Question 1. [15 points] Show that the following language is not context-free using the Pumping lemma for context-free languages.

$$L = \{a^{n_1}b^{n_2}c^nb^{n_3}a^{n_4} : \exists n_1, n_2, n_3, n_4, n \in \mathbb{N} \text{ such that } n_1 = n_3, n_2 = n_4, n \geq 1\}$$

Opponent pick some $p \in \mathbb{N}$, $p > 0$

we pick $w \in L$, $w = a^p b^p c b^p a^p$, $4p+1 \geq p$

opponent pick

$\forall$: $w = \overbrace{aa\ldots aa}^{p}\overbrace{bb\ldots bb}^{p}c\overbrace{bb\ldots bb}^{p}\overbrace{aa\ldots aa}^{p}$

$\boxed{v}\boxed{x}\boxed{y}$ Case 1 $\quad\boxed{v}\boxed{x}\boxed{y}$ Case 2 $\quad\boxed{v}\boxed{x}\boxed{y}$ Case 3 $\quad\boxed{v}\boxed{x}\boxed{y}$ Case 4

$\boxed{\text{Case 1}}$ $v = a^{k_1}$ $y = a^{k_2}$ $1 \leq k_1 + k_2 \leq p$ $|vxy| \leq p$

pick i s.t. $w_i = u\underbrace{vvv\ldots v}_{i}x\underbrace{yyy\ldots y}_{i}z \notin L$

let $i = 2$ $a^{p+k_1+k_2}b^pcb^pa^p \notin L$

since $p+k_1+k_2 \geq p+1 > p$ ✗

$\boxed{\text{Case 2/3/4}}$ Similar argument as Case 1

$\forall$: $w = \overbrace{aa\quad aa}^{p}\overbrace{bb\ldots bb}^{p}c\overbrace{bb\ldots bb}^{p}\overbrace{aa\ldots aa}^{p}$

$\boxed{v}\boxed{x}\boxed{y}$ case 5 $\quad\boxed{v}\boxed{x}\boxed{y}$ case 6 $\quad\boxed{v}\boxed{x}\boxed{y}$ case 7

$\boxed{\text{Case 5}}$ $v = a^{k_1}$ $y = b^{k_2}$ $2 \leq k_1 + k_2 \leq p$ $k_1, k_2 \geq 1$ $|vxy| \leq p$

let $i = 2$ $w_2 = a^{p+k_1}b^{p+k_2}cb^pa^p \notin L$

$p+k_1 \geq p+1 > p$

$p+k_2 \geq p+1 > p$

therefore, this case is also bad ✗

$\boxed{\text{Case 6/7}}$ similar argument to Case 5

$\forall$: $w = \overbrace{aa\ \ \ \ aa}^{p}\overbrace{bb\ldots bb}^{p}\,c\,\overbrace{bb\ldots bb}^{p}\overbrace{aa\ldots aa}^{p}$

Case 12 $\boxed{v}\,\boxed{x}\,\boxed{y}$

$\boxed{v}\,\boxed{x}\,\boxed{y}$ Case 13

$\boxed{\text{Case 8}}$ $v = b^{k_1}$ $y = b^{k_2} c\, b^{k_3}$ $0 \le k_1 + k_2 + k_3 \le p-1$

$|vxy| \le p$ $k_1, k_2, k_3 \ge 0$

$$w_i = a^p (b^{k_1})^i\, b^{p-k_2-k_1} (b^{k_2} c\, b^{k_3})^i\, b^{p-k_3} a^p$$

let $i = 0 \Rightarrow a^p b^{p-k_2-k_1} b^{p-k_3} a^p \notin L$

missing "c" in the middle again! pattern incorrect.

$\boxed{\text{Case 9}}$ same idea as Case 12.

