Draw DFA transition diagram for binary strings which –

1. end with 101
2. start with 101
3. contain 101 as a substring
4. have even length. Check if your DFA accepts the string 0100.
5. have odd length
6. **contain no 0s**
7. **contain exactly one 0. Check if your DFA accepts the string 11010110**
8. **contain exactly two 0s. Check if your DFA accepts the string 11010110**
9. contain even number of 0s. Check if your DFA accepts the string 0100.
10. contain odd number of 1s
11. contain at most three 1s
12. contain at least three 1s
13. contain exactly three 1s
14. contain at most two 0s

For opposite patterns (do not …), design a DFA for the do … pattern and then reverse final and non-final states, i.e., make final states non-final and make non-final states final.

Draw DFA transition diagram and transition table for binary strings which do not –

1. end with 101. Check if your DFA accepts 011010.
2. start with 101. Check if your DFA accepts 011.
3. contain 101 as a substring
4. ~~have even length~~
5. ~~have odd length~~
6. contain no 0s
7. contain exactly one 0. Check if your DFA accepts the string 11010110
8. contain exactly two 0s. Check if your DFA accepts the string 11010110
9. contain even number of 0s
10. contain odd number of 1s
11. contain at most three 1s
12. contain at least three 1s
13. contain exactly three 1s

Draw transition diagram and transition table of a DFA for binary strings which end with 10 or 101. Then check if this DFA accepts the string 1010 using recursive definition of δ.

Solution of last part (checking via recursion):



