1. ∀m ∈ N, choose ℓ = a^{m+1}b^mc^m ∈ L such that ℓ ≥ m. ∀u, v, x, y, z ∈ \{a, b, c\}^* such that ℓ = uvxyz, |vxy| ≤ m, and |vy| ≥ 1, we have the following cases:

- vy = a^+: uv^0xy^0z = a^{m'}b^mc^m /∈ L, where m' ≤ m
- vy = a^b+: uv^0xy^0z = a^{m'}b^{m''}c^m /∈ L, where m' ≤ m, m'' < m
- vy = b^+: uv^2xy^2z = a^{m+1}b^{m'}c^m /∈ L, where m' ≥ m + 1
- vy = b^c+: uv^2xy^2z = a^{m+1}b^{m''}c^{m'''} /∈ L, where m' ≥ m + 1, m'' ≥ m + 1
- vy = c^+: uv^2xy^2z = a^{m+1}b^mc^{m''} /∈ L, where m' ≥ m + 1

Thus, L = \{a^n b^j c^k : n > j, n > k\} is not context-free.

2. Let L' = \{w ∈ \{a, b\}^* : n_a(w) = n_b(w)\} and L'' = \{w ∈ \{a, b\}^* : w contains 'aab' as a string\}. We show that L' is context-free and L'' is regular:

- NPDA M' = (\{q_0, q_1\}, \{a, b\}, \{a, b, z\}, δ, q_0, z, \{q_1\}) accepts L', where δ:
  δ(q_0, \lambda, z) = (q_1, z),
  δ(q_0, a, z) = (q_0, az), δ(q_0, a, a) = (q_0, aa), δ(q_0, a, b) = (q_0, \lambda),
  δ(q_0, b, z) = (q_0, bz), δ(q_0, b, a) = (q_0, \lambda), δ(q_0, b, b) = (q_0, bb)

- NFA M'' = (\{q_0, q_1, q_2, q_3\}, \{a, b\}, δ, q_0, \{q_3\}) accepts L'', where δ:
  δ(q_0, a) = q_1, δ(q_0, b) = q_0, δ(q_1, a) = q_2, δ(q_1, b) = q_0,
  δ(q_2, a) = q_0, δ(q_2, b) = q_3, δ(q_3, a) = q_3, δ(q_3, b) = q_3

Thus, L = L' ∩ L'' is context-free by the closure property under regular intersection.
3. (a) \( L = \{ w \in \{a, b\}^* : n_a(w) = n_b(w) \} \)

(b) \( L = \{ a^n b^m a^{n+m} : n \geq 0, m \geq 1 \} \)
4. (a) \( f(x) = 3x \)

(b) \( f(x, y) = x/2 \) if \( x \) is even, and \( (x + 1)/2 \) if \( x \) is odd.

5. (a) \( L = \{ w_1w_2 \in \{a, b\}^* : |w_1| = |w_2|, w_1 \neq w_2 \} \)

   if (reading blank) **Reject**;

   **Left**:
   if (reading marked character or blank) move right;
   else move left, goto **Left**;
if (reading 'a') mark 'a' as '0', move right;
else if (reading 'b') mark 'b' as '1', move right;
else goto Match;

Right:
if (reading marked character or blank) move left;
else move right, goto Right;
if (reading 'a') mark 'a' as '0', move left, goto Left;
else if (reading 'b') mark 'b' as '1', move left, goto Left;
else Reject;

Match:
if (reading '0') unmark '0' as 'a', move left, goto A;
else if (reading '1') unmark '1' as 'b', move left, goto B;
else move right, goto Match;

A:
if (reading blank) move right;
else move left, goto A;

AL:
if (reading '0') unmark '0' as 'a', move right;
else if (reading '1') Accept;
else move right, goto AL;
if (reading unmarked character) Reject;
else move right, goto Next;

Next:
if (reading unmarked character) move right, goto Match;
else move right, goto Next;

B:
if (reading blank) move right;
else move left, goto B;

BL:
if (reading '0') Accept;
else if (reading '1') unmark '1' as 'b', move right;
else move right, goto BL;
if (reading unmarked character) Reject;
else move right, goto Next;
(b) \( L = \{ a^n b^m : m = n^2 \} \)

`Loop:`
- if (reading 'a') mark 'a' as 'A', move left;
- else if (reading 'A') move right, goto `Loop`;
- else if (reading blank) **Accept**;
- else **Reject**;

`Rewind:`
- if (reading blank) move right;
- else move left, goto `Rewind`;

`Mark:`
- if (reading 'a') mark 'a' as '0', move right;
- else if (reading 'A') mark 'A' as '1', move right;
- else if (reading '0' or '1') move right, goto `Mark`;
- else move left, goto `Unmark`;

