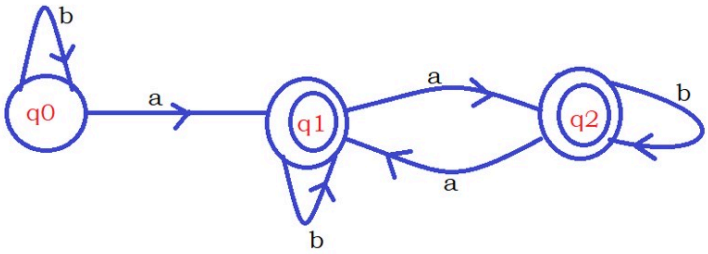
Q2.) Write a program for acceptance of string by DFA

```
#include <iostream>
#include <map>
#include <vector>
#include <unordered_set>
#include <unordered_map>

using namespace std;

class DFA{
    unordered_set<char> Q;  // Set of States
    char input_state; // Input State
    unordered_set<char> F; // Set Final States
    unordered_set<char> alphabets; // Set of alphabets
    unordered_map<char, unordered_map<char, char>> transitions;// Transitions

    public:

    DFA(){
        input("State", Q);
        input_start();
        input("Final State", F);
        input("Alphabet", alphabets);
        input_transitions();
    }

    void check_string(string &w){
        if(acceptString(w)) cout<<"String "<<w<<" is Accepted by DFA\n";
        else cout<<"String "<<w<<" is Rejected by DFA\n";
    }

    void input(string name, unordered_set<char>&input_set){
        cout<<"\nEnter number of " <<name << "s in DFA : ";
        int num;
        cin>>num;

        while(num--){
            cout<<"Enter "<<name <<" : ";
            char inp;
            cin>>inp;

            if(input_set.count(inp)){
                ++num;
                cout<<name<<" already exists \n";
            }
            else input_set.insert(inp);
        }
```

```
    }

    void input_start(){
        cout<<"Select a state from set of states to be input state : ";
        cin>>input_state;
    }

    void input_transitions(){
        cout<<"\nAnswer the destination state for each input state and input alphabet.";
        cout<<"Enter {!} if there is no transition \n";

        for(auto& start : Q){
            for(auto& alpha : alphabets){
                char dest;
                cout<<start<<" --"<<alpha<<"--> ";
                cin>>dest;

                transitions[start][alpha] = dest;
            }
        }
    }

    bool acceptString(string &w){
        int curr = input_state;

        for(auto& c : w){
            curr = transitions[curr][c];

            if(curr == '!') return false;
        }

        if(F.count(curr)) return true;
        else return false;
    }
};

int main(){
    DFA dfa;

    string input_str1 = "bbbabbaaab";
    string input_str2 = "bbbbbbbb";

    dfa.check_string(input_str1);
    dfa.check_string(input_str2);

    return 0;
}
```

Output)




```
Enter number of States in DFA : 3
Enter State : 0
Enter State : 1
Enter State : 2
Select a state from set of states to be input state : 0

Enter number of Final States in DFA : 2
Enter Final State : 1
Enter Final State : 2

Enter number of Alphabets in DFA : 2
Enter Alphabet : a
Enter Alphabet : b

Answer the destination state for each input state and input alphabet.Enter {!} if there is no transition
2 --b--> 2
2 --a--> 1
1 --b--> 1
1 --a--> 2
0 --b--> 0
0 --a--> 1
String bbbabbaaab is Accepted by DFA
String bbbbbbbb is Rejected by DFA
```