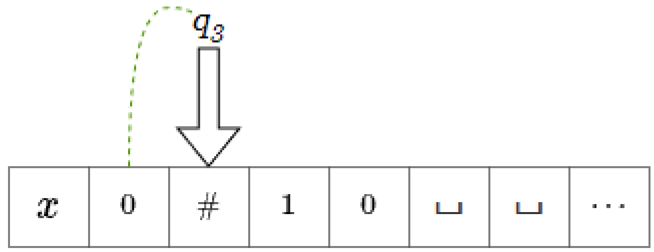
* 10#10 - Accepted
  - q_1 (starting state) $q_1\ 10\#10\sqcup$



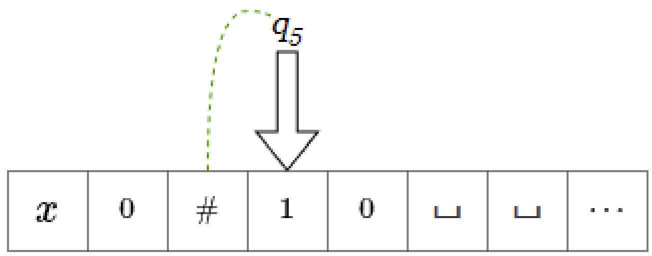
- $\delta(q_1, 1) = (q_3, x, R)$ $x\ q_3\ 0\#10\sqcup$



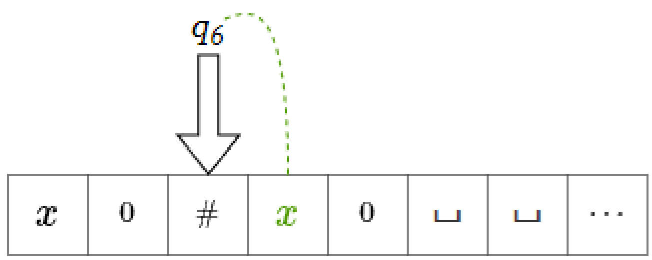
- $\delta(q_3, 0) = (q_3, -, R)$ $x0\ q_3\ \#10\sqcup$



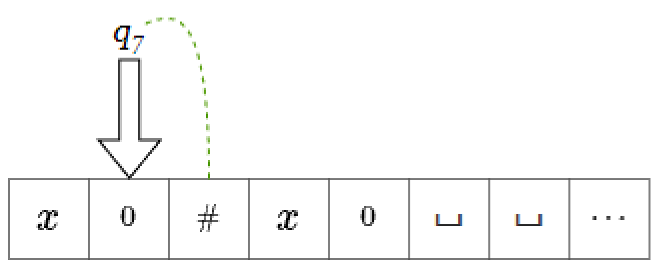
- $\delta(q_3, \#) = (q_5, -, R)$ $x0\#\ q_5\ 10\sqcup$



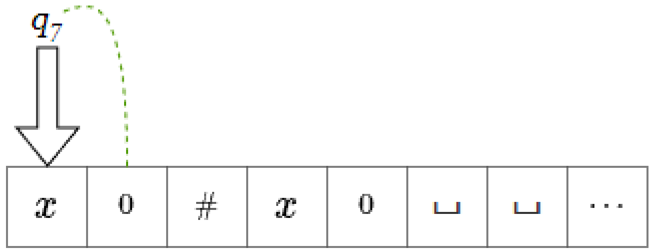
- $\delta(q_5, 1) = (q_6, x, L)$ $x0\ q_6\ \#x0\sqcup$



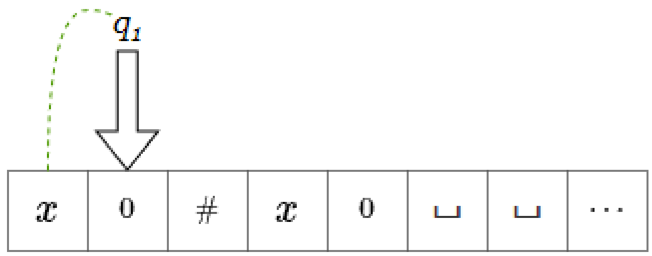
- $\delta(q_6, \#) = (q_7, -, L)$ $x\ q_7\ 0\#x0\sqcup$



