Practical-2

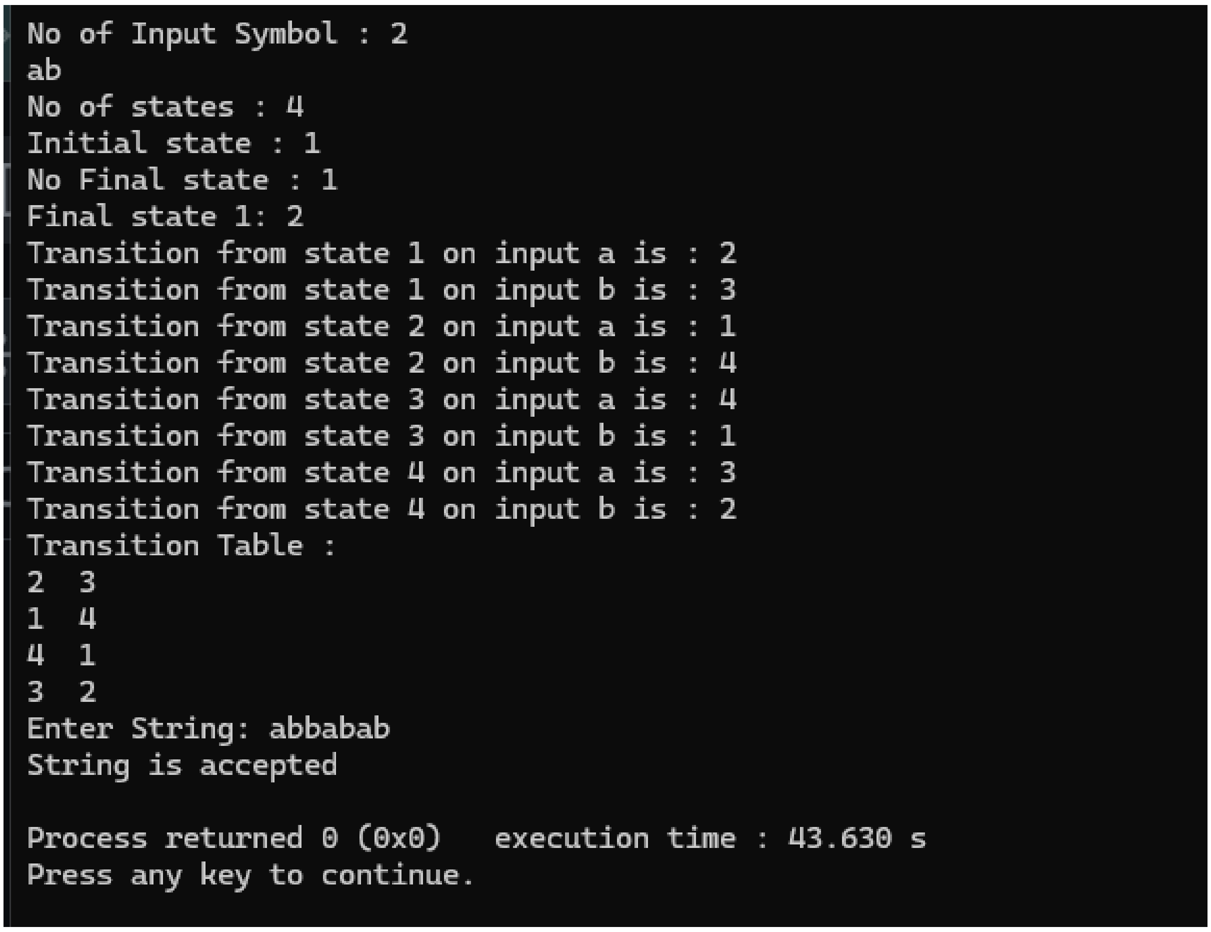
DEFINATION: String Validation Using Finite Automata

OBJECTIVE: To implement a program that validates a given string against rules defined in terms of finite automata.

CODE:

```
#include <iostream>
#include <string>
#include <unordered_map>
#include <vector>
#include <algorithm>
using namespace std;



typedef unordered_map<char, int> Transition;
typedef unordered_map<int, Transition> TransitionTable;

int main() {

    int numStates;
    int initialState;
    vector<int> acceptingStates;
    int numSymbols;
    cout<<"Enter Number of Symbols";
    cin>>numSymbols;
    cout << "Enter the number of states: ";
    cin >> numStates;
    cout << "Enter the initial state: ";
    cin >> initialState;


    int numAcceptingStates;
    cout << "Enter the number of accepting states: ";
    cin >> numAcceptingStates;
    cout << "Enter the accepting states: ";
    for (int i = 0; i < numAcceptingStates; ++i) {
        int state;
        cin >> state;
        acceptingStates.push_back(state);
    }


    TransitionTable transitionTable;
    int numTransitions=numStates*numSymbols;
    cout << "Enter transitions in the format <current_state> <input_symbol>
<next_state>:\n";
    for (int i = 0; i < numTransitions; ++i) {
        int currentState, nextState;
        char symbol;
        cin >> currentState >> symbol >> nextState;
        transitionTable[currentState][symbol] = nextState;
    }
    string input;
    cout << "Enter the input string: ";
    cin >> input;

    int currentState = initialState;

    for (char symbol : input) {
        if (transitionTable[currentState].count(symbol) == 0) {
            cout << "Rejected: Invalid input symbol \"" << symbol << "\"." << endl;
            return 0;
        }
        currentState = transitionTable[currentState][symbol];

    }



   if (find(acceptingStates.begin(), acceptingStates.end(), currentState) != 
acceptingStates.end())
 {
        cout << "Accepted" << endl;
    } else {
        cout << "Rejected" << endl;
    }

    return 0;
}
```

OUTPUT:



```
No of Input Symbol : 2
ab
No of states : 4
Initial state : 1
No Final state : 1
Final state 1: 2
Transition from state 1 on input a is : 2
Transition from state 1 on input b is : 3
Transition from state 2 on input a is : 1
Transition from state 2 on input b is : 4
Transition from state 3 on input a is : 4
Transition from state 3 on input b is : 1
Transition from state 4 on input a is : 3
Transition from state 4 on input b is : 2
Transition Table :
2  3
1  4
4  1
3  2
Enter String: abbabab
String is accepted

Process returned 0 (0x0)   execution time : 43.630 s
Press any key to continue.
```