CSE 303 PRACTICE FINAL SOLUTIONS

FOR FINAL study Practice Final (minus PUMPING LEMMA and Turing Machines) and Problems from Q1 – Q4, Practice Q1 – Q4, and Midterm and Practice midterm. I will choose some of these problems for your FINAL TEST.

THE FINAL TEST will also contain YES/NO questions from the questions below, Q1 – Q4, Practice quizzes and Midterm and Practice Midterm. There will be more questions from the second part of the semester then from the first.

PART 1: Yes/No Questions Circle the correct answer. Write ONE-SENTENCE justification.

1. There is a set $A$ and an equivalence relation defined on $A$ that is an order relation with 2 Maximal elements.
   **Justify**: $A = \{a, b\}, R = " = "$
   y

2. $(ab \cup a^*b)^*$ is a regular language.
   **Justify**: this is a regular expression
   n

3. Let $\Sigma = \phi$, there is $L \neq \phi$ over $\Sigma$.
   **Justify**: $0^* = \{e\}$ and $L = \{e\} \subseteq \Sigma^*$
   y

4. $A$ is uncountable iff $|A| = c$ (continuum).
   **Justify**: $2^R$, $R$ real numbers, is uncountable and $|2^R| > c$
   n

5. There are uncountably many languages over $\Sigma = \{a\}$.
   **Justify**: $|\{a\}^*| = \aleph_0$ and $|2^{\{a\}}^*| = c$ and any set of cardinality $c$ is uncountable.
   y

6. Let $RE$ be a set of regular expressions. $L \subseteq \Sigma^*$ is regular iff $L = L(r), r \in RE$.
   **Justify**: definition
   y

7. $L^* = \{w \in \Sigma^* : \exists q \in F(s, w) \vdash^*_M (q, e)\}$.
   **Justify**: this is definition of $L(M)$, not $L^*$
   n
8. \((a^*b \cup \phi^*)\) is a regular expression.
   **Justify:** definition

9. \(\{a\}^*\{b\} \cup \{ab\}\) is a regular language
   **Justify:** it is a union of two regular languages, and hence is regular

10. Let \(L\) be a language defined by \((a^*b \cup ab)\), i.e. (shorthand) \(L = a^*b \cup ab\).
    Then \(L \subseteq \{a, b\}^*\).
    **Justify:** definition

11. \(\Sigma = \{a\}\), there are \(c\) (continuum) languages over \(\Sigma\).
    **Justify:** \(|2^{\{a\}^*}| = c\)

12. \(L^* = L^+ - \{e\}\).
    **Justify:** only when \(e \notin L\)

13. \(L^* = \{w_1 \ldots w_n, w_i \in L, i = 1, \ldots, n\}\).
    **Justify:** \(i = 0, 1, \ldots, n\)

14. For any languages \(L_1, L_2, L_3 \subseteq \Sigma^*\), if \(L_1 \subseteq L_2\), then \((L_1 \cup L_2)^* = L_2^*\).
    **Justify:** languages are sets

15. For any languages \(L_1, L_2 \subseteq \Sigma^*\), if \(L_1 \subseteq L_2\), then \((L_1 \cup L_2)^* = L_2^*\).
    **Justify:** languages are sets, so \((L_1 \cup L_2)^* = L_2^*\)

16. \(((\phi^* \cap a) \cup (\phi \cup b^*)) \cap \phi^*\) represents a language \(L = \{e\}\).
    **Justify:** \(((\{e\} \cap \{a\}) \cup \{b\}^*) \cap \{e\} = \{b\}^* \cap \{e\} = \{e\}\)

17. \(L = ((\phi^* \cup b) \cap (b^* \cup \phi))\) (shorthand) has only one element.
    **Justify:** \(\{e, b\} \cap \{b\}^* = \{e, b\}\)

18. \(L(M) = \{w \in \Sigma^*: (q, w) \xrightarrow{s} M (s, e)\}\).
    **Justify:** only when \(q \in F\)

19. If \(M\) is a FA, then \(L(M) \neq \phi\).
    **Justify:** take \(M\) with \(\Sigma = \phi\)

20. If \(M\) is a nondeterministic FA, then \(L(M) \neq \phi\).
    **Justify:** take \(M\) with \(\Sigma = \phi\) or \(F = \phi\)

2
21. \( L(M_1) = L(M_2) \) iff \( M_1 \) and \( M_2 \) are finite automata.
   **Justify:** take as \( M_1 \) any automata such that \( L(M_1) \neq \phi \) and \( M_2 \) such that \( L(M_2) = \phi \)