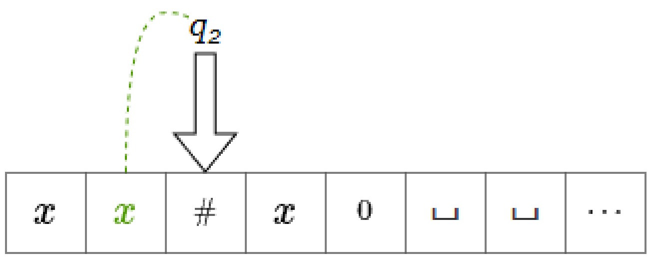
- $\delta(q_7, 0) = (q_7, -, L)$ $q_7\ x0\#x0␣$



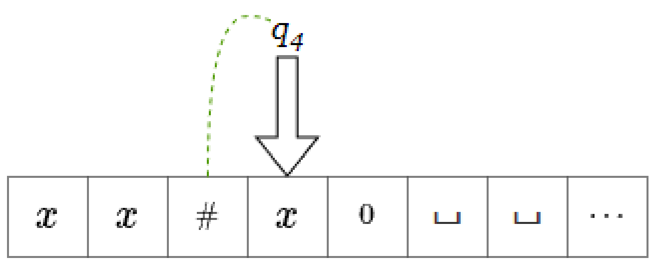
- $\delta(q_7, x) = (q_1, -, R)$ $x\ q_1\ 0\#x0␣$



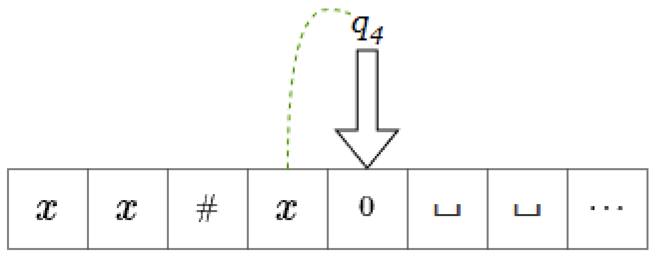
- $\delta(q_1, 0) = (q_2, x, R)$ $xx\ q_2\ \#x0␣$



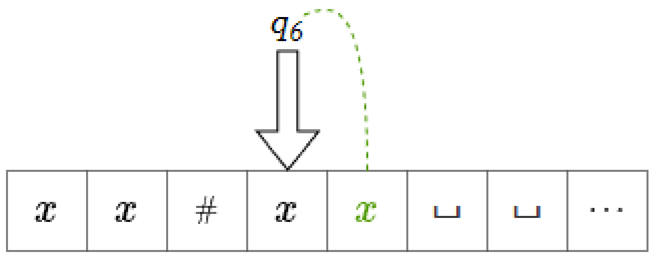
- $\delta(q_2, \#) = (q_4, -, R)$ $xx\#\ q_4\ x0␣$



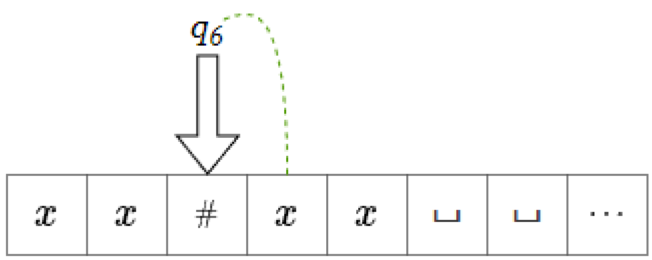
- $\delta(q_4, x) = (q_4, -, R)$ $xx\#x \; q_4 \; 0\sqcup$



