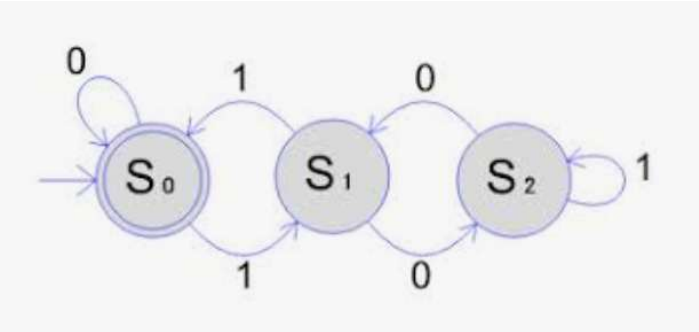
Exercise 1. Show the encoding of the following DFA.



- **States (Q):** Represented in unary encoding. $S_0 \to 0, S_1 \to 00, S_2 \to 000.$
- **Alphabet (Σ):** $a_1 = 0 \to 0, a_2 = 1 \to 00.$
- **Transition Function (δ):** $\delta(q_i, a_j) = q_k \to 0^{i+1}10^{j}10^{k+1}.$
- **Accept States (F):** $F = \{S_0\} \to 0.$
- **Overall Structure:** 11 [transitions separated by 1] 11 [accept states] 11.

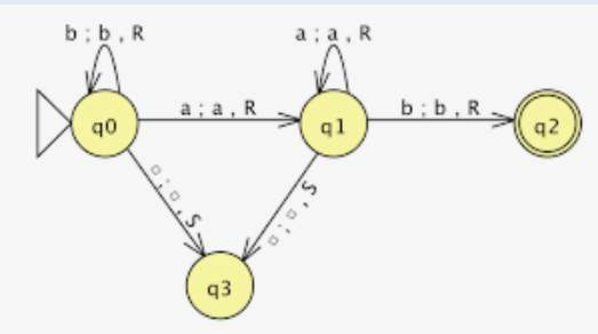
Transition Encodings:

1. $\delta(S_0, 0) = S_0 \Longrightarrow 0\,1\,0\,1\,0$
2. $\delta(S_0, 1) = S_1 \Longrightarrow 0\,1\,00\,1\,00$
3. $\delta(S_1, 0) = S_2 \Longrightarrow 00\,1\,0\,1000$
4. $\delta(S_1, 1) = S_0 \Longrightarrow 00\,1\,00\,1\,0$
5. $\delta(S_2, 0) = S_1 \Longrightarrow 000\,1\,0\,1\,00$
6. $\delta(S_2, 1) = S_2 \Longrightarrow 000\,1\,00\,1\,000$

Complete Encoding String:

11010101010010010010100010010010100010100100010010000110 11

Exercise 2. Show the encoding of the following Turing machine



Show a true instance of the problem ATM with the given Turing machine.

1. Encoding of the Turing Machine

Encoding Rules (Unit 9, slide 4):

- **States (Q):** $q_0 \to 0, q_1 \to 00, q_2 = q_{accept} \to 000, q_3 = q_{reject} \to 0000.$
- **Tape Alphabet (Γ):** $a_1 = a \to 0, a_2 = b \to 00, a_3 = \sqcup \to 000.$
- **Directions (Δ):** $L \to 0, R \to 00, S \to 000.$
- **Transition Entry:** $\delta(q_i, a_j) = (q_k, a_m, \Delta_t) \Longrightarrow 0^{i+1}10^{j}10^{k+1}10^{m}10^{t}.$
- Entries are separated by 11, and the entire machine description begins and ends with 111.