$\forall$: $w = \overbrace{aa\ldots aa}^{p}\overbrace{bb\ldots bb}^{p}\,c\,\overbrace{bb\ldots bb}^{p}\overbrace{aa\ldots aa}^{p}$

Case 14 $\boxed{v}\,\boxed{x}\,\boxed{y}$ Case 16 $\boxed{v}\,\boxed{x}\,\boxed{y}$

Case 15 $\boxed{v}\,\boxed{x}\,\boxed{y}$ Case 17 $\boxed{v}\,\boxed{x}\,\boxed{y}$

$\boxed{\text{Case 10}}$ $v = a^{k_1}$ $y = a^{k_2} b^{k_3}$ $k_2 + k_3 \ge 2$ $k_1 \ge 0$, $k_2, k_3 \ge 1$

$2 \le k_1 + k_2 + k_3 \le p$ $|vxy| \le p$

$$w_i = (a^{k_1})^i a^{p-k_1-k_2} (a^{k_2} b^{k_3})^i\, b^{p-k_3} c\, b^p a^p$$

let $i = 2$ $w_2 = a^{p+k_1} b^{k_3} a^{k_2} b^p c\, b^p a^p \notin L$

pattern incorrect! $a\ldots ab\ldots ba\ldots ab\ldots bcb\ldots ba\ldots a$

$\boxed{\text{Case 11/12/13}}$: Same idea as Case 14

Therefore by pumping lemma

this language is not context-free.

Question 2. [25 points] ***Read this carefully.*** Consider the following languages over the alphabet $\Sigma = \{a, b, c\}$.

$k < j$

$$L_1 = \{a^i b^j c^k : \exists i, j, k \in \mathbb{N} \text{ such that } i < j \text{ and } j > k\}$$

$$L_2 = \{a^i b^{j+n} c^k : \exists i, j, k, n \in \mathbb{N} \text{ such that } i + j < n + k\}$$

For each language decide whether it is

1. Regular.
2. Not regular, but context-free.
3. Not context-free.

If you answer "Regular" for a language, you must provide an FA (DFA, NFA, NFA+ε) which accepts it ***or*** a regular expression which describes it. If you answer "Not regular, but context-free", you must show that the language is not regular using any method/fact seen in class ***and*** provide ***either*** a CFG which generates it ***or*** a PDA (DPDA, NPDA) which accepts it. If you answer "Not context-free", you must provide a Pumping lemma proof. You do not need to prove the correctness of any of your constructions **but you should explain how they work**.

L1 This is not context free. with pumping lemma ✓

opponent picks $p \in \mathbb{N}$, $p > 0$

we pick $w \in L$ $a^p b^{p+1} c^p$ $|w| = 3p+1 \geq p$

opponent picks

$\boxed{\text{Case 1}}$

$\underbrace{aa \ldots aa}_{p}\ \underbrace{bb \ldots bb}_{p+1}\ \underbrace{cc \ldots cc}_{p}$

$\boxed{v}\,\boxed{x}\,\boxed{y}$ $\boxed{v}\,\boxed{x}\,\boxed{y}$ $\boxed{v}\,\boxed{x}\,\boxed{y}$

Case 1   case 3   case 2

let $v = a^{k_1}$ $y = a^{k_2}$ $1 \leq k_1 + k_2 \leq p$ $|vxy| \leq p$

$w_i = a^{p-k_1-k_2} (a^{k_1+k_2})^i b^{p+1} c^p$

let $i = 2$ $w_2 = a^{p+k_1+k_2} b^{p+1} c^p \notin L$ $p+k_1+k_2 \leq 2p \not< p$

$\boxed{\text{Case 2}}$ similar argument as case 1

$\boxed{\text{Case 3}}$ also similar, but instead we pump the string down instead of up like we did in case 1 and 2.