22. A language is regular iff \( L = L(M) \) and \( M \) is a deterministic automaton.
   **Justify:** \( M \) is a finite automata

23. If \( L \) is regular, then there is a nondeterministic \( M \), such that \( L = L(M) \).
   **Justify:** a finite automata

24. Any finite language is CF.
   **Justify:** any finite language is regular and \( RL \subset CFL \)

25. Intersection of any two regular languages is CF language.
   **Justify:** Regular languages are closed under intersection and \( RL \subset CFL \)

26. Union of a regular and a CF language is a CF language.
   **Justify:** \( RL \subseteq CFL \) and FCL are closed under union

27. \( L_1 \) is regular, \( L_2 \) is CF, \( L_1, L_2 \subseteq \Sigma^* \), then \( L_1 \cap L_2 \subseteq \Sigma^* \) is CF.
   **Justify:** theorem

28. If \( L \) is regular, there is a PDA \( M \) such that \( L = L(M) \).
   **Justify:** FA is a PDA operating on an empty stock

29. If \( L \) is regular, there is a CF grammar \( G \), such that \( L = L(G) \).
   **Justify:** \( RL \subseteq CFL \)

30. \( L = \{a^nb^nca^n : n \geq 0\} \) is CF.
   **Justify:** is not CF, as proved by Pumping Lemma for CF languages

31. \( L = \{a^nb^n : n \geq 0\} \) is CF.
   **Justify:** \( L = L(G) \) for \( G \) with \( R = \{S \rightarrow aSb|e\} \)

32. Let \( \Sigma = \{a\} \), then for any \( w \in \Sigma^*, w^Rw \in \Sigma^* \).
   **Justify:** \( a^R = a \) and \( w^R = w \) for \( w \in \{a\}^* \)

33. \( A \rightarrow Ax, A \in V, x \in \Sigma^* \) is a rule of a regular grammar.
   **Justify:** this is a rule of a left-linear grammar and we defined regular
grammar as a right-linear

34. Regular grammar has only rules \( A \to xA, A \to x, x \in \Sigma^*, A \in V - \Sigma \).
   **Justify**: not only, \( A \to xB \) for \( B \neq A \) is also a rule of a regular grammar

35. Let \( G = (\{S, (,), \}, \{(,), \}, R, S) \) for \( R = \{S \to SS | (S)\} \). \( L(G) \) is regular.
   **Justify**: \( L(G) = \emptyset \) and hence regular

36. The grammar with rules \( S \to AB, B \to b | bB, A \to e | aAb \) generates a language \( L = \{a^kb^j : k < j\} \).
   **Justify**: the rule \( A \to e | aAb \) produces the same amount of a’s and b’s, the rule \( B \to bB \) adds only b’s.
   More formally, let’s look at the derivations
   \[
   S \Rightarrow AB \Rightarrow ... \Rightarrow a^n b^n B \Rightarrow ... \Rightarrow a^n b^n b^k \Rightarrow a^n b^{n+k} \in L(G) \text{ and } n < n + k, \text{ and } a^n b^{n+1} \in L(G) \text{ and } n < n + 1
   \]
   we get \( a^n b^{n+k} \in L(G) \) and \( n < n + k \), and \( a^n b^{n+1} \in L(G) \) and \( n < n + 1 \)

37. \( L = \{w \in \{a, b\}^* : w = w^R\} \) is regular.
   **Justify**: we use Pumping Lemma; while pumping the string \( a^k b a^k \) with \( y \) containing only a’s we get that \( xy^2z \not\in L \)

38. We can always show that \( L \) is regular using Pumping Lemma.
   **Justify**: we use Pumping Lemma to prove (if possible) that \( L \) is not regular

39. \((p, e, \beta), (q, \gamma)\) \(\in \Delta \) means: read nothing, move from \( p \) to \( q \)
   **Justify**: and replace \( \gamma \) by \( \beta \) on the top of the stack

40. \( L = \{a^n b^m c^n : n, m \in N\} \) is CF.
   **Justify**: when \( n = m \) we get \( L = \{a^n b^n c^n : n \in N\} \) that is not CF

41. If \( L \) is regular, then there is a CF grammar \( G \), such that \( L = L(G) \).
   **Justify**: \( RL \subseteq CF \)

42. There is countably many non CF languages over \( \Sigma \neq \emptyset \)
   **Justify**: contradicts the fact that \( |\Sigma^*| = c \), i.e. is uncountable

43. Every subset of a regular language is a language.
   **Justify**: subset of a set is a set
44. A parse tree is always finite.
   **Justify:** derivations are finite.

