1. a) No. M does not accept 1011.
   b) Yes. M accepts 1000.
   c) No. Strings in A DFA must encode a DFA and a string.
   d) No. Strings in A regex must encode a regular expression.
   e) No. Strings in EQ DFA must encode two DFAs.
   f) No. \( L(M) \neq \emptyset \).

2. a) \( f \) is not onto: Y3 is not a square of any integer.
    b) \( f \) is one-to-one: For \( n_1, n_2 \in \mathbb{N} \):
       \( n_1^2 = n_2^2 \) implies \( n_1 = n_2 \).
    c) \( f \) is not a correspondence because it is not onto.
2. b) The following list contains all elements of $T$:

**Stage 1:** List all triples $(i, j, k)$ where

\[ i + j + k = 3 \quad \text{(only } (1, 1, 1)) \]

**Stage 2:** List all triples $(i, j, k)$ where

\[ i + j + k = 4 \quad \text{((1, 1, 2), (1, 2, 1), (2, 1, 1))} \]

**Stage $x$:** List all triples $(i, j, k)$ where

\[ i + j + k = x + 2 \]
3. For a CFG $G$ the pumping length $p$ can be effectively found. If $G$ has $n$ variables and longest production has length $m$, we can choose $p = m^{n+1} + 1$. Define

$$A = \{ w \in \Sigma^* \mid |w| \geq p \}.$$

Since $A$ is regular, $L(G) \cap A$ is context-free.

By the pumping lemma, $L(G)$ is infinite if $L(G) \cap A \neq \emptyset$.

Emptiness of the context-free language $L(G) \cap A$ can be decided because $E_{CFG}$ is decidable (Thm. 4.8)
4. We reduce \( A_{TM} \) to \( T \).

Suppose \( TM \ P \) decides \( T \). Construct following decider for \( A_{TM} \):

\[
D = \{ \text{On input } \langle M, w \rangle \text{ where } M \text{ is TM, } w \text{ is string} \}
\]

1. Construct \( TM \ X \) as follows

\[
X = \{ \text{On input } y \}
\]

1. Run \( M \) on \( w \).
2. If \( M \) accepts, accept.
   If \( M \) rejects, reject.

2. Run decider \( P \) on input \( \langle X, w \rangle \)
3. If \( P \) accepted, accept. If \( P \) rejected, reject.

\( D \) decides \( A_{TM} \) correctly because:

- \( M \) accepts \( w \) implies \( X \) accepts \( w^k \) for all \( k \)
- \( M \) does not accept \( w \) implies \( L(X) = \emptyset \)
5. \( L_{\text{blank}} = \{ \langle M \rangle \mid M \text{ is a TM that in some computation writes a blank symbol over a nonblank symbol} \} \)

We show that \( L_{\text{blank}} \) is undecidable by reducing \( E_TM \) to \( L_{\text{blank}} \). Assume that \( N \) is a TM that decides \( L_{\text{blank}} \). We construct the following TM for \( E_TM \):

\[ P = " \text{On input } \langle M \rangle : " \]

1. Construct the following TM \( M' \):

   \( M' \) is as \( M \) except that always when \( M \) writes a blank \( \lambda \), \( M' \) writes a new symbol \( \Delta \) instead. When reading \( \Delta \), \( M' \) simulates computation of \( M \) on \( \lambda \). Always when \( M \) has transition to "accept" state, \( M' \) simulates this with a subroutine that first writes a nonblank symbol and then returns to same place and writes a blank symbol there.

   /* end of \( M' \) description */

2. Run \( N \) on \( \langle M' \rangle \). If \( N \) accepts, reject. If \( N \) rejects, accept."

The machine \( M' \) simulates computations of \( M \), with the change that \( M' \) writes a blank symbol only when accepting (and thus it always writes a blank symbol over a nonblank one). Thus \( M' \in L_{\text{blank}} \iff L(M) \neq \emptyset \).
6. (a) False.

Let $M_0$ be a TM where $L(M_0) = \emptyset$. Now \{ $\langle M_0 \rangle$ \} $\subseteq E_{TM}$, \{ $\langle M_0 \rangle$ \} is decidable (because it is a finite set) and $E_{TM}$ is undecidable. ($E_{TM}$ can be encoded over \{0, 1\}).

(b) False.

We can encode $A_{TM}$ over binary alphabet $\Sigma$.

Now $A_{TM} \subseteq \Sigma^*$, $A_{TM}$ is undecidable and $\Sigma^*$ is decidable (because it is regular.)