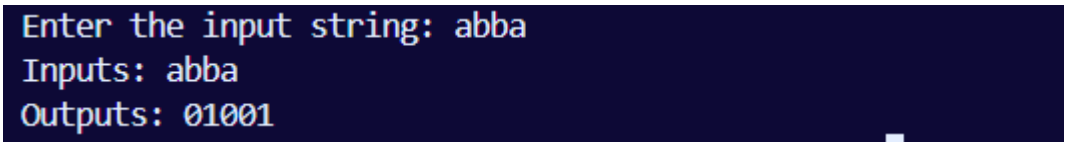
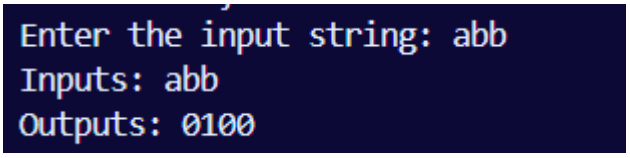
Program:

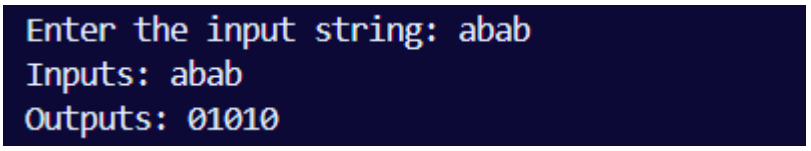
```
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

class State {
    private String name;
    private String output;
    private Map<Character, State> transitions;

    public State(String name, String output) {
        this.name = name;
        this.output = output;
        this.transitions = new HashMap<>();
    }
    public void addTransition(char input, State nextState) {
        transitions.put(input, nextState);
    }
    public State getNextState(char input) {
        return transitions.get(input);
    }
    public String getOutput() {
        return output;
    }
    @Override
    public String toString() {
        return name;
    }
}

class MooreMachine {
    private State initialState;
    private State currentState;

    public MooreMachine(State initialState) {
        this.initialState = initialState;
        this.currentState = initialState;
    }
    public void reset() {
        currentState = initialState;
    }
    public String process(String inputs) {
        StringBuilder outputs = new StringBuilder();
        for (char input : inputs.toCharArray()) {
            outputs.append(currentState.getOutput());
            currentState = currentState.getNextState(input);
```

```
        }
        outputs.append(currentState.getOutput()); // Moore machine outputs after the last
transition
        return outputs.toString();
    }

    public static void main(String[] args) {
        // Define states
        State s0 = new State("S0", "0");
        State s1 = new State("S1", "1");
        State s2 = new State("S2", "0");

        // Define transitions
        s0.addTransition('a', s1);
        s0.addTransition('b', s0);
        s1.addTransition('a', s1);
        s1.addTransition('b', s2);
        s2.addTransition('a', s1);
        s2.addTransition('b', s0);

        // Create the Moore machine
        MooreMachine mooreMachine = new MooreMachine(s0);

        // Get user input
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the input string: ");
        String inputs = scanner.nextLine();

        // Process inputs
        String outputs = mooreMachine.process(inputs);

        System.out.println("Inputs: " + inputs);
        System.out.println("Outputs: " + outputs);
    }
}
```

Output:







Observation:

The objective of this experiment is to design, implement, and analyse the behavior of a Moore machine. The primary focus is on understanding how the outputs of the machine are determined solely by the current state, independent of the current input, distinguishing it from the Mealy machine where the output is dependent on both the state and the input.

Conclusion:

The Moore machine was successfully designed and implemented, demonstrating its key characteristic of output dependency solely on the current state. The experiment validated that the Moore machine produces stable outputs associated with each state, unaffected by immediate input changes.