45. Any regular language is accepted by some PD automata.
   **Justify:** $RL = FA, FA \subseteq PDA$.

46. Class of context-free languages is closed under intersection.
   **Justify:** $L_1 = \{a^n b^n c^m, n, m \geq 0\}$ is CF, $L_1 = \{a^n b^n c^n, n, m \geq 0\}$ is CF, but $L_1 \cap L_2 = \{a^n b^n c^n, n \geq 0\}$ is not CF.

47. There is countably many non-regular languages.
   **Justify:** contradicts the fact that $|\Sigma^*| = \mathfrak{c}$, i.e. is uncountable.

48. Every subset of a regular language is a regular language.
   **Justify:** $L = \{a^n b^n : n \geq 0\} \subseteq a^* b^*$ and $L$ is not regular.

49. A CF language is a regular language.
   **Justify:** $L = \{a^n b^n : n \geq 0\}$ is CF and not regular.

50. Class of regular languages is closed under intersection.
   **Justify:** theorem.

51. A regular language is a CF language.
   **Justify:** Regular grammar is a special case of a context-free grammar.

52. Every subset of a regular language is a regular language.
   **Justify:** $L_1 = a^n b^n$ is a non-regular subset of a regular language $L_2 = a^* b^*$.

53. Any regular language is accepted by some PD automata.
   **Justify:** Any regular language is accepted by a finite automata, and a finite automaton is a PD automaton (that never operates on the stock).

54. A parse tree is always finite.
   **Justify:** Any derivation of $w$ in a CF grammar is finite.

55. Parse trees are equivalence classes.
   **Justify:** represent equivalence classes.

56. For all languages, all grammars are ambiguous.
   **Justify:** programming languages are never inherently ambiguous.

57. A CF grammar $G$ is called ambiguous if there is $w \in L(G)$ with at least two distinct parse trees.
   **Justify:** definition.
58. A CF language \( L \) is inherently ambiguous iff all context-free grammars \( G \), such that \( L(G) = L \) are ambiguous.
   **Justify:** definition

59. Programming languages are sometimes inherently ambiguous.
   **Justify:** never

60. The largest number of symbols on the right-hand side of any rule of a CF grammar \( G \) is called called a fanout and denoted by \( \phi(G) \).
   **Justify:** definition

61. The Pumping Lemma for CF languages uses the notion of the fanout.
   **Justify:** condition on the length of \( w \in L \)

62. Turing Machines are as powerful as today’s computers.
   **Justify:** thesis

63. It is proved that everything computable (algorithm) is computable by a Turing Machine and vice versa.
   **Justify:** this is Church - Turing Hypothesis, not a theorem

64. Church’s Thesis says that Turing Machines are the most powerful.
   **Justify:** We adopt a Turing Machine that halts on all inputs as a formal notion of "an algorithm".

65. Turing Machines can read and write.
   **Justify:** by definition

66. A configuration of a Turing machine \( M = (K, \Sigma, \delta, s, H) \) is any element of a set \( K \times \Sigma^* \times (\Sigma^* (\Sigma - \{\#\}) \cup \{e\}) \), where \( \# \) denotes a blanc symbol.
   **Justify:** a configuration is an element of a set \( K \times \Delta \Sigma^* \times (\Sigma^* (\Sigma - \{\#\}) \cup \{e\}) \)

67. A computation of a Turing machine can start at any position of \( w \in \Sigma \).
   **Justify:** by definition

68. A computation of a Turing machine can start at any state.
   **Justify:** definition

69. In Turing machines, words \( w \in \Sigma^* \) can’t contain blanc symbols.
   **Justify:** \( \Sigma \) contains the blanc symbol

70. A Turing machine \( M \) decides a language \( L \subseteq \Sigma^* \), if for any word \( w \in \Sigma^* \) the following is true.
   
   If \( w \in L \), then \( M \) accepts \( w \); and if \( w \not\in L \) then \( M \) rejects \( w \).
   **Justify:** any word \( w \in \Sigma_0^* \), for \( \Sigma_0 = \Sigma - \{\#\} \)
QUESTION 1  Let $\Sigma$ be any alphabet, $L_1, L_2$ two languages over $\Sigma$ such that $e \in L_1$ and $e \in L_2$. Show that 
\[(L_1\Sigma^*L_2)^* = \Sigma^*\]

Solution: By definition, $L_1 \subseteq \Sigma^*$, $L_2 \subseteq \Sigma^*$ and $\Sigma^* \subseteq \Sigma^*$. Hence
\[(L_1\Sigma^*L_2)^* \subseteq \Sigma^*.\]

We have to show that also $\Sigma^* \subseteq (L_1\Sigma^*L_2)^*$. Let $w \in \Sigma^*$ we have that also $w \in (L_1\Sigma^*L_2)^*$ because $w = e^2 e$ and $e \in L_1$ and $e \in L_2$.