$\underbrace{a \ldots a}_{p}\ \underbrace{b \ldots b}_{p+1-(k_1+k_2)}\ \underbrace{c \ldots c}_{p}$ $p+1-(k_1+k_2) \leq p$ ✗

Case 4

$$\overbrace{aa\ldots aa}^{p}\,\overbrace{bb\ldots bb}^{p+1}\,\overbrace{cc\ldots cc}^{p}$$

case 4 $\boxed{v}\ \boxed{x}\ \boxed{y}$

$\boxed{v}\ \boxed{x}\ \boxed{y}$ case 5

let $v = a^{k_1}$ $y = b^{k_2}$ $\quad k_1, k_2 \geq 1 \quad 2 \leq k_1 + k_2 \leq p$

$$w_i = a^{p-k_1}(a^{k_1})^i\, b^{p+1-k_2}(b^{k_2})^i\, c^p$$

let $i = 0$ $\quad w_0 = a^{p-k_1}\, b^{p+1-k_2}\, c^p \notin L$

$$p+1-k_2 \leq p \quad \times$$

Case 5

Similar argument as Case 4

$$\overbrace{aa\ldots aa}^{p}\,\overbrace{bb\ldots bb}^{p+1}\,\overbrace{cc\ldots cc}^{p}$$

case 6 $\boxed{v}\ \boxed{x}\ \boxed{y}$ $\boxed{v}\ \boxed{x}\ \boxed{y}$ case 7

case 8 $\boxed{v}\ \boxed{x}\ \boxed{y}$ $\boxed{v}\ \boxed{x}\ \boxed{y}$ case 9

Case 6

$v = a^{k_1}$ $y = a^{k_2} b^{k_3}$ $\quad k_2, k_3 \geq 1 \quad 2 \leq k_1 + k_2 + k_3 \leq p \quad |vxy| \leq p$

$$w_i = (a^{k_1})^i a^{p-k_1-k_2} (a^{k_2} b^{k_3})^i\, b^{p+1-k_3}\, c^p$$

let $i = 2$ $\quad w_2 = a^{p+k_1-k_2}\, a^{k_2} b^{k_3} a^{k_2} b^{k_3}\, b^{p+1-k_3}\, c^p \notin L$

the pattern is incorrect $\quad a\ldots a\, b\ldots b\, a\ldots a\, b\ldots b\; c\ldots c.$

Case 7/8/9

same idea as above

for case 7 we expand y, case 8 expand v, 9 expand v

in order to make the pattern incorrect s.t. $\notin L$

therefore, by PL. L_1 is not context-free.

Question 2. [25 points] ***Read this carefully.*** Consider the following languages over the alphabet $\Sigma = \{a, b, c\}$.

$$L_1 = \{a^i b^j c^k : \exists i, j, k \in \mathbb{N} \text{ such that } i < j \text{ and } j > k\}$$

$k < j$

$$L_2 = \{a^i b^{j+n} c^k : \exists i, j, k, n \in \mathbb{N} \text{ such that } i + j < n + k\}$$

For each language decide whether it is

1. Regular.
2. Not regular, but context-free.
3. Not context-free.

If you answer "Regular" for a language, you must provide an FA (DFA, NFA, NFA+ε) which accepts it ***or*** a regular expression which describes it. If you answer "Not regular, but context-free", you must show that the language is not regular using any method/fact seen in class ***and*** provide ***either*** a CFG which generates it ***or*** a PDA (DPDA, NPDA) which accepts it. If you answer "Not context-free", you must provide a Pumping lemma proof. You do not need to prove the correctness of any of your constructions **but you should explain how they work**.

$\boxed{L_2}$ is not regular but context-free

to prove non-regular $\Rightarrow$ pumping lemma

$\forall$ opponent picks $p > 0$ $p \in \mathbb{N}$

$\exists$: $w = a^p c^{p+1} \in L$ $|w| = 2p+1 > p$

$\forall$: opponent decomposes into $w = xyz$ $|xy| \le p$ $|y| > 0$

$$W = \overbrace{aa \ldots a\, a}^{p}\ \overbrace{c \ldots cc}^{p+1}$$

$y = a^k$ $1 \le k \le p$

$\exists$ pick some $i \in \mathbb{N}$ s.t. $w_i = xy^iz$ and $w \notin L$

let $i = 2$

$$w_2 = xy^2z = a^{p-k} a^{2k} c^{p+1} = a^{p+k} c^{p+1} \notin L$$

$p+k \ge p+1$ $p+1 \nless p+1$ since $p+k$ can't be strictly less than $p+1$

Therefore according to PL, L_2 is not regular.

