```python
# DFA accepting strings containing three consecutive 1's.

dfa_data = {
    "alphabet": {"0", "1"},
    "input_states": {"A", "B", "C", "D"},
    "transition_table": {
        "A": {"0": "A", "1": "B"},
        "B": {"0": "A", "1": "C"},
        "C": {"0": "A", "1": "D"},
        "D": {"0": "D", "1": "D"},
    },
    "initial_state": "A",
    "final_states": {"D"},
}


class DeterministicFiniteAutomata:
    def __init__(self, **kwargs):
        self.input_states = kwargs.get("input_states")
        self.alphabet = kwargs.get("alphabet")
        self.initial_state = kwargs.get("initial_state")
        self.final_states = kwargs.get("final_states")
        self.transition_table = kwargs.get("transition_table")

    def print_components(self):
        print("=" * 24)
        print("Components:")
        print("-" * 24)
        print(f"Q: {self.input_states}")
        print(f"Σ: {self.alphabet}")
        print(f"δ: Q ⨯ Σ → Q")
        print(f"q₀: {self.initial_state}")
        print(f"F: {self.final_states}")

    def print_transition_table(self):
        print("=" * 24)
        print("Transition Table:")
        print("-" * 24)

        # Heading row
        print(f"{"δ |":<5}", end="")
        for symbol in sorted(self.alphabet):
            print(f"{symbol:<5}", end="")
        print()
        print("-" * (len(self.alphabet) + 1) * 4)

        # Data
        for state in sorted(self.input_states):
            print(f"{f"{state} |":<5}", end="")
            for symbol in sorted(self.alphabet):
                print(f"{self.transition_table[state][symbol]:<5}", end="")
            print()

    def is_accepted(self, string: str) -> bool:
        current = self.initial_state
        print(current, end="")
        for symbol in string:
            current = self.transition_table[current][symbol]
            print(f" ={symbol}⇒ {current}", end="")
        print()
        return current in self.final_states


dfa = DeterministicFiniteAutomata(**dfa_data)

dfa.print_components()
dfa.print_transition_table()


if __name__ == "__main__":
    while True:
        string = input(f"Enter a string: ")
        if string.lower() == "q":
            break
        if dfa.is_accepted(string):
            print(f"{string} is accepted")
            continue
        print(f"{string} is rejected")

```

```
❯ python practical-4.py
==========================
Components:
--------------------------
Q: {'D', 'C', 'A', 'B'}
Σ: {'0', '1'}
δ: Q ⨯ Σ → Q
q₀: A
F: {'D'}
==========================
Transition Table:
--------------------------
δ |  0    1
------------
A |  A    B
B |  A    C
C |  A    D
D |  D    D
Enter a string: 101
A ═1⇒ B ═0⇒ A ═1⇒ B
101 is rejected
Enter a string: 101110
A ═1⇒ B ═0⇒ A ═1⇒ B ═1⇒ C ═1⇒ D ═0⇒ D
101110 is accepted
Enter a string: q
```