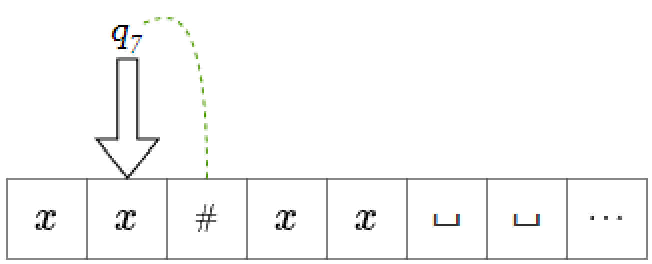
- $\delta(q_4, 0) = (q_6, x, L)$ $xx\# \; q_6 \; xx\sqcup$



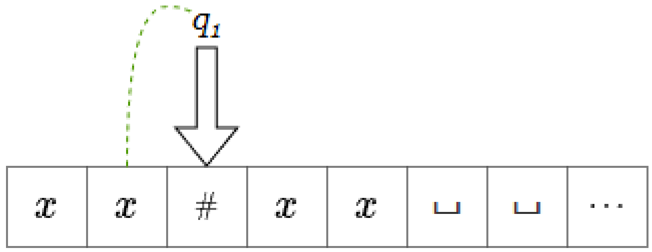
- $\delta(q_6, x) = (q_6, -, L)$ $xx \; q_6 \; \#xx\sqcup$



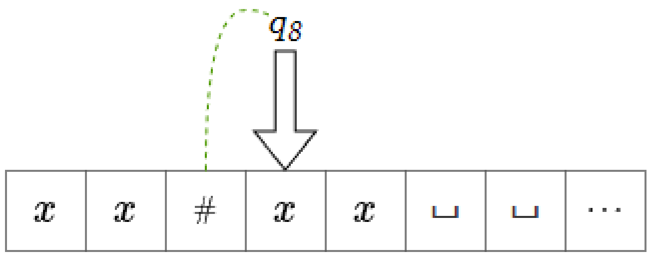
- $\delta(q_6, \#) = (q_7, -, L)$ $x \; q_7 \; x\#xx\sqcup$



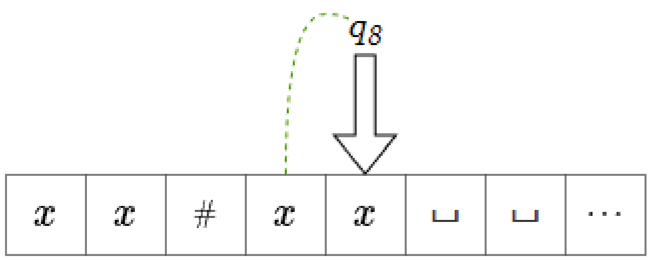
- ○ $\delta(q_7, x) = (q_1, -, R)$ $xx\ q_1\ \#xx\sqcup$



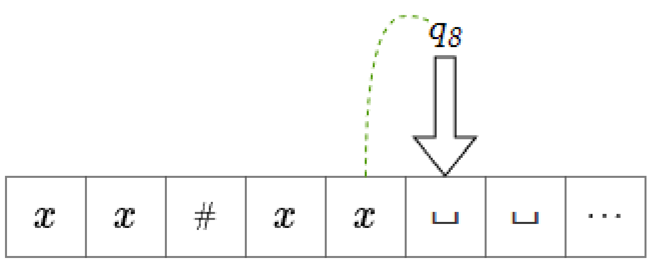
- ○ $\delta(q_1, \#) = (q_8, -, R)$ $xx\#\ q_8\ xx\sqcup$



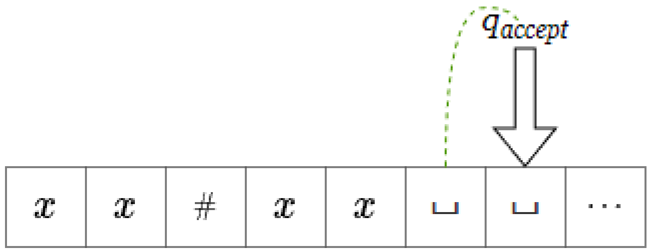
- ○ $\delta(q_8, x) = (q_8, -, R)$ $xx\#x\ q_8\ x\sqcup$



- ○ $\delta(q_8, x) = (q_8, -, R)$ $xx\#xx\ q_8\sqcup$



- $\delta(q_8, \sqcup) = (q_{accept}, -, R)$ $xx\#xx\sqcup\ q_{accept}$



Problem 3: (10 points) What is Turing recognizable? What is Turing decidable? What is the difference between them?

