1. Exercises (70 points)

1.3, 1.4(a, c, e, f, g), 1.6, 1.7(b, c, d, e, g, h), 1.16, 1.18, 1.21, 1.22, 1.28, 1.29b.

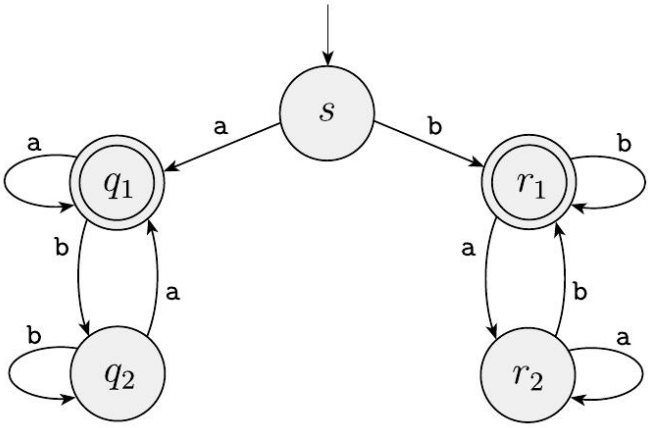
2. Programming Problems (30 points)

a) Write a Java class named DFA that represents a deterministic finite automaton (DFA). The class should allow a user to define the states and trnsitions programmatically, and the run the machine with an input string to see if it is accepted.

The DFA class suports the following operations:

- Constructor: **DFA()**
- **void addState(String state, boolean accepted)**
- **void setStartState(String state)**
- **void addTransition(String fromState, char symbol, String toState)**
- **boolean accept(String input)**: This method processes the input string character by character, following the transitions. It returns **true** if the machine ends in an accept state, and **false** otherwise (or if the input gets stuck because a transition is missing).
- **String toString()**

Test your program with the following deterministic finite automaton:



Example inputs:

- "aa" -> Accept
- "aabba" -> Accept
- "bbb" -> Accept
- "abababab" -> Reject

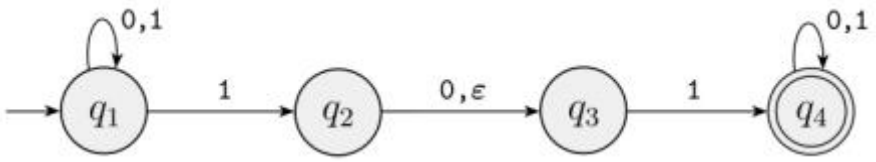
b) Write a Java class named NFA that represents a nondeterministic finite automaton (DFA). The class should allow a user to define the states and trnsitions programmatically, and the run the machine with an input string to see if it is accepted.

The NFA class suports the following operations:

- Constructor: **NFA()**
- **void addState(String state, boolean accepted)**
- **void setStartState(String state)**
- **void addTransition(String fromState, char symbol, String toState)**
- **void addEpsilonTransition(String fromState, String toState)**

- **boolean accept(String input)**: This method processes the input string character by character, following the transitions. It returns **true** if the machine ends in an accept state, and **false** otherwise (or if the input gets stuck because a transition is missing).
- **String toString()**

Test your program with the following nondeterministic finite automaton:



Example inputs:

- “11” -> Accept
- “01” -> Reject
- “101” -> Accept
- “100” -> Reject