a) **Construct a truth table associated with the finite-state machine:**

This FSM has two states, **A** and **B**, and two inputs, **0** and **1**. We can construct a truth table that describes the transitions between these states based on the input.

Current State	Input	Next State
A	0	B
A	1	A
B	0	B
B	1	A

- **From state A**:
  - If the input is **0**, the FSM moves to state **B**.
  - If the input is **1**, it stays in state **A**.
- **From state B**:
  - If the input is **0**, the FSM stays in state **B**.
  - If the input is **1**, the FSM moves to state **A**.

---

b) **Determine the elements required to define the finite-state machine:**

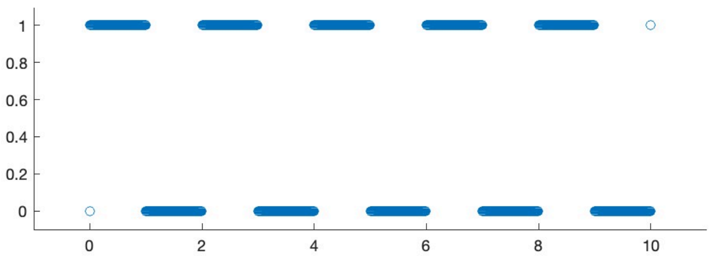
To define a finite-state machine (FSM), we need the following elements:

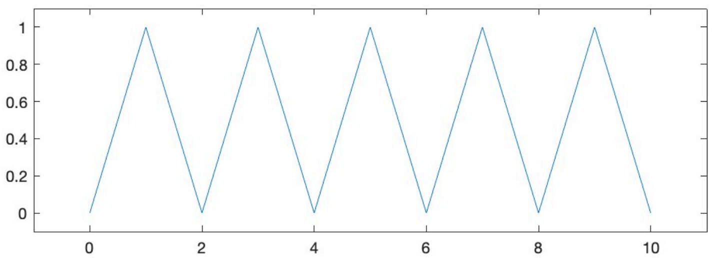
1. **States**: The different states the machine can be in.
   - In this case, the states are **A** and **B**.
2. **Input Alphabet (Σ)**: The set of possible inputs.
   - Here, the input alphabet is {**0**, **1**}.
3. **Transition Function (δ)**: A function that takes a state and an input and returns the next state.
   - From the truth table, the transition function can be defined as:
     - δ(A, 0) = B
     - δ(A, 1) = A
     - δ(B, 0) = B
     - δ(B, 1) = A
4. **Start State**: The state where the FSM begins.
   - The start state is **A** (as indicated by the arrow).
5. **Accept States (Final States)**: The states that signify the successful completion of the machine's operation.
   - The problem does not mention any accept states, so this is likely a regular FSM with no specific accept state for now.

C) **Simulation:**

The top diagram represents the output q state and lower diagram represents the input sequence.

1:11001100





2:01010101