Now here is a CFG that generates this language

let $G = \{ V = \{S, B, C, X, Y\}, S, T = \{a, b, c\}, P \}$

$P =$

$$S \Rightarrow B \mid C \mid BC \mid X \mid Y$$
$$B \Rightarrow bB \mid b$$
$$C \Rightarrow cC \mid c$$
$$X \Rightarrow aXb \mid B$$
$$Y \Rightarrow aYc \mid aXC \mid C$$

The idea is that every time we add an "a", we must also add a "b" or "c". And since we will always finish by adding a "b" or a "c", we are guarantee s.t. the amount of "a"s is always less than the sum of the amount of "b"s + "c"s.

Note that we can always only have "b"s or "c"s or "b"s and "c"s without any "a"

Question 3. [30 points] Let Σ be a non-empty alphabet and $L \subseteq \Sigma^*$ be a context-free language. Which of the following languages is/are *necessarily* context-free? Note: For some string $w \in \Sigma^*$, w^R is the string reversal.

1. $\texttt{reverse\_1}(L) = \{xy : \exists x, y \in \Sigma^*, x \in L \text{ and } y^R \in L\}$

2. $\texttt{reverse\_2}(L) = \{x : \exists x \in \Sigma^*, x \in L \text{ and } x^R \in L\}$

If a language is *necessarily* context-free, prove your claim. Any construction you make does not require a proof of correctness.

If a language is *not necessarily* context-free, give a counterexample language L which is context-free but for which $\texttt{reverse\_i}(L)$ (`i = 1, 2`) is not context-free. To do so, you must show that your language L is context-free by giving either a PDA which accepts it or a CFG which generates it. You do not need to prove the correctness of your construction but you should explain how it works. In addition, you must give a Pumping lemma proof which shows that $\texttt{reverse}_i(L)$ is not context-free.

Before we start, lets first prove that CFL is actually closed under reversal.

In another world, if L is a CFL, the L^R is also CF.

Here is the claim:

Let $G(V, S, T, P)$ s.t. $L(G) = L$

for the reverse of this CFL, the gramar will simply be

$G^R(V, S, T, P^R)$ s.t. $L(G^R) = L^R$

For P^R, this is basically the "reverse" of each prod rule.

aka if $S \Rightarrow ab$ is a prod rule of G then
$S \Rightarrow (ab)^R$ is a prod rule of G^R

Take the CFL $L = \{a^n b^n, n \geq 0\}$

the prod rule is $S \Rightarrow aSb \mid \varepsilon$

and the reverse is $S \Rightarrow bSa \mid \varepsilon$

To prove the correctness we will need to use induction which I will omit here.

1) reverse_1(L) is necessarily context free

$reverse\_1(L) = \{xy : \exists x, y \in \Sigma^*, x \in L \text{ and } y^R \in L\}$

this is the same idea as

$$\{xy : \exists x, y \in \Sigma^*, x \in L \text{ and } y \in L^R\}$$

where L^R is the reversal, and L^R is also CF which we have proved on the last page

Now we can construct a CFG for reverse_1

let $G = (V, S, T, P)$ s.t. $L(G) = L$

$G^R = (V^R, S^R, T^R, P^R)$ s.t $L(G^R) = L^R$

$G^{reverse\_1} = (V', S', T', P')$ s.t. $L(G^{reverse\_1}) = L \cdot L^R$

$V' = V = V^R \quad S' = S' \quad T' = T = T^R$

$P' = \{(S' \to S S^R)\} \cup P \cup P^R$

we will skip the correctness proof for this CFG $\quad \square$

I think alternative way to prove this is that since we know CFG is closed under both reversal and concatenation, thus it is easy to see that $L \cdot L^R$ is also CF

assume that L is CF.