Transition Encodings:

1. $\delta(q_0, b) = (q_0, b, R) \Longrightarrow 0\,1\,00\,1\,0\,1\,00\,1\,00$
2. $\delta(q_0, a) = (q_1, a, R) \Longrightarrow 0\,1\,0\,1\,00\,1\,0\,1\,00$
3. $\delta(q_0, \sqcup) = (q_3, \sqcup, S) \Longrightarrow 0\,1\,000\,1\,0000\,1\,000\,1\,000$
4. $\delta(q_1, a) = (q_1, a, R) \Longrightarrow 00\,1\,0\,1\,00\,1\,0\,1\,00$
5. $\delta(q_1, b) = (q_2, b, R) \Longrightarrow 00\,1\,00\,1\,000\,1\,00\,1\,00$
6. $\delta(q_1, \sqcup) = (q_3, \sqcup, S) \Longrightarrow 00\,1\,000\,1\,0000\,1\,000\,1\,000$

Encoding of M:

1110100101001001101010010100110100010000100010001100101001010011001001000
1001001100100010000100010001111

2. True Instance of the Problem A_{TM}

- The Turing machine accepts any string matching the regular expression b^*aa^*b ..., such as $w = ab$.
- **True Instance:** $(\langle M \rangle, ab)$.
- Using the input encoding format for string $w = ab \rightarrow 0100$, the combined tape input instance is: ⟨M⟩1110100 (or formally stated as the pair $(\langle M \rangle, ab)$).

Exercise 3. Say that a variable A in CFL G is usable if it appears in some derivation of some string w ∈ G. Given a CFG G and a variable A, consider the problem of testing whether A is usable. Formulate this problem as a language and show that it is decidable.

1. Language Formulation:

$$USE_{CFG} = \{\langle G, A \rangle \mid G \text{ is a CFG and variable } A \text{ is usable in } G\}$$

A variable A is usable if and only if it satisfies two conditions:

1. A **is Generating:** $A \Rightarrow^* w$ for some terminal string $w \in \Sigma^*$.
2. A **is Reachable from** S**:** $S \Rightarrow^* uAv$ for some $u, v \in (V \cup \Sigma)^*$.

True instance

G: S → SS | T

T → aTb | ab

T is usable in G

String: S → SS | T; T → aTb | ab.T over alphabet {S,T,a,b, →;|.}

T is generating: T⇒ab

T reachable from S: S ⇒ T

G: S → SS | T | TB

T → aTb | ab

B is not usable in G

G: S → SS |a

T → aTb | ab

T is not usable in G

3. **Proof of Decidability:**
   The Turing machine decides USE_{CFG}

Check if A is Generating (A can derive a string of terminals)

- Mark all terminal symbols in the grammar.
- Scan the rules of G. If a rule has variable B on the left-hand side and all symbols on the right-hand side are already marked, mark B
- Repeat this scan until no new variables can be marked.
- If A is marked at the end of this process, then A is generating. If A is not marked, reject.

Check if A is Reachable

- Mark the start variable S
- Scan the rules of G If a variable on the left-hand side is marked, mark all variables that appear on the right-hand side.
- Repeat this scan until no new variables can be marked.
- If A is marked at the end of this process, then A is reachable. A is not marked, reject.

Conclusion

If variable A passes both tests (it is both generating and reachable), the Turing machine halts and accepts. Otherwise, it halts and rejects.

Because both marking procedures operate on a finite number of variables and rules, they are guaranteed to terminate in polynomial time. Therefore, the language is decidable.

Exercise 4. Find a match in the following set of dominos

	A	B
1	1	111
2	10111	10
3	10	0

Given the domino set:

- Domino 1: $\left[\frac{1}{111}\right]$
- Domino 2: $\left[\frac{10111}{10}\right]$
- Domino 3: $\left[\frac{10}{0}\right]$

Solution (Match Sequence): $(2,1,1,3)$

Verification:

- Top string (A): $t_2\, t_1\, t_1\, t_3 = 10111 \cdot 1 \cdot 1 \cdot 10 = \mathbf{101111110}$
- Bottom string (B): $b_2\, b_1\, b_1\, b_3 = 10 \cdot 111 \cdot 111 \cdot 0 = \mathbf{101111110}$

Both strings are identical.