`Blank:`
- if (reading blank) move left;
- else move right, goto `Blank`;
- if (reading 'b') erase 'b' to blank, move left, goto `Rewind`;
- else **Reject**;

`Unmark:`
- if (reading blank) move right, goto `Loop`;
- else if (reading '0') mark '0' as 'a', move left, goto `Unmark`;
- else mark '1' as 'A', move left, goto `Unmark`;

6. Turing Thesis establishes that Turing Machine is the mathematical model of
current computers. Algorithms, computations or anything which can be per-
formed by current computers are also can be performed by Turing Machines.
Although there are many other mathematical models for current computers,
even variants of Turing Machines, they are not more powerful than Turing Ma-
chines. Here, we consider the capability of computations not the efficiency of
computations.

7. Let \( M = (Q, \Sigma, \Gamma, \delta, q_0, \sqcup, F) \) be a standard Turing Machine. Turing Machine
\( M' = (Q, \Sigma, \Gamma \cup \{*, \}, \delta', q_0, \sqcup, F) \), where \(* \notin \Gamma\) and \( \forall \delta'(q_i, a) = (q_j, b, L/R), q_i, q_j \in Q, a, b \in \Gamma \cup \{\}, b \neq \sqcup, \), can simulate \( M \) as follows:

- \( \forall \delta(q_i, a) = (q_j, \sqcup, L/R), M' \) has \( \delta'(q_i, a) = (q_j, *, L/R) \)
- \( \forall \delta(q_i, \sqcup) = (q_j, a, L/R), M' \) has \( \delta'(q_i, \sqcup) = (q_j, a, L/R) \) and \( \delta'(q_i, \ast) = (q_j, a, L/R) \)
8. \textit{ND}:
   \begin{itemize}
   \item if (reading blank) \textbf{Reject};
   \item else move right;
   \item goto \textit{ND} non-deterministically;
   \end{itemize}

   if (reading blank) \textbf{Reject};

\textbf{Match}:
   \begin{itemize}
   \item if (reading 'a') mark 'a' as '0', move left, goto \textit{A};
   \item else if (reading 'b') mark 'b' as '1', move left, goto \textit{B};
   \item else if (reading marked character) move right, goto \textit{Match};
   \item else move left, goto \textit{MatchR};
   \end{itemize}

\textit{A}:
   \begin{itemize}
   \item if (reading blank) move right;
   \item else move left, goto \textit{A};
   \item if (reading 'a') erase 'a' to blank, move right, goto \textit{Right};
   \item else \textbf{Reject};
   \end{itemize}

\textit{Right}:
   \begin{itemize}
   \item if (reading masked character) move right, goto \textit{Match};
   \item else move right, goto \textit{Right};
   \end{itemize}

\textit{B}:
   \begin{itemize}
   \item if (reading blank) move right;
   \item else move left, goto \textit{B};
   \item if (reading 'b') erase 'b' to blank, move right, goto \textit{Right};
   \item else \textbf{Reject};
   \end{itemize}

\textit{MatchR}:
   \begin{itemize}
   \item if (reading '0') erase '0' to blank, move left, goto \textit{AR};
   \item else if (reading '1') erase '1' to blank, move left, goto \textit{BR};
   \item else if (reading blank) \textbf{Accept};
   \item else \textbf{Reject};
   \end{itemize}

\textit{AR}:
   \begin{itemize}
   \item if (reading blank) move right;
   \item else move left, goto \textit{AR};
   \item if (reading 'a') erase 'a' to blank, move right, goto \textit{RightR};
   \end{itemize}
else Reject;

RightR:
if (reading blank) move left, goto MatchR;
else move right, goto RightR;

BR:
if (reading blank) move right;
else move left, goto BR;
if (reading 'b') erase 'b' to blank, move right, goto RightR;
else Reject;

9. \[ \delta(q_1, a_1) = (q_1, a_1, R) : 1010101011 \]
\[ \delta(q_1, a_2) = (q_3, a_1, L) : 101101110101 \]
\[ \delta(q_3, a_1) = (q_2, a_2, L) : 1110101101101 \]
The complete encoding is 101010110010101101110101001110101101101

10. A Turing Machine is an algorithm for a specific proposal. However, an Universal Turing Machine is a programmable Turing Machine that can simulate other Turing Machines and perform different computations. Universal Turing Machine takes a description of a Turing Machine \( M \) and an input \( w \) of \( M \) as inputs, and performs \( M(w) \). Universal Turing Machine has a tape for description of \( M \), a tape for \( w \), and a tape for the runtime status simulated during performing \( M(w) \).