Question 3. [30 points] Let Σ be a non-empty alphabet and $L \subseteq \Sigma^*$ be a context-free language. Which of the following languages is/are *necessarily* context-free? Note: For some string $w \in \Sigma^*$, w^R is the string reversal.

1. `reverse_1`$(L) = \{xy : \exists x, y \in \Sigma^*, x \in L \text{ and } y^R \in L\}$
2. `reverse_2`$(L) = \{x : \exists x \in \Sigma^*, x \in L \text{ and } x^R \in L\}$

If a language is *necessarily* context-free, prove your claim. Any construction you make does not require a proof of correctness.

If a language is *not necessarily* context-free, give a counterexample language L which is context-free but for which `reverse_i`(L) (`i = 1, 2`) is not context-free. To do so, you must show that your language L is context-free by giving either a PDA which accepts it or a CFG which generates it. You do not need to prove the correctness of your construction but you should explain how it works. In addition, you must give a Pumping lemma proof which shows that `reverse`$_i(L)$ is not context-free.

(2) this is not necessarily context-free.

let $L_1 = \{a^n b^n c^m : n, m \in \mathbb{N}\}$

we know L_1 is CF, and here is its CFG

$G_1 = (V = \{S, X, C\}, S_1, T = \{a, b, c\}, P)$

$P \Rightarrow \quad S_1 \Rightarrow X_1 C_1$

$X_1 \Rightarrow aX_1b \mid \varepsilon \qquad$ s.t $L(G_1) = L_1$

$C_1 \Rightarrow cC_1 \mid \varepsilon$

let $L_2 = \{c^n b^n a^m : n, m \in \mathbb{N}\}$

by similar construction as L_1, L_2 is also context-free.

Lastly, let $L = L_1 \cup L_2$.

$G = (V, S, T, P) \quad V = V_1 = V_2 \quad S = S$

$T = T_1 = T_2 \quad P = \{(S \Rightarrow S_1 \mid S_2)\} \cup P_1 \cup P_2$

$L(G) = L$

Since both L_1 and L_2 are CF, and CFL is closed under Union, we know that L is also CF

However, reverse_2(L) is not necessary context-free.

for example the language $a^n b^n c^n \in$ reverse_2(L)

we can prove this is not regular with the pumping lemma.

$\forall$ opponent picks some $p \in \mathbb{N}$ $p > 0$

$\exists$ some $w = a^p b^p c^p \in$ reverse_2(L) $|w| = 3p \geq p$

$\forall$ opponent picks

$$\overbrace{aa\ldots aa}^{p}\,\overbrace{bb\ldots bb}^{p}\,\overbrace{cc\ldots cc}^{p}$$

$$\underbrace{\boxed{v}\ \boxed{x}\ \boxed{y}}_{\text{case 1}} \qquad \underbrace{\boxed{v}\ \boxed{x}\ \boxed{y}}_{\text{case 2}} \qquad \underbrace{\boxed{v}\ \boxed{x}\ \boxed{y}}_{\text{case 3}}$$

$\boxed{\text{Case 1}}$

$v = a^{k_1}$ $y = a^{k_2}$ $\quad 1 \leq k_1 + k_2 \leq p$ $\quad |vxy| \leq p$

$w_i = a^{p-k_1-k_2}(a^{k_1+k_2})^i b^p c^p$

let $i = 2$ $\quad w_2 = a^{p+k_1+k_2} b^p c^p \notin$ reverse_2(L)

$p + k_1 + k_2 \geq p + 1 \neq p$

$\boxed{\text{Case 2/3}}$ similar argument as Case 1

$$\overbrace{aa\ldots aa}^{p}\,\overbrace{bb\ldots bb}^{p}\,\overbrace{cc\ldots cc}^{p}$$

$$\underbrace{\boxed{v}\ \boxed{x}\ \boxed{y}}_{\text{case 4}} \qquad \underbrace{\boxed{v}\ \boxed{x}\ \boxed{y}}_{\text{case 5}}$$