Language L is considered Turing Recognizable if there exists a Turing Machine M that determines if a given input string w is a member of the language or not by accepting and halting, rejecting and halting, or looping. Of these three possible outcomes, if $w \in L$, then M halts in q_{accept} and if $w \notin L$, M either halts in q_{reject} or enters a loop and does not halt. The language recognized by the Turing Machine is denoted as $L(M)$.

The terms Recursive Enumerability and Turing recognizability are often used interchangeably due to the Turing Machine adaptation, the enumerator. An enumerator is a Turing Machine with an attached printer where it produces, or enumerates, strings to an output tape. Language L is considered Turing Recognizable "if and only if some enumerator enumerates it." In other words, L is Turing Recognizable if and only if there exists a Turing Machine M that accepts and halts on all strings in L and loops or rejects all strings not in L.

However, this concept has its limitations. In some cases, it can be difficult to determine if a machine is looping or is still computing. This problem arises because there is no way to predict how long it will take for a machine to solve a problem. For example, if an enumerator's output tape does not contain string w, there is no possible way to distinguish if the machine is still processing or if w is not part of the language recognized by the machine.

Turing Decidability arises precisely to address this challenge. Language L is considered Turing Decidable if there exists a Turing Machine M that determines if a given input string w is a member of the language or not, halting in a state of accept or reject. Of these two possible outcomes, if $w \in L$, then M halts in q_{accept} and if $w \notin L$, M halts in q_{reject}.

The main difference between Turing Decidable and Turing Recognizable is that Turing Decidable machines will always halt and 'decide' a definitive answer on all inputs, rather than continuing to loop or process indefinitely. As a result, the concept of Turing Decidability can be seen as a stronger condition than Turing Recognizability as the former can determine whether an input belongs to a language or not, whereas the latter may not necessarily provide a definite answer for all cases.

It is important to note that the concept of Turing Decidability is a subset of Turing Recognizability, which means that every Turing decidable problem is also Turing recognizable. However, it is essential to note that not all problems are Turing decidable. Turing Recognizability is a broader concept that encompasses all problems that can be recognized by a Turing Machine, regardless of whether they can be decided by such machines or not. As such, it is crucial to understand the subtle differences between these two concepts and their implications in the field of computation theory.

Problem 4: (10 points) Answer each part for the following Language A

$$A = \{a^n b^n c^n | n \geq 0\}$$

- Is A a regular language?

By applying the pumping lemma for regular languages, I successfully proved that A is not a regular language. I began the proof under the assumption that A was a regular language and then identified a contradiction in the necessary conditions. As this assumption led to a contradiction, A does not satisfy the conditions required for a language to be considered regular.

Proof

If we assume that A is regular, the Pumping Lemma definition tells us that any string s in A can be 'pumped' at least a 'pumping length' of p , then divided into three pieces $s = xyz$. For it to be regular, it must satisfy the following conditions:

(1) $\forall i \geq 0, xy^i z \in A$
(2) $|y| \geq 0$
(3) $|xy| \leq p$

We can test these conditions by doing the following:

- Assume A is regular
- Let p be the pumping length
- Choose a string s in language A to test: $s = a^P b^P c^P$
  - As we assume A is regular, s is a member of A
  - Because $|s| = 3p$, and $3p \geq p$, then $|s| \geq p$
  - Since both statements above are true, the Pumping Lemma definition tells us that s can be split into three pieces $s = xyz$