Exercise 5. Prove that the following is a false instance of PCP

	A	B
1	100	1
2	0	100
3	1	00

Given the domino set:

- Domino 1: $\left[\frac{100}{1}\right]$
- Domino 2: $\left[\frac{0}{100}\right]$
- Domino 3: $\left[\frac{1}{00}\right]$

Proof:

1. **Starting Domino Restriction:** A valid match must begin with a domino where both top and bottom strings share the same initial symbol.
    - Domino 1: Top starts with 1, bottom starts with 1 (Valid).
    - Domino 2: Top starts with 0, bottom starts with 1 (Cannot start).
    - Domino 3: Top starts with 1, bottom starts with 0 (Cannot start).
    - Therefore, every match sequence must start with **Domino 1**.
2. **Prefix Mismatch Analysis:**
    - After choosing Domino 1, Top = 100 and Bottom = 1.
    - Bottom requires a continuation starting with 00 to match the top prefix.
    - To append a bottom string starting with 0, the only available option is Domino 2.
    - Appending Domino 2 yields:

        - Top: 100 + 0 = 1000…
        - Bottom: 1 + 100 = 100100…
    - Comparing at index 4 (1-based): Top has symbol 0, while Bottom has symbol 1.
    - A symbol mismatch occurs immediately and cannot be resolved by appending any subsequent domino.

Hence, this instance has no match and is a **false instance**.

Exercise 6. Prove that the Travelling Salesman Problem (TSP) is in NP.

"The traveling salesman problem consists of a salesman and a set of cities. The salesman has to visit each one of the cities starting from a certain one (e.g. the hometown) and returning to the same city. Check if there is a path with total length not over than a given number k".

Language Definition:

$TSP = \{\langle G, k\rangle \mid G = (V, E, w)$ contains a Hamiltonian cycle with total weight $\leq k\}$

Proof via Polynomial-Time Verifier:

- **Certificate (c):** An ordered sequence of vertices representing the tour: $c = (v_1, v_2, \ldots, v_n, v_1)$, where $n = |V|$.
- **Verifier Algorithm V:**
    - $V =$ "On input $(\langle G, k\rangle, c)$:
        1. Verify that c contains exactly n distinct vertices from G and ends at the starting vertex v_1.
        2. Check that each consecutive pair (v_i, v_{i+1}) for $1 \leq i \leq n-1$, as well as (v_n, v_1), is an edge in E.
        3. Compute the total tour weight: $W = \sum_{i=1}^{n-1} w\,(v_i, v_{i+1}) + w(v_n, v_1)$.
        4. If $W \leq k$, **Accept**; otherwise, **Reject**."

Complexity:

- The length of certificate c is $O(n)$.
- Steps 1 through 3 execute in $O(n)$ to $O(n^2)$ time, which is strictly polynomial in the input size.

Therefore, $TSP \in NP$.

Exercise 7. Is the following formula satisfiable? (xVy) ∧ (xVy) ∧ (xVy) ∧ (xVy)

Evaluating the canonical 2-SAT formulation containing all four literal combinations:

$$\varphi = (x \vee y) \wedge (x \vee \overline{y}) \wedge (\overline{x} \vee y) \wedge (\overline{x} \vee \overline{y})$$

Evaluating all possible truth assignments for (x, y):

- $x = 0, y = 0 \Longrightarrow (x \vee y) = 0 \Longrightarrow \varphi = 0$
- $x = 0, y = 1 \Longrightarrow (x \vee \overline{y}) = 0 \Longrightarrow \varphi = 0$
- $x = 1, y = 0 \Longrightarrow (\overline{x} \vee y) = 0 \Longrightarrow \varphi = 0$
- $x = 1, y = 1 \Longrightarrow (\overline{x} \vee \overline{y}) = 0 \Longrightarrow \varphi = 0$

(If the question strictly denotes identical clauses without negation: $\varphi = (x \vee y)$, it is satisfiable under any assignment where $x = 1$ or $y = 1$).

Under the standard complete 4-clause formulation, the formula is **Unsatisfiable** (NOT satisfiable).