$\boxed{\text{Case 4}}$ $v = a^{k_1}$ $y = b^{k_2}$ $\quad k_1, k_2 \geq 1$ $\quad 2 \leq k_1 + k_2 \leq p$ $\quad |vxy| \leq p$

$w_i = a^{p-k_1}(a^{k_1})^i b^{p-k_2}(b^{k_2})^i c^p$

let $i = 2$ $\quad w_2 = a^{p+k_1} b^{p+k_2} c^p \notin$ reverse_2(L)

$p + k_1 \neq p \quad p + k_2 \neq p$

$\boxed{\text{Case 5}}$ similar to case 4

Case 6

$$\underbrace{aa\ldots aa}_{p}\underbrace{bb\ldots bb}_{p}\underbrace{cc\ldots cc}_{p}$$

Case 6 $\boxed{v}\ \boxed{x}\ \boxed{y}$ $\boxed{v}\boxed{x}\boxed{y}$ Case 8

Case 7 $\boxed{v}\ \boxed{x}\boxed{y}$ $\boxed{v}\boxed{x}\boxed{y}$ Case 9

$|vxy| \le p$

$v = a^{k_1}$ $y = a^{k_2}b^{k_3}$ $k_2, k_3 \ge 1$ $2 \le k_1+k_2+k_3 \le p$

$$w_i = a^{p-k_1-k_2}(a^{k_1})^i (a^{k_2}b^{k_3})^i b^{p-k_3} c^p$$

at $i=2$ $w_2 = a^{p+k_1} a^{k_2} b^{k_3} a^{k_2} b^{k_3} b^{p-k_3} c^p$

$= a^{p+k_1} b^{k_3} a^{k_2} b^p c^p \notin$ reverse_2(L)

pattern mismatch

Case 7/8/9 similar idea as <u>Case 6</u>

case 7 we expand v
8 we expand y
9 we expand v

all pattern mismatch when we pump up.

by PL, reverse_2(L) is not context-free.