- For conditions (2) $|y| \geq 0$ and (3) $|xy| \leq p$ to be met, piece y must contain only a's
  - For example, consider $p = 3$

$$a^P b^P c^P = a^3 b^3 c^3$$
$$a^3 b^3 c^3 = aaabbbccc$$

  - To split $s = xyz$, with $|xy| \leq 3, y$ may only contain a's

$$aaabbbccc$$
$$x\ y\ \ \ \ z$$

- According to condition (1) $\forall i \geq 0, xy^i z \in A$
  - assume $i = 2$

$$xy^i z \in A$$
$$xy^2 z \in A$$

  - Visual Representation

$$aaaaabbbccc$$

$x\ y\ y \quad z$

- $xy^2z \notin A$
  - However, there is no possible division of $aaaaabbbccc$ that will result in the required $a^n b^n c^n$ format. This is evident in the visual representation above as it has the format $a^5 b^3 b^3$ and $5 \neq 3$. As i increases, this also remains true since additional a's are placed at the beginning of the string, creating a further imbalance between a^n and $b^n c^n$ (which should be identical n's). Therefore, language A does not meet condition (1) for Pumping Lemma.

Since not all pumping lemma conditions are met, the assumption that A is regular is contradicted and proves that A is not regular.

Q.E.D

- Is A a context free language?

By applying the pumping lemma for context-free languages, I successfully proved that A is not a context free language. I began the proof under the assumption that A was a context free language and then identified a contradiction in the necessary conditions. As this assumption led to a contradiction, A does not satisfy the conditions required for a language to be considered context free.

<u>Proof</u>

The Context-Free Pumping Lemma definition tells us that any string s in A can be 'pumped' at least a 'pumping length' of p , then divided into five pieces $s = uvxyz$. For it to be regular, it must satisfy the following conditions:

(1) $\forall i \geq 0, xy^i z \in A$
(2) $|y| \geq 0$
(3) $|xy| \leq p$

We can test these conditions by doing the following:

- Assume A is a context-free language (CFL)
- Let p be the pumping length
- Choose a string s in language A to test: $s = a^P b^P c^P$
  - As we assume A is CFL, s is a member of A
  - Because $|s| = 3p$, and $3p \geq p$, then $|s| \geq p$
  - Since both statements above are true, the Pumping Lemma definition tells us that s can be split into five pieces $s = uvxyz$

- Case 1: v and x contain only one type of symbol
  - Testing conditions (2) $|vy| \geq 0$ and (3) $|vxy| \leq p$ under Case 1
    - assume $p = 3$

$$a^P b^P c^P = a^3 b^3 c^3$$
$$a^3 b^3 c^3 = aaabbbccc$$

- Split $s = uvxyz$, with $|vxy| \leq 3$

$$aaabbbccc$$
$$u \ \ vxy \ \ z$$
$$|vxy| = 3$$

- Testing condition (1) $\forall i \geq 0, uv^i xy^i z \in A$ under Case 1

  - assume $i = 2$

$$uv^i xy^i z \in A$$
$$uv^2 xy^2 z \in A$$

  - Visual Representation

$$aaabbbbbccc$$
$$u \ \ vvxyy \ \ z$$

  - $uv^i xy^i z \notin A$
    - There is no possible division of $aaabbbbbccc$ that will result in the required $a^n b^n \text{c}^n$ format. This is evident in the visual representation above which contains the format $a^3 b^5 c^3$ and $5 \neq 3$. As i increases, this remains true since additional b's are placed in the middle of the string, creating a further imbalance between the b^n and $\text{a}^n \ldots \text{c}^n$ (which should be equivalent n's). Therefore, language A does not meet condition (1) for Pumping Lemma when split in such a way that v and x contain only one type of symbol.

- Case 2: Either v or x contain more than one type of symbol
  - Testing conditions (2) $|vy| \geq 0$ and (3) $|vxy| \leq p$ under Case 2
    - assume $p = 5$

$$a^P b^P c^P = a^5 b^5 c^5$$
$$a^5 b^5 c^5 = aaaaabbbbbccccc$$

  - Split $s = uvxyz$, with $|vxy| \leq 5$

$$aaaaabbbbbccccc$$
$$u \quad v \ x \ y \quad z$$
$$|vxy| = 5$$

- Testing condition (1) $\forall i \geq 0, uv^i xy^i z \in A$ under Case 2
  - assume $i = 2$