QUESTION 2 Use book or lecture definition (specify which are you using) to construct a non-deterministic finite automaton $M$, such that 
$L(M) = (ab)^*(ba)^*$. 

Draw a state diagram and specify all components $K$, $\Sigma$, $\Delta$, $s$, $F$ of $M$. Justify your construction by listing some strings accepted by the state diagram.

Solution 1 We use the lecture definition.

Components of $M$ are: $\Sigma = \{a, b\}$, $K = \{q_0, q_1\}$, $s = q_0$, $F = \{q_0, q_1\}$.

We define $\Delta$ as follows.
$\Delta = \{(q_0, ab, q_0), (q_0, a, q_1), (q_1, ba, q_1)\}$.

Strings accepted: $ab, abab, abba, ababba, ababbaba, ....$

Solution 2 We use the book definition.

Components of $M$ are: $\Sigma = \{a, b\}$, $K = \{q_0, q_1, q_2, q_3\}$, $s = q_0$, $F = \{q_2\}$.

We define $\Delta$ as follows.
$\Delta = \{(q_0, a, q_1), (q_1, b, q_0), (q_0, e, q_2), (q_2, b, q_3), (q_3, a, q_2)\}$.

Strings accepted: $ab, abab, abba, ababba, ababbaba, ....$

QUESTION 3 Given a Regular grammar $G = (V, \Sigma, R, S)$, where 
$V = \{a, b, S, A\}$, $\Sigma = \{a, b\}$,
$R = \{S \to aS | A | e, \ A \to abA | a | b\}$. 

7
1. Construct a finite automaton $M$, such that $L(G) = L(M)$.

**Solution** We construct a non-deterministic finite automata

$$M = (K, \Sigma, \Delta, s, F)$$

as follows:

- $K = (V - \Sigma) \cup \{f\}$, $\Sigma = \Sigma$, $s = S$, $F = \{f\}$
- $\Delta = \{(S, a, S), (S, e, A), (S, e, f), (A, ab, A), (A, a, f), (A, b, f)\}$

2. Trace a transitions of $M$ that lead to the acceptance of the string $aaaababa$, and compare with a derivation of the same string in $G$.

**Solution**

The accepting computation is:

$$(S, aaaababa) \vdash_{M} (S, aaababa) \vdash_{M} (S, aababa) \vdash_{M} (S, ababa) \vdash_{M} (A, ababa) \vdash_{M} (A, aba) \vdash_{M} (A, a) \vdash_{M} (f, e)$$

$G$ derivation is:

$$S \Rightarrow aS \Rightarrow aaS \Rightarrow aaaS \Rightarrow aaaS \Rightarrow aaaA \Rightarrow aaaaA \Rightarrow aaaaA \Rightarrow aaaaaba \Rightarrow aaaaaba$$

**QUESTION 4** Construct a context-free grammar $G$ such that

$$L(G) = \{w \in \{a, b\}^* : w = w^R\}.$$

Justify your answer.

**Solution** $G = (V, \Sigma, R, S)$, where

- $V = \{a, b, S\}$, $\Sigma = \{a, b\}$,
- $R = \{S \rightarrow aSa \mid bSb \mid a \mid b \mid e\}$.

**Derivation example:** $S \Rightarrow aSa \Rightarrow abSba \Rightarrow ababa$

$$ababa^R = ((ab)a(ba))^R = (ba)^Ra^Ra^R(ab)^R = ababa.$$

**Observation 1** We proved in class that for any $x, y \in \Sigma^*$, $(xy)^R = y^Rx^R$.

From this we have that

$$(xyz)^R = ((xy)z)^R = z^R(xy)^R = z^Ry^Rx^R$$
Grammar correctness justification: observe that the rules $S \rightarrow aS | bS | e$ generate the language $L_1 = \{ww^R : w \in \Sigma^*\}$. With additional rules $S \rightarrow a | b$ we get hence the language $L = L_1 \cup \{waw^R : w \in \Sigma^*\} \cup \{wbw^R : w \in \Sigma^*\}$. Now we are ready to prove that $L = L(G) = \{w \in \{a, b\}^* : w = w^R\}$.