$\square$

Question 4. [15 points] Consider the following language

$$L = \{w\#w : w \in \{a, b\}^*\}$$

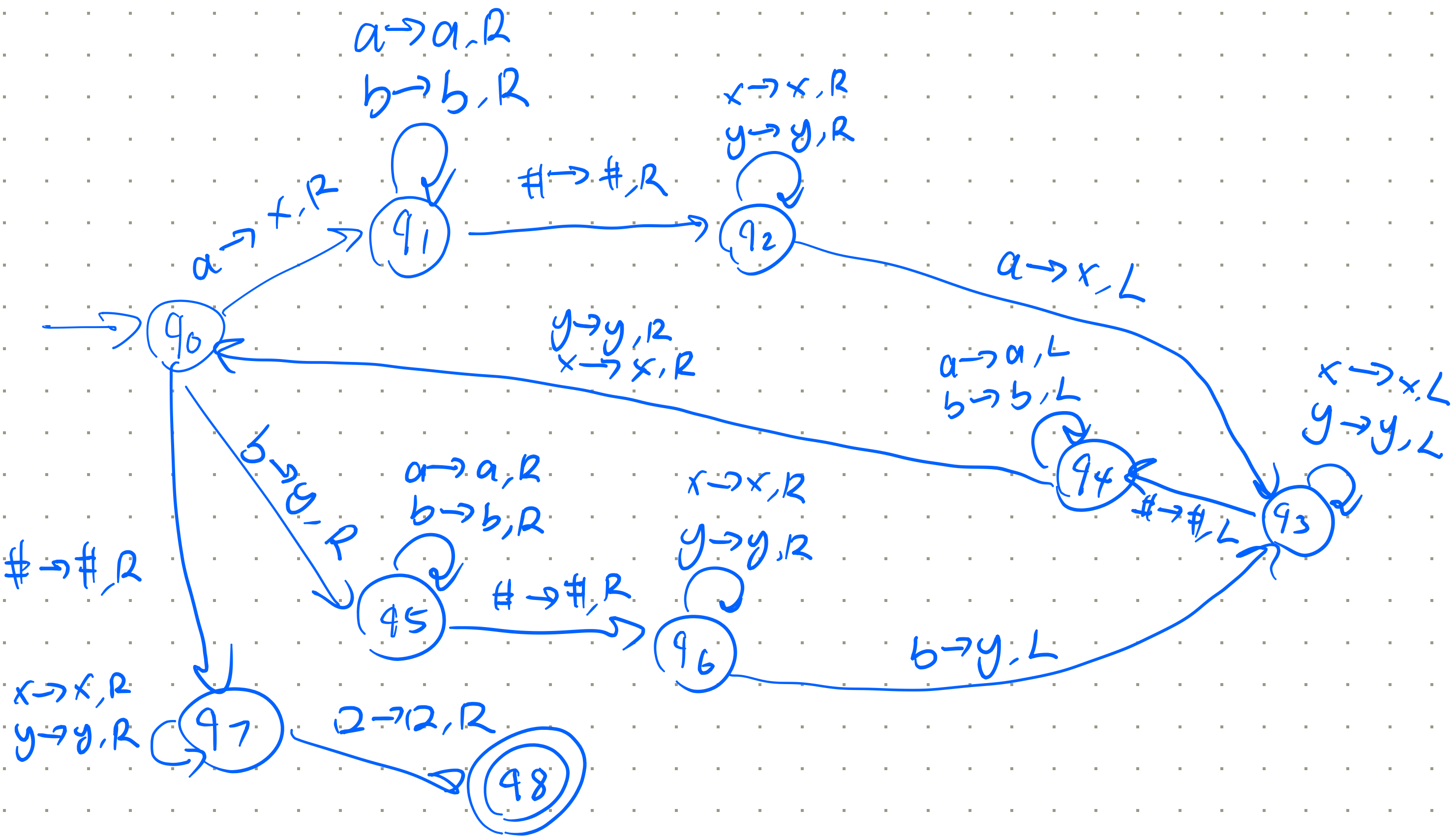
Show that this language is decidable by

1. Providing a high-level description of a deterministic semi-infinite tape Turing machine M which decides it.
2. Providing the state transition diagram of M.

You do not need to prove the correctness of your construction.

High Level Idea   Start at leftmost of the tape

1. if the pointer points to an "a" ("b"), mark it with "x" ("y") and move to the right. else if it sees "#", move R and go to #4
2. keep moving right pass "#" until we find the first "a" or "b". if this is not an "a" ("b"), we reject.
3. move left passing "#" until we find the left-most "x" or "y" and move right to go back to step #1
4. keep moving right and pass all the "x" and "y" if there are any. Lastly we will accept when it reaches the end.



Question 5. [15 points] Let $\Sigma = \{0\}$ and consider the following function $f : \Sigma^+ \to \Sigma^+$

$$f(x) = x \cdot x \cdot x$$

Show that f is a total computable function by

1. Providing a high-level description of a deterministic semi-infinite tape Turing machine M which computes f.
2. Providing the state transition diagram of M.

You do not need to prove the correctness of your construction.

High level idea: On input $x \in \Sigma^+$

1. if we have a "0" under the pointer, we mark it with "x" and move to the end (□) to write a "y". then move all the way back to the leftmost 0 and move one step right. repeat this step until the pointer no longer point to a "0" afterward.
2. the pointer will now be on a "y". now same idea as #1. we mark it with "x", move to the end to add a "z". the move all the way back to the leftmost y, and then move one to right. and repeat this step until we exhaust all "y"
3. now we should point to a "z", we move all the way to the end, then move towards the beginning and mark all "x" and "z" with 0 until we reach ⊢.