$$uv^i xy^i z \in A$$
$$uv^2 xy^2 z \in A$$

- Visual Representation

$$aaaaababbbbbbbccccc$$
$$u \quad vv \quad x \quad yy \quad z$$

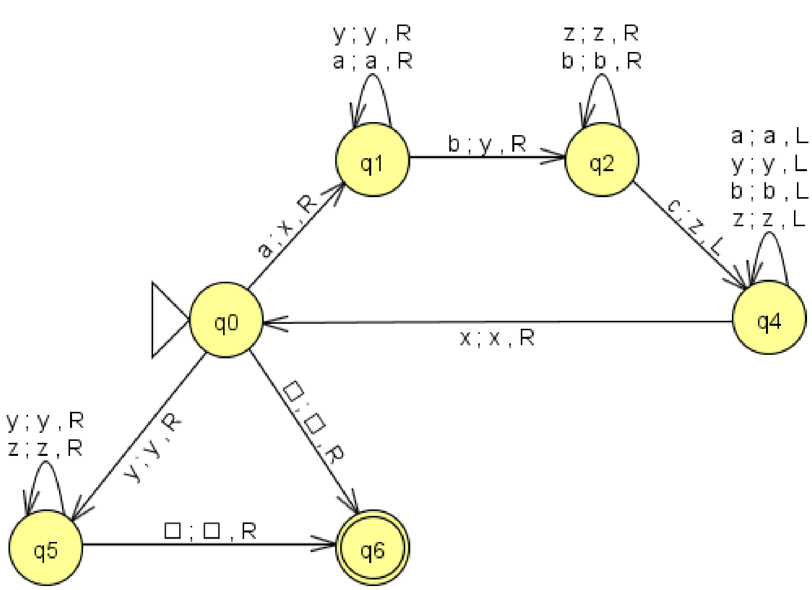
- $uv^ixy^iz \notin A$
    - There is no possible division of $aaaaababbbbbbbccccc$ that will result in the required $a^nb^nc^n$ format. This is evident in the visual representation above which contains the format $a^5b^1a^1b^7c^5$ as $a^nb^na^nb^nc^n \neq a^nb^nc^n$. As i increases, this will remain true since v contains more than one type of symbol; as v^i is pumped, the string places a's in the incorrect spot, disrupting the language format entirely. Therefore, language A does not meet condition (1) for Pumping Lemma when split in such a way that either v or x contain more than one type of symbol.

Since not all pumping lemma conditions are met in either Case 1 or Case 2, the assumption that A is a CFL is contradicted and proves that A is not context-free.

Q.E.D

- Is A Turing recognizable? Justify your answer.

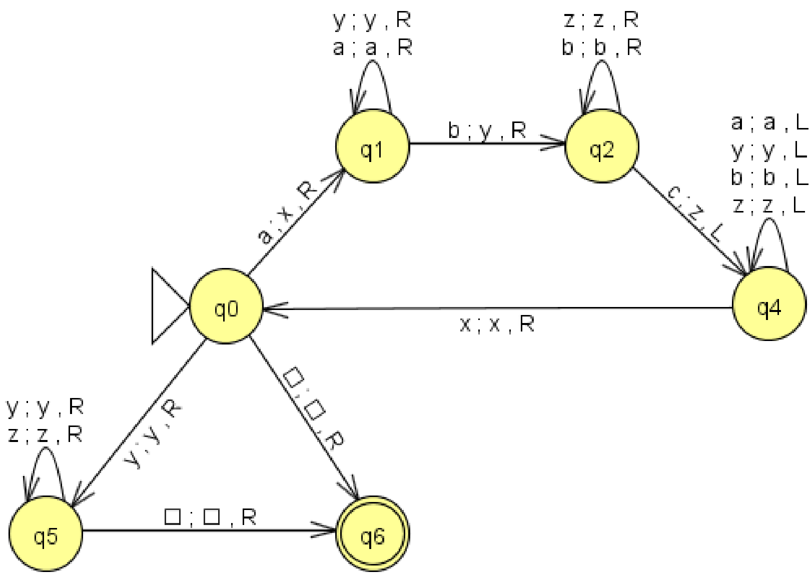
Yes, the language A is Turing recognizable as there exists a Turing machine M that accepts and halts on all input strings $w \in L$. An example of this Turing machine is below.