Proof Let $w \in L$, i.e. $w = xx^R$ or $w = xax^R$ or $w = xbx^R$. We show that in each case $w = w^R$ as follows.

c1: $w^R = (xx^R)^R = (x^R)^R = x = w$ (used property: $(x^R)^R = x$).

c2: $w^R = (xax^R)^R = (x^R)^Ra^Rx^R = xax^R = w$ (used Observation 1 and properties: $(x^R)^R = x$ and $a^R = a$).

c3: $w^R = (xbx^R)^R = (x^R)^Rb^Rx^R = xbx^R = w$ (used Observation 1 and properties: $(x^R)^R = x$ and $b^R = b$).

QUESTION 5 Construct a pushdown automaton $M$ such that $L(M) = \{w \in \{a, b\}^* : w = w^R\}$

Solution 1 We define $M$ as follows: $M = (K, \Sigma, \Gamma, \Delta, s, F)$

$M$ components are

$K = \{s, f\}$, $\Sigma = \{a, b\}$, $\Gamma = \{a, b\}$, $F = \{f\}$

$\Delta = \{((s, a, e), (s, a)), ((s, b, e), (s, b)), ((s, e, e), (f, e)), (s, a, e), (f, a)),

((s, b, e), (f, b)), ((f, a, a), (f, e)), (f, b, b), (f, e))\}$

Trace a transitions of $M$ that lead to the acceptance of the string $ababa$.

Solution

$S \quad ababa \quad e$
$S \quad baba \quad a$
$S \quad aba \quad ba$
$f \quad ba \quad ba$
$f \quad a \quad a$
$f \quad e \quad e$
QUESTION 6  Construct a PDA $M$, such that

$$L(M) = \{b^n a^{2n} : n \geq 0\}.$$

**Solution** $M = (K, \Sigma, \Gamma, \Delta, s, F)$ for

- $K = \{s, f\}$,
- $\Sigma = \{a, b\}$,
- $\Gamma = \{a\}$,
- $s, F = \{f\}$,
- $\Delta = \{(s, b, e), (s, aa), ((s, c, e), (f, e)), ((f, a, a), (f, e))\}$

**Explain** the construction. Write motivation.

**Solution** $M$ operates as follows: $\Delta$ pushes $aa$ on the top of the stock while $M$ is reading $b$, switches to $f$ (final state) non-deterministically; and pops $a$ while reading $a$ (all in final state). $M$ puts on the stock two $a$’s for each $b$, and then remove all $a$’s from the stock comparing them with $a$’s in the word while in the final state.

**Trace** a transitions of $M$ that leads to the acceptance of the string $bbaaaa$.

**Solution** The accepting computation is:

\[
(s, bbaaaa, e) \vdash_M (s, ba, aa) \vdash_M (s, aaaa, aaaa) \\
\vdash_M (f, aaaa, aaaa) \vdash_M (f, aa, aa) \vdash_M (f, a, a) \vdash_M (f, e, e)
\]

**Solution 2** $M = (K, \Sigma, \Gamma, \Delta, s, F)$ for

- $K = \{s, f\}$,
- $\Sigma = \{a, b\}$,
- $\Gamma = \{b\}$,
- $s, F = \{f\}$,
- $\Delta = \{(s, b, e), (s, b), ((s, e, e), (f, e)), ((f, aa, b), (f, e))\}$

**QUESTION 7** Use PUMPING LEMMA to prove that $L = \{ww : w \in \{a, b\}^*\}$ in NOT regular. Consider ALL cases.

**Solution** Assume $L$ is regular, then by PM Lemma there is $k \geq 0$ such that the Condition holds for all $w \in L$. Take $w = a^kba^kb$. Observe that $|w| = 2k + 2 \geq k$, and so $|w| \geq k$. So there are $x, y, z \in \Sigma^*$, such that $y \neq e$, $w = xyz$ and $|xy| \leq k$.

Observe that $y$ can’t contain first (or the second) $b$. If $y = b$ then $x = a^k$ and $|xy| = k + 1 > k$. Argument for the second $b$, and any location between first and the second $b$ is the same. It proves that $x = a^j, y = a^i, z = a^m ba^kb$, for $i > 0, m \geq 0, j \geq 0$ and $j + i + m = k$.