Input	Result
abc	Accept
aabbcc	Accept
aaabbbccc	Accept
	Accept
abcc	Reject
abbc	Reject
aabc	Reject
aabbc	Reject
abbcc	Reject
bc	Reject
ab	Reject
ac	Reject
cb	Reject
cba	Reject
bac	Reject

- Is A Turing decidable? Justify your answer.

Yes, the language A is Turing decidable as there exists a Turing machine M that accepts and halts on all input strings $w \in A$ and rejects all input strings $w \notin A$. As a reminder, Turing Decidability is a subset of Turing Recognizability. This implies that since machine M halts and produces a definitive answer for all inputs, it meets the conditions for Turing Decidability, and subsequently, Turing Recognizability. A copy of this Turing machine is below.



Input	Result
abc	Accept
aabbcc	Accept
aaabbbccc	Accept
	Accept
abcc	Reject
abbc	Reject
aabc	Reject
aabbc	Reject
abbcc	Reject
bc	Reject
ab	Reject
ac	Reject
cb	Reject
cba	Reject
bac	Reject

Problem 5: (20 points) Let $X = \{1, 2, 3, 4, 5\}$ and $Y = \{6, 7, 8, 9, 10\}$. Answer each part and give a reason for each negative answer for the functions $f: X \to Y$ and $g: X \to Y$

n	$f(n)$
1	6
2	7
3	6
4	7
5	6

n	$g(n)$
1	10
2	9
3	8
4	7
5	6

- Is f one-to-one?
  - No, function f is not one-to-one because multiple elements in X map to the same element in Y. For a function to be one-to-one, it must map each unique input from X (denoted as x_1 and x_2) to its own unique output ($f(x_1)$ and $f(x_2)$). In other words, $f(x_1) \neq f(x_2)$ when $x_1 \neq x_2$. However, function f violates this property because unique inputs 1, 3, and 5 produce the same output value 6, and unique inputs 2 and 4 produce the same output value 7. Thus, f is not one-to-one because it maps multiple inputs to the same output value.

- Is f onto?
  - No, function f is not onto because each element in Y is not mapped to by X. For a function to be onto, there must be at least one $x \in X$ for every $y \in Y$ such that $f(x) = y$. Thus, function f can not be onto as there are no input values which map to 8, 9, or 10.

- Is f a correspondence?
  - No, function f is not a correspondence. A correspondence is a function that is both one-to-one and onto; since f does not meet both of these criteria, then it cannot be a correspondence.

- Is g one-to-one?
  - Yes, function g is one-to-one because every element in X maps to a unique element in Y. Therfore, it satisfies the property that $f(X_1) \neq f(X_2)$ when $X_1 \neq X_2$.

- Is g onto?
  - Yes, function g is onto because every element in Y is mapped to by a unique element in X and there are no elements in Y that are left unmapped.

- Is g a correspondence?
  - Yes, function g is a correspondence. A correspondence is a function that is both one-to-one and onto; since g meets both of these criteria, it is a correspondence.

Problem 6: (20 points) Let $T = \{ (i, j, k) \mid i, j, k \in \mathbb{N}\}$. Show that T is countable.

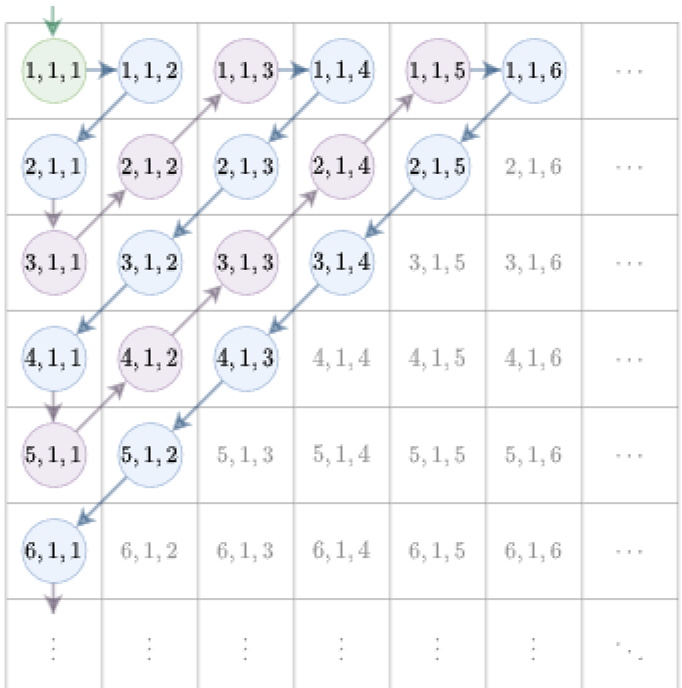
For a set to be countable, it must either be finite or it have the same size as $\mathbb{N}$. T, the set of all ordered triples (i, j, k) where i, j, and k are natural numbers, is clearly infinite as it contains every possible combination of i, j, and k. However, it is not immediately clear whether T is countable or uncountable. To determine this, we will use Cantor's method of size comparison, which is a useful tool for determining the relative sizes of infinite sets as it allows us to compare the sizes of sets that are too large to count directly. Cantor's method compares the cardinalities of two sets by establishing a one-to-one (injective) correspondence between their elements; if such correspondence exists between T and the set of natural numbers $\mathbb{N}$, then they have the same size.

1. Create a Matrix of T
   - To analyze the size of the set T, we need to consider all possible ordered triples (i, j, k) where $i, j, k \in \mathbb{N}$. To approach this methodically, we create a matrix of T to list all these possible ordered triples in a table format, where each row represents a specific value of the first index i. For example, the first row represents all ordered triples that start with 1, the second row represents all ordered triples that start with 2, and so on. By continuing the table in this manner, we ensure that every possible combination of of i, j, and k will be included and appear in a unique cell. Therefore, we can use the table below to analyze the size of the set T.

$1,1,1$	$1,1,2$	$1,1,3$	$1,1,4$	$1,1,5$	$1,1,6$	$\cdots$
$2,1,1$	$2,1,2$	$2,1,3$	$2,1,4$	$2,1,5$	$2,1,6$	$\cdots$
$3,1,1$	$3,1,2$	$3,1,3$	$3,1,4$	$3,1,5$	$3,1,6$	$\cdots$
$4,1,1$	$4,1,2$	$4,1,3$	$4,1,4$	$4,1,5$	$4,1,6$	$\cdots$
$5,1,1$	$5,1,2$	$5,1,3$	$5,1,4$	$5,1,5$	$5,1,6$	$\cdots$
$6,1,1$	$6,1,2$	$6,1,3$	$6,1,4$	$6,1,5$	$6,1,6$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

2. Establish a correspondence between T and the set of natural numbers $\mathbb{N}$
   - Using the matrix above, we can create a correspondence table to begin mapping each element of T to a natural number $\mathbb{N}$. As shown in the figure below, highlighted in green, our method will begin at the top left corner $\{1,1,1\}$ and traverse the matrix in a zig-zag pattern, assigning natural numbers to the elements of T as we progress. The arrows

demonstrate the specific route, which includes cells {1,1,2}, {2,1,1}, {3,1,1}, {2,1,2}, and so on. This approach ensures that every element of T will eventually be assigned to a natural number. If we were to start with just the first row of the matrix, we would never assign natural numbers to the elements in the subsequent rows, as there are infinitely many elements in the first row alone. The resulting correspondence table is also shown below.



$\mathbb{N}$	{i, j, k}
1	{1,1,1}
2	{1,1,2}
3	{2,1,1}
4	{3,1,1}
5	{2,1,2}
6	{1,1,3}
7	{1,1,4}
8	{2,1,3}
9	{3,1,2}
10	{4,1,1}
…	…

From the correspondence table above, it is clear that T is an injective function as every element in $\mathbb{N}$ maps to a unique element in T, and a surjective function as each element of T is assigned a unique natural number. Therefore, T is a bijection between $\mathbb{N}$ and T, and thus countable according to Cantor's method of size comparison. In conclusion, the zig-zag matrix above provides a bijective correspondence between T and the set of natural numbers $\mathbb{N}$, demonstrating that the set T is countable.