By PM Lemma $xy^n z \in L$ for all $n \geq 0$. Consider $xy^2 z = a^j a^{2i} a^m ba^kb$. Observe that $xy^2 z \in L$ iff $j + 2i + m = k$. On the other hand we had that $j + i + m = k$, and it gives $2i = i$. This contradiction proves that $L$ is not regular.
Question 8 Use Pumping Lemma to prove that
\[ L = \{a^{n^2} : n \geq 0 \} \]
is not CF.

Solution look at the solutions to hmk 4.

QUESTION 9 Here is the definition:

Let \( L \subseteq \Sigma^* \). For any \( x, y \in \Sigma^* \) we define an equivalence relation on \( \Sigma^* \) as follows.
\[ x \approx_L y \text{ iff } \forall z \in \Sigma^* (xz \in L \iff yz \in L). \]

Let now \( L = (aab \cup ab)^* \).

FIND all equivalence classes of \( x \approx_L y \).

Write all definitions and show work.

Solution We evaluate the equivalence classes as follows.
\[ [e] = \{ y \in \Sigma^* : \forall z \in \Sigma^* (z \in L \iff yz \in L) \} = L. \]

Observe that the main operator of \( L \) construction is \( * \), hence \( yz \in L \) iff \( x, y \in L \).

\[ [a] = \{ y \in \Sigma^* : \forall z \in \Sigma^* (az \in L \iff yz \in L) \} = La. \]

Observe that \( az \in L \) iff \( z \in bL \) (\( z \) begins with \( b \)), or \( z \in aL \) (\( z \) begins with \( a \)). Let \( z \in bL \), hence when \( yz \in L \), we get that \( y \in Laa \) or \( y \in La \) (\( y \) ends with \( aa \), or \( a \)). But the case \( y \in Laa \) is impossible, as for \( y = aa(e \in L) \) we get \( \forall z \in \Sigma^* (az \in L \iff aaz \in L) \) what is not true for \( z = ab; aab \in L \) and \( aab \notin L \).

Let now \( z \in aL \) we get \( yz \in L \) iff \( y \in La \).

\[ [aa] = \{ y \in \Sigma^* : \forall z \in \Sigma^* (aaaz \in L \iff yz \in L) \} = Laa. \]

Observe that \( aaaz \in L \) iff \( z \in bL \) (\( z \) begins with \( b \)), and hence \( yz \in L \) iff \( y \in Laa \) or \( y \in La \) (\( y \) ends with \( aa \), or \( a \)). But the case \( y \in Laa \) is impossible, as for \( y = a \) we get \( \forall z \in \Sigma^* (aaaz \in L \iff az \in L) \) what is not true for \( z = ab \).

Now observe that \( bb \notin L, aaa \notin L \) and \( L \) can’t contain any word in which \( bb \) or \( aaa \) appear. So we evaluate, as the next step \([bb]\) and \([aa]\).

\[ [aaa] = \{ y \in \Sigma^* : \forall z \in \Sigma^* (aaaz \in L \iff yz \in L) \} \]
\[ [bb] = \{ y \in \Sigma^* : \forall z \in \Sigma^* (bbz \in L \Leftrightarrow yz \in L) \} \]

Observe that the statements: \( aaaz \in L, bbz \in L \) are false for all \( z \) and hence we are looking for \( y \in \Sigma^* \) such that the statement \( yz \in L \) is false for all \( z \in \Sigma^* \). So \( y \) is any word from \( \Sigma^* \) that must contain at least one appearance of \( aaa \) or \( bb \). It means that \( y \in \Sigma^* (aaa \cup bb) \Sigma^* \) and 

\[ [aaa] = [bb] = \Sigma^* (aaa \cup bb) \Sigma^*. \]

We have hence 4 equivalence classes:

\[ L, \ La, \ Laa, \ \Sigma^* (aaa \cup bb) \Sigma^*. \]

**Question 10** Show that the following language \( L \) is NOT CF.

\[ L = \{ w \in \{a, b, c\}^* : \text{all numbers of accurences of } a, b, c \text{ in } w \text{ are different} \} \]

**Solution** First we represent \( L \) as \( L = L_1 \cup L_2 \cup L_3 \), for \( L_1 = \{ w \in \{a, b, c\}^* : \#a \neq \#b \text{ in } w \} \) - CF;

\( L_2 = \{ w \in \{a, b, c\}^* : \#b \neq \#c \text{ in } w \} \) - CF;

\( L_3 = \{ w \in \{a, b, c\}^* : \#c \neq \#a \text{ in } w \} \) - CF;

and use the closure of